Near-Linear Time Approximation Algorithms for Curve Simplification *

Pankaj K. Agarwal † Sariel Har-Peled‡ Nabil H. Mustafa† Yusu Wang†

December 22, 2003

Abstract

We consider the problem of approximating a polygonal curve P under a given error criterion by another polygonal curve P' whose vertices are a subset of the vertices of P. The goal is to minimize the number of vertices of P' while ensuring that the error between P' and P is below a certain threshold. We consider two different error measures: Hausdorff and Fréchet. For both error criteria, we present near-linear time approximation algorithms that, given a parameter $\varepsilon > 0$, compute a simplified polygonal curve P' whose error is less than ε and size at most the size of an optimal simplified polygonal curve with error $\varepsilon/2$. We consider monotone curves in $\mathbb{R}^2$ in the case of Hausdorff error measure under the uniform distance metric and arbitrary curves in any dimension for the Fréchet error measure under L_p metrics. We present experimental results demonstrating that our algorithms are simple and fast, and produce close to optimal simplifications in practice.

1 Introduction

Given a polygonal curve, the curve-simplification problem asks for computing another polygonal curve that approximates the original curve based on a predefined error criterion and whose size is as small as possible. In a wide range of application areas, such as geographic information systems (GIS), computer vision, computer graphics and data compression, curve simplification is used to remove unnecessary cluttering due to excessive details, to save memory space needed to store a curve, and to expedite the processing of a curve. For example, one of the main problems in computational cartography is to visualize geo-spatial information as a simple and easily readable map. To this end, curve simplification is used to represent rivers, road-lines, coast-lines and other linear features at an appropriate level of detail when a map of a large area is being produced.

*Research by the first, the third, and the fourth authors is supported by NSF under grants CCR-00-86013 EIA-98-70724, EIA-01-31905, and CCR-02-04118, and by a grant from the U.S.–Israel Binational Science Foundation. Research by the second author is supported by NSF CAREER award CCR-0132901.

†Department of Computer Science, Box 90129, Duke University, Durham NC 27708-0129; pankaj, nabil, wys@cs.duke.edu.

‡Department of Computer Science, DCL 2111; University of Illinois; 1304 West Springfield Ave., Urbana, IL 61801; sariel@uiuc.edu.

1.1 Problem definition

Let P denote a polygonal curve in $\mathbb{R}^d$ with $\langle p_1, \ldots, p_n \rangle$ as its sequence of vertices. Define $P[p_i, p_j]$ as the polygonal curve $\langle p_i, \ldots, p_j \rangle$. A polygonal curve $P' = \langle p_{i_1}, \ldots, p_{i_k} \rangle$ *simplifies* P if $1 = i_1 < \cdots < i_k = n$. Given an error measure M and a pair of indices $1 \leq i < j \leq n$, let $\delta_M(p_i p_j, P)$ denote the error of the segment $p_i p_j$ with respect to P under the error measure M. Intuitively, $\delta_M(p_i p_j, P)$ measures how well $p_i p_j$ approximates the portion of P between p_i and p_j. The error of a simplification $P' = \langle p_{i_1} = p_1, \ldots, p_{i_k} = p_n \rangle$ of P is defined as

$$\delta_M(P', P) = \max_{1 \leq j < k} \delta_M(p_{i_j} p_{i_{j+1}}, P).$$

We call P' an *ε-simplification* of P, under the error measure M, if $\delta_M(P', P) \leq \varepsilon$. Let $\kappa_M(\varepsilon, P)$ denote the minimum number of vertices in an ε-simplification of P under the error measure M. Given a polygonal curve P, an error measure M, and a parameter ε, the *curve-simplification problem* asks for computing an ε-simplification of size $\kappa_M(\varepsilon, P)$.

In this paper, we consider two error measures:

- *Hausdorff error measure.* Let $\rho : \mathbb{R}^d \times \mathbb{R}^d \to \mathbb{R}$ be a distance function. The distance between a point p and a line segment e is defined as $d(p, e) = \min_{q \in e} \rho(p, q)$. The Hausdorff error under the metric ρ, also referred to as ρ-Hausdorff error measure, is defined as

  $$\delta_H(p_i p_j, P) = \max_{i \leq k \leq j} d(p_k, p_i p_j).$$

  We consider the Hausdorff error measure under the so-called *uniform metric* defined as

  $$\rho(p, q) = \begin{cases} |p_y - q_y| & \text{if } p_x = q_x \,, \\ \infty & \text{otherwise.} \end{cases}$$

- *Fréchet error measure.* Given two curves $f : [a, a'] \to \mathbb{R}^d$, and $g : [b, b'] \to \mathbb{R}^d$ and a distance function $\rho : \mathbb{R}^d \times \mathbb{R}^d$, the Fréchet distance $\mathcal{F}(f, g)$ between them is defined as

  $$\mathcal{F}(f, g) \quad = \quad \inf_{\substack{\alpha : [0,1] \to [a, a'] \\ \beta : [0,1] \to [b, b']}} \max_{t \in [0,1]} \rho(f(\alpha(t)), g(\beta(t))) \tag{1}$$

  where α and β range over continuous and monotonically non-decreasing functions with $\alpha(0) = a, \alpha(1) = a', \beta(0) = b$ and $\beta(1) = b'$. If $\hat{\alpha}$ and $\hat{\beta}$ are the maps that realize the Fréchet distance, then we refer to the map $\phi = \hat{\beta} \circ \hat{\alpha}^{-1}$ as the *Fréchet map* from f to g. For a pair of indices $1 \leq i < j \leq n$, the *Fréchet error* of a line segment $p_i p_j$ is defined to be

  $$\delta_F(p_i p_j, P) = \mathcal{F}(p_i p_j, P[p_i, p_j]),$$

  where $P[p, q]$ denotes the portion of P from p to q. Given curves P and P' (its vertices need not be a subset of the vertices of P), we say that P' is a *weak ε-simplification* of P if $\mathcal{F}(P, P') \leq \varepsilon$. Let $\hat{\kappa}_F(\varepsilon, P)$ denote the minimum number of vertices in a weak ε-simplification of P.

1.2 Previous results

The problem of approximating a polygonal curve P has been studied extensively during the last two decades; see [11, 16] for surveys. Imai and Iri [14] formulated the curve-simplification problem as computing a shortest path between two nodes in a directed acyclic graph G_P: each vertex of P corresponds to a node in G_P, and there is an edge between two nodes p_i, p_j if $\delta_M(p_ip_j, P) \leq \varepsilon$. A shortest path from p_1 to p_n in G_P corresponds to an optimal ε-simplification of P under the error measure M. In $\mathbb{R}^2$, under the Hausdorff measure with uniform metric, their algorithm takes $O(n^2 \log n)$ time. Chin and Chan [7], and Melkman and O'Rourke [15] improve the running time of their algorithm to quadratic. Agarwal and Varadarajan [4] improve the running time to $O(n^{4/3+\delta})$ for L_1- and uniform-Hausdorff error measures , for an arbitrarily small constant $\delta > 0$, by implicitly representing the graph G_P. In dimensions higher than two, Barequet *et al.* compute the optimal ε-simplification under L_1- or L_∞-Hausdorff error measures in quadratic time [6]. For L_2-Hausdorff error measure, an optimal simplification can be computed in near-quadratic time for $d = 3$ and in $O(n^{3-2/(\lfloor \frac{d}{2} \rfloor + 1)}\text{polylog}n)$ in $\mathbb{R}^d$ for $d > 3$.

Curve simplification using the Fréchet error measure was first proposed by Godau [9], who showed that $\kappa_F(\varepsilon, P) \leq \hat{\kappa}_F(\varepsilon/7, P)$. Alt and Godau [5] also proposed an $O(mn)$-time algorithm to determine whether $\delta_F(P, Q) \leq \varepsilon$ for given polygonal curves P and Q of size n and m, respectively, and for a given error measure $\varepsilon > 0$. Following the approach of Imai and Iri [14], an ε-simplification of P, under the Fréchet error measure, of size $\kappa_F(\varepsilon, P)$ can be computed in $O(n^3)$ time.

The problem of developing a near-linear algorithm for computing an optimal ε-simplification remains elusive. Among the several heuristics that have been proposed over the years, the most widely used is the Douglas-Peucker method [8] (together with its variants). Originally proposed for simplifying curves under the Hausdorff error measure, its worst-case running time is $O(n^2)$ in $\mathbb{R}^d$. For $d = 2$ the running time is improved by Snoeyink *et al.* [12] to $O(n \log n)$. However, the Douglas-Peucker heuristic does not offer any guarantee on the size of the simplified curve—it can return an ε-simplification of size $\Omega(n)$ even if $\kappa_H(\varepsilon, P) = O(1)$.

Much work has been done on computing a weak ε-simplification of a polygonal curve P. Imai and Iri [13] give an optimal $O(n)$-time algorithm for finding an optimal weak ε-simplification (under the Hausdorff error measure) of a x-monotone curve in $\mathbb{R}^2$. As for weak ε-simplification of planar curves under Fréchet distance, Guibas *et al.* [10] proposed an $O(n \log n)$-time factor-2 approximation algorithm and an $O(n^2)$-time exact algorithm. They also proposed linear-time algorithms for approximating some other variants of weak simplifications.

1.3 Our results

Let P be a polygonal curve in $\mathbb{R}^d$, and let $\varepsilon > 0$ be a parameter. We present simple, near-linear algorithms for computing ε-simplifications of P size at most $\kappa_M(\varepsilon/c, P)$, where $c \geq 1$ is a constant. We first develop an algorithm for an x-monotone polygonal curve in $\mathbb{R}^2$ under the Hausdorff error with respect to uniform metric.

Theorem 1.1 *Let P be a x-monotone polygonal curve in $\mathbb{R}^2$ with n vertices, and let $\varepsilon > 0$ be a parameter. We can compute in $O(n)$ time a simplification P' of P of size at most $\kappa_H(\varepsilon/2, P)$ so*

that $\delta_H(P, P') \leq \varepsilon$, assuming that the distance between points is measured in uniform metric.

We have implemented the algorithm in $\mathbb{R}^2$. Experimental results demonstrate that our algorithm computes simplifications of small size, and it is faster than the widely used Douglas-Peucker algorithm.

Our second result is under the Fréchet-error measure. Unlike the previous case, we do not assume P to be monotone, and the algorithm extends to higher dimensions.

Theorem 1.2 *Let P be a polygonal curve in $\mathbb{R}^d$ with n vertices, and let $\varepsilon > 0$ be a parameter. We can compute in $O(n \log n)$ time a simplification P' of P of size at most $\kappa_F(\varepsilon/2, P)$ so that $\delta_F(P', P) \leq \varepsilon$, assuming that the distance between points is measured in any L_p-metric.*

To our knowledge, this is the first simple, near-linear approximation algorithm for curve simplification under the Fréchet error measure that extends to $d > 2$. We illustrate its simplicity and efficiency by comparing its performance with Douglas-Peucker and exact algorithms. Our experimental results on various data sets show that our algorithm is efficient and produces ε-simplifications of near-optimal size.

Finally, we analyze the relationship between simplification under Hausdorff and Fréchet error measures. We note that $\kappa_H(\varepsilon, P) \leq \kappa_F(\varepsilon, P)$ under any L_p-metric. We show that $\kappa_F(\varepsilon, P) \leq \hat{\kappa}_F(\varepsilon/4, P)$, thereby improving the result by Godau [9].

2 Hausdorff Simplification

Let $P = \langle p_1, \ldots, p_n \rangle$ be a x-monotone polygonal curve in $\mathbb{R}^2$, and let $\varepsilon > 0$ be a parameter. Let $D(p, r)$ be the disk of radius r centered at p under uniform metric, i.e. $D(p, r)$ is a vertical segment of length $2r$ with p as its midpoint. A segment $p_j p_k$ is a valid segment, i.e. $\delta_H(p_j p_k, P) \leq \varepsilon$, if and only if $p_j p_k$ intersects the segments $D(p_i, \varepsilon)$ for all $j \leq i \leq k$.

Lemma 2.1 *Given an x-monotone polygonal curve P in $\mathbb{R}^2$ and four indices $1 \leq i \leq l < r \leq j \leq n$, $\delta_H(p_l p_r, P) \leq 2 \cdot \delta_H(p_i p_j, P)$.*

PROOF. Let $\varepsilon' = \delta_H(p_i p_j, P)$. Let q_k (resp. q'_k) be the intersection point of $p_i p_j$ (resp. $p_l p_r$) with the vertical line passing through p_k if such an intersection point exists. By definition, $q_k \in D(p_k, \varepsilon')$ for all $i \leq k \leq j$. Hence $|p_l q_l| \leq \varepsilon'$ and $|p_r q_r| \leq \varepsilon'$, and it follows that $|q_k q'_k| \leq \varepsilon'$ since P is an x-monotone curve. Therefore by triangle inequality, we have that

$$|p_k q'_k| \leq |p_k q_k| + |q_k q'_k| \leq 2\varepsilon'$$

See Figure 1 for an illustration. □

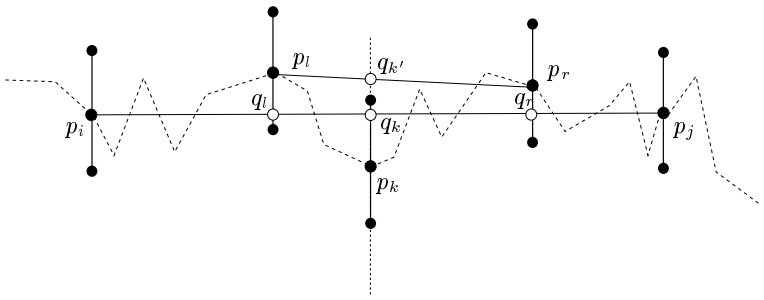


Figure 1. Dashed chain is the original curve. p_ip_j stabs $D(p_l, \varepsilon')$, $D(p_k, \varepsilon')$, and $D(p_r, \varepsilon')$ (vertical segments) in order. Segment p_lp_r stabs $D(p_k, 2\varepsilon')$ (dotted vertical segment), for any $l < k < r$.

2.1 Algorithm

We compute an ε-simplification P' of P using a greedy approach. Suppose we have added $p_{i_1}, \ldots p_{i_{j-1}}$ to P'. We find the smallest index $k > i_{j-1}$ so that $\delta_H(p_{i_{j-1}}p_{k+1}, P) > \varepsilon$. We set $i_j = k$ and add p_{i_j} to P'. We repeat this process until p_n is encountered. We add p_n to the sequence and return the resulting sequence of vertices P' as the desired simplification.

Given the vertex $p_{i_{j-1}} = p_l$, we find p_k in time $O(k - l + 1)$ as follows. Let $\mathrm{Cone}(p_i, p_j)$ be the set of rays emanating from p_i that intersect the vertical segment $D(p_j, \varepsilon)$. $\mathrm{Cone}(p_i, p_j)$ is the cone bounded by rays emanating from p_i and passing through the endpoints of $D(p_j, \varepsilon)$. Set

$$C(i, j) = \bigcap_{r=i+1}^{j} \mathrm{Cone}(p_i, p_r)$$

Then $\delta_H(p_ip_j, P) \leq \varepsilon$ if and only if $p_j \in C(i, j)$. $C(i, r) = C(i, r - 1) \cap \mathrm{Cone}(p_i, p_r)$ can be computed in $O(1)$ from $C(i, r - 1)$. To compute p_k, we visit the vertices p_r of P one by one, starting from p_{l+1} and maintain $C(l, r)$ by spending $O(1)$ time at p_r. We stop as soon as we reach a vertex p_{k+1} such that $p_{k+1} \notin C(l, k + 1)$. The total time spent in computing p_k is $O(k - l + 1)$. Hence the total running time is $O(n)$.

Lemma 2.2 *P' is an ε-simplification of P whose size is at most $\kappa_H(\varepsilon/2, P)$.*

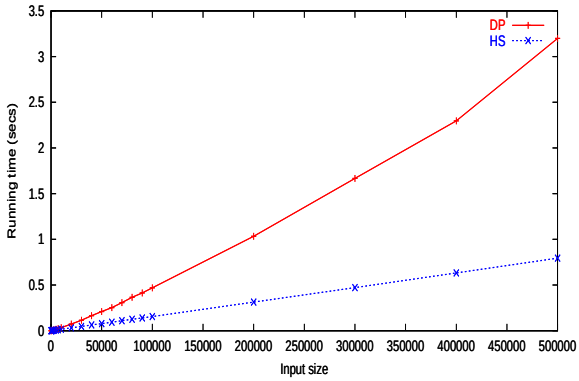
PROOF. It is clear that $\delta_H(P', P) \leq \varepsilon$. We bound the size of P' by induction on the number of vertices in P'. Let $P' = \langle p_{i_1}, \ldots, p_{i_k} \rangle$, and let an optimal $(\varepsilon/2)$-simplification be $P^* = \langle p_{j_1}, \ldots, p_{j_l} \rangle$. We prove the following claim: $i_m \geq j_m$ for all $1 \leq m \leq k$, which then implies the lemma. This claim is clearly true for $m = 1$ since $p_{i_1} = p_{j_1} = p_1$. Assume it is true for $m - 1$. If $i_{m-1} \geq j_m$, we are done as $i_m \geq i_{m-1} \geq j_m$. Otherwise, $\delta_H(p_{j_{m-1}}p_{j_m}, P) \leq \varepsilon/2$, and Lemma 2.1 implies that $\delta_H(p_{i_{m-1}}p_r, P) \leq \varepsilon$ for any $i_{m-1} \leq r \leq j_m$. Hence $i_m \geq j_m$, as required. □

This completes the proof of Theorem 1.1.

Remark 2.3 Our algorithm does not compute an ε-simplification of size at most $\kappa_H(\varepsilon/2, P)$ if we measure the distance between points in an L_p-metric because the observation that $\delta_H(p_ip_j, P) \leq \varepsilon$ if and only if p_ip_j intersects $D(p_k, \varepsilon)$, for every $i \leq k \leq j$ no longer holds.

Size	HS	DP
500	106	92
1000	227	208
5000	1111	1020
10000	2156	2116
20000	4454	4158
40000	9118	8402
80000	17676	17284
100000	21934	21504

(a)



(b)

Figure 2. Comparison between our (HS) and Douglas-Peucker (DP) algorithm on a sine curve P; $\varepsilon = 0.6$. (a) Output size as a function of the size of P, (b) running time as a function of the size of P.

2.2 Experiments

We have implemented our algorithm, called HS, on a SUN Blade-100 workstation with 500 MHz CPU speed and 256MB RAM, running Sun OS 5.8. We also implemented the Douglas-Peucker algorithm (DP), a widely used algorithm for curve simplification. We compare the performance of our algorithm with the DP algorithm in terms of the output size and the running time. We use the following two data sets.

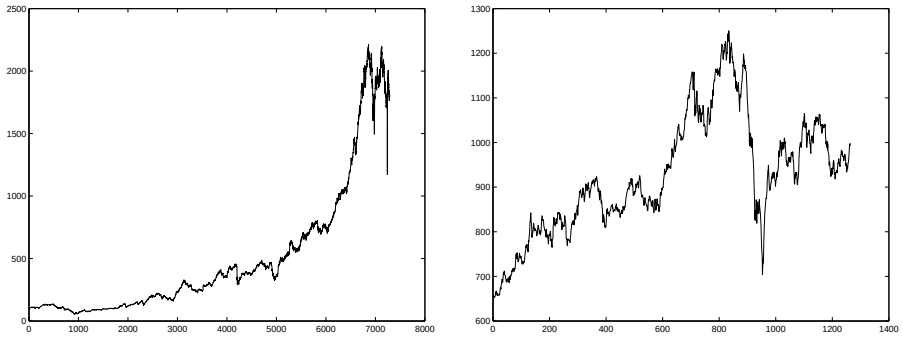
Synthetic data. Points are sampled uniformly on a sinusoidal curve in the angular interval $[0, \pi]$. More precisely, given a value of n, we choose the vertices to be

$$P = \langle (i, n \sin(i\pi/n)) \mid 1 \leq i \leq n \rangle.$$

We note that the sinusoidal curve is one of the most favorable curves for the Douglas-Peucker algorithm in terms of its running time. The sinusoidal curve is convex in the interval $[0, \pi]$. For a segment $p_i p_j$, the vertex p_k of P farthest from $p_i p_j$ is roughly in the middle of p_i and p_j. The curve is partitioned at p_k and each portion of the curve is simplified recursively. The running time of the Douglas-Peucker algorithm is therefore $O(n \log n)$. Figure 2(b) confirms this hypothesis. Figure 2(b) also exhibits linear running time of our algorithm, outperforming the Douglas-Peucker algorithm. The constant of proportionality in $O(n)$ of the running time of our algorithm is small, as is evident from the low slope of the curve depicting the running time of our algorithm.

Figure 2(a) shows the output size of the simplification computed by our and Douglas-Peucker algorithms. Since we choose $\varepsilon = 0.6$, irrespective of the value of n, we are in effect computing finer simplifications as we increase the value of n. This is why the output size increases with n. The size of the simplifications computed by our algorithm is very close to that of the one computed by the Douglas-Peucker algorithm.

Time-series data. We choose four NASDAQ stock indices: Composite index, industry index, telecommunications index, and biotech index. The data was obtained from [1]. Each of the first two



(a) (b)

Figure 3. (a) Composite-index curve ($7,289$ vertices), (b) Telecommunications-index curves ($1,263$ vertices).

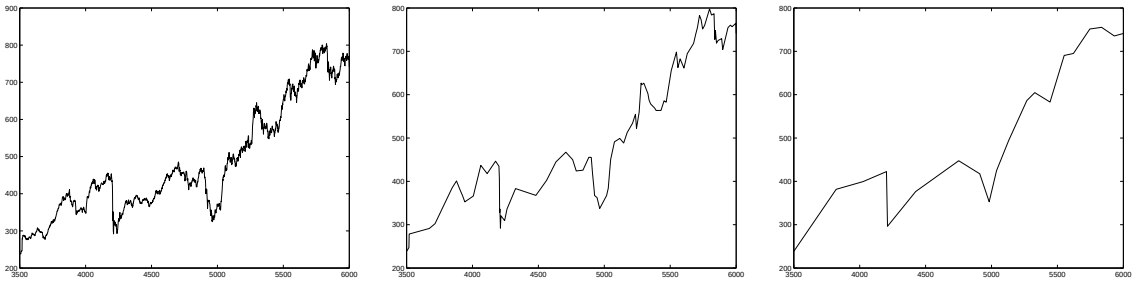
	Industry-Index (7,289)		*Composite-Index (7,289)*		*Telecomm-Index (1,263)*		*Biotech-Index (1,263)*	
ε	HS	Exact	HS	Exact	HS	Exact	HS	Exact
0.05	6990	6989	6913	6905	1254	1254	1260	1260
0.10	6711	6695	6568	6543	1245	1245	1255	1253
0.50	5117	4962	4727	4511	1164	1161	1226	1225
1.00	4011	3765	3596	3327	1089	1087	1178	1178
3.00	2275	2001	1917	1645	812	769	1015	999
5.00	1636	1394	1333	1113	598	530	856	826
10.0	961	784	741	597	349	253	658	600
20.0	520	397	367	261	147	101	445	386
50.0	190	123	113	70	37	22	213	159

Figure 4. Size of ε-simplifications of stock-index curves computed using HS and the optimal algorithms.

curves have $7,289$ vertices, corresponding to the daily index values from January 1971 till January 2003. Each of the last two curves have $1,263$ vertices, corresponding to the daily index values from January 1993 till January 2003. Figure 3 shows the composite-index and telecommunications-index curves. Note that while the sizes of the stock time-series curves are not very large, simplification can be used to understand the general trend of the stock data over time (Figure5 illustrates this fact nicely).

Figure 4 gives the size of ε-simplifications for various values of ε computed by the two algorithms on these four curves. Since the curves in Figure 3 are too dense and it is hard to see all the details, we magnify a portion of the composite-index of the curve and show its simplifications (as part of the simplification of the entire curve) in Figure 5.

The size of the simplifications produced by our algorithm are quite close to the size of the optimal simplifications. The running time of our algorithm for the first two index curves (with $7,289$ vertices) varies between 12 and 18 milliseconds, and the running time for the next two index



(a) (b) (c)

Figure 5. (a) Portion of the composite-index curve ($2,500$ vertices), and its simplifications with (b) $\varepsilon = 20.0$ (88 vertices) (c) $\varepsilon = 50.0$ (20 vertices).

curves (with $1,263$ vertices) varies between 1 and 3 milliseconds. The optimal algorithm runs in $O(n^2)$ time, and takes between 2 to 5 minutes on these datasets.

3 Fréchet Simplification

Let $P = \langle p_1, \ldots, p_n \rangle$ be a polygonal curve in $\mathbb{R}^d$, and let $\varepsilon > 0$ be a parameter. Unlike Section 2, we do not assume P to be x-monotone. We present an approximation algorithm for simplification under the Fréchet error measure, but we first prove a few properties of the Fréchet error measure. Let $\mathcal{F}(\cdot, \cdot)$ be as defined in (1), and let $d(\cdot, \cdot)$ denote the Euclidean distance between two points in $\mathbb{R}^d$.

Lemma 3.1 *Given two directed segments uv and xy in $\mathbb{R}^d$,*

$$\mathcal{F}(uv, xy) = \max\{d(u, x), d(v, y)\}.$$

PROOF. Let $\delta = \max\{d(u, x), d(v, y)\}$. Then $\mathcal{F}(uv, xy) \geq \delta$, since u (resp. v) has to be matched to x (resp. y). Assume the natural parameterization for segment uv, $A(t) : [0, 1] \rightarrow uv$, such that $A(t) = u + t(v - u)$. Similarly, define $B(t) : [0, 1] \rightarrow xy$ for segment xy, such that $B(t) = x + t(y - x)$. For any two matched points $A(t)$ and $B(t)$, let

$$C(t) = A(t) - B(t) = (1 - t)(u - x) + t(v - y).$$

Since $C(t)$ is a convex function, $\|C(t)\| \leq \delta$ for any $t \in [0, 1]$. Therefore $\mathcal{F}(uv, xy) \leq \delta$. □

Lemma 3.2 *Given a polygonal curve P in $\mathbb{R}^d$ and two directed segments uv and xy,*

$$| \mathcal{F}(uv, P) - \mathcal{F}(xy, P) | \leq \mathcal{F}(uv, xy).$$

PROOF. Assume that $A(t) : [0,1] \to uv$ is the natural parameterization of uv, such that $A(t) = u + t(v-u)$. Let $P(t)$, $t \in [0,1]$, be a parameterization of the polygonal curve P such that $P(t)$ and $A(t)$ realize the Fréchet distance between P and uv. As in the proof of Lemma 3.1, let $B(t) : [0,1] \to xy$ be such that $B(t) = x + t(y-x)$. By triangle inequality, for any $t \in [0,1]$, $d(P(t), B(t)) \leq d(P(t), A(t)) + d(A(t), B(t))$, yielding the lemma. □

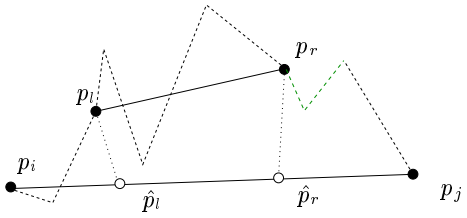


Figure 6. Dashed curve is $P[p_i, p_j]$, p_l and p_r are mapped to $\hat{p}_l$ and $\hat{p}_r$ on $p_i p_j$, respectively.

Lemma 3.3 *Let $P = \langle p_1, p_2, \ldots, p_n \rangle$ be a polygonal curve in $\mathbb{R}^d$. For $i \leq l \leq r \leq j$,*

$$\delta_F(p_l p_r, P) \leq 2 \cdot \delta_F(p_i p_j, P).$$

PROOF. Let $\delta^* = \delta_F(p_i p_j, P)$. Let $\alpha : P[p_i, p_j] \to p_i p_j$ be the Fréchet map from $p_i p_j$ to $P[p_i, p_j]$ (see Section 1.1 for definition). For any vertex $p_k \in P[p_i, p_j]$, set $\hat{p}_k = \alpha(p_k)$; see Figure 6 for an illustration. By definition, $\mathcal{F}(\hat{p}_l \hat{p}_r, P[p_l, p_r]) \leq \delta^*$. In particular, $d(p_l, \hat{p}_l), d(p_r, \hat{p}_r) \leq \delta^*$. By Lemma 3.1, $\mathcal{F}(p_l p_r, \hat{p}_l \hat{p}_r) \leq \delta^*$. It then follows from Lemma 3.2 that

$$\delta_F(p_l p_r, P) = \mathcal{F}(p_l p_r, P[p_l, p_r]) \leq \mathcal{F}(\hat{p}_l \hat{p}_r, P[p_l, p_r]) + \delta^* \leq 2\delta^*.$$

□

3.1 Algorithm

As in Section 2.2, we use a greedy algorithm to compute a simplification P' of P, but with one twist. Suppose we have added $\langle p_1 = p_{i_1}, \ldots, p_{i_j} \rangle$ to P'. We find an index $k > i_j$ such that (i) $\delta_F(p_{i_j} p_k, P) \leq \varepsilon$ and (ii) $\delta_F(p_{i_j} p_{k+1}, P) > \varepsilon$. We set $i_{j+1} = k$ and add $p_{i_{j+1}}$ to P'. We repeat this process until we encounter p_n. We then add p_n to P'.

Alt and Godau [5] have developed an algorithm that, given a pair $1 \leq i < j \leq n$, can determine in $O(j-i)$ time whether $\delta_F(p_i p_j, P) \leq \varepsilon$. Therefore, a first approach would be to add vertices greedily one by one, starting with the first vertex p_{i_j}, and testing each edge $p_{i_j} p_l$, for $l > i_j$, by invoking the Alt-Godau algorithm, until we find the index k. However, the overall algorithm could take $O(n^2)$ time. To limit the number of times that Alt-Godau algorithm is invoked when computing the index k, we proceed as follows.

First using an exponential search, we determine an integer $l \geq 0$ such that $\delta_F(p_{i_j} p_{i_j + 2^l}, P) \leq \varepsilon$ and $\delta_F(p_{i_j} p_{i_j + 2^{l+1}}, P) > \varepsilon$. Next, by performing a binary search in the interval $[2^l, 2^{l+1}]$, we determine an integer $r \in [2^l, 2^{l+1}]$ such that $\delta_F(p_{i_j} p_{i_j + r}, P) \leq \varepsilon$ and $\delta_F(p_{i_j} p_{i_j + r + 1}, P) > \varepsilon$. Note

```
ALGORITHM FS (P, ε)
  Input: P = ⟨p_1, ··· , p_n⟩; ε > 0.
  Output: P' ⊆ P such that δ_F(P', P) ≤ ε.
begin
  j = 1; i_j = 1; P' = ⟨p_{i_j}⟩ ;
  while ( i_j < n ) do
    l = 0;
    while (δ_F(p_{i_j} p_{i_j + 2^{l+1}}, P) ≤ ε) do
      l = l + 1 ;
    end while
    low = 2^l; high = 2^{l+1};
    while (low < (high − 1)) do
      mid = (low + high)/2;
      if ( δ_F(p_{i_j} p_{i_j + mid}, P) ≤ ε)
        low = mid;
      else high = mid;
    end while
    i_{j+1} = i_j + low; P' = P' · ⟨p_{i_{j+1}}⟩; j = j + 1 ;
  end while
end
```

Figure 7. Computing ε-simplification under the Fréchet error measure.

that in the worst case, the asymptotic costs of the exponential and binary searches are the same. Set $k = i_j + r$. See Figure 7 for pseudo-code of this algorithm. Since computing the value of k requires invoking the Alt-Godau algorithm $O(l) = O(\log n)$ times, each with a pair (α, β) such that $\beta - \alpha \leq 2r$, the total time spent in computing the value of k is $O((k - i_j + 1) \log n)$. Hence, the overall running time of the algorithm is $O(n \log n)$.

Theorem 3.4 *Given a polygonal curve $P = \langle p_1, \ldots, p_n \rangle$ in $\mathbb{R}^d$ and a parameter $\varepsilon > 0$, we can compute in $O(n \log n)$ time an ε-simplification P' of P under the Fréchet error measure so that $|P'| \leq \kappa_F(\varepsilon/2)$.*

PROOF. Compute P' by the greedy algorithm described above. By construction $\delta_F(P', P) \leq \varepsilon$, so it suffices to prove that $|P'| \leq \kappa_F(\varepsilon/2, P)$. Let $P' = \langle p_{i_1} = p_1, \ldots, p_{i_k} = p_n \rangle$, and let $P^* = \langle p_{j_1} = p_1, \ldots, p_{j_l} = p_n \rangle$ be an optimal $(\varepsilon/2)$-simplification of P of size $\kappa_F(\varepsilon/2, P)$. We claim that $i_m \geq j_m$ for all m. This would imply that $k \leq l \leq \kappa_F(\varepsilon/2, P)$.

We prove the above claim by induction on m. For $m = 1$, the claim is obviously true because $i_1 = j_1 = 1$. Suppose $i_{m-1} \geq j_{m-1}$. If $j_m \leq i_{m-1}$, we are done. So assume that $j_m > i_{m-1}$. Since P^* is an $(\varepsilon/2)$-simplification, $\delta_F(p_{j_{m-1}} p_{j_m}, P) \leq \varepsilon/2$. Lemma 3.3 implies that for all $i_{m-1} \leq j' \leq j_m$, $\delta_F(p_{i_{m-1}} p_{j'}, P) \leq \varepsilon$. But by construction, $\delta_F(p_{i_{m-1}} p_{i_m+1}, P) > \varepsilon$, therefore $i_m + 1 > j_m$ and thus $i_m \geq j_m$. ∎

Remark 3.5 Our algorithm works within the same running time even if we measure the distance between two points in any L_p-metric.

3.2 Experiments

We have implemented our simplification algorithm, which we refer to as `FS`, and the $O(n^3)$-time optimal Fréchet-simplification algorithm (referred to as `Exact`), outlined in Section 1.2, that computes an ε-simplification of P of size $\kappa(\varepsilon, P)$. In this section, we measure the performance of our algorithm in terms of the output size and the running time.

Data sets. We test our algorithms on two different types of data sets, each of which is a family of polygonal curves in $\mathbb{R}^3$.

- *Protein backbones.* The first set of curves are derived from protein backbones by adding small random "noise" vertices along the edges of the backbone. We have chosen two backbones from the protein data bank [3]: PDB file ids $1cja$ and $1cdk$. The number of vertices in the original backbones of $1cja$ and $1cdk$ are 327 and 343, respectively. Protein A is the original protein $1cja$. ProteinB is $1cdk$ with $9,434$ random vertices added: the new vertices are uniformly distributed along the curve edges, and then perturbed slightly in a random direction.

- *Stock-index curves.* The second family of curves is generated from the daily NASDAQ index values over the period January 1971 to January 2003 (data is obtained from [1]). We take a pair of index curves $\{(x_i, t) | 1 \leq i \leq n\}$ and $\{(y_i, t) | 1 \leq i \leq n\}$ and generate the curve $\{(x_i, y_i, t) | 1 \leq i \leq n\}$ in $\mathbb{R}^3$. In particular, we take telecommunication index and Biotechnology index as the x- and y-coordinates and time as the z-coordinates to construct the curve Tel-bio in $\mathbb{R}^3$. In the second case, we take transportation, telecommunication index, and time as x-, y-, and z-coordinates, respectively, to construct curve Trans-Tel in $\mathbb{R}^3$.

These two families of curves have different structures. The stock-index curves exhibit an easily distinguishable global trend; however, locally there is a lot of noise. The protein curves, though coiled and irregular-looking, exhibit local patterns that represent the structural elements of the protein backbones (commonly referred to as the *secondary structure*). In each of these cases, simplification helps identify certain patterns (e.g., secondary structure elements) and trends in the data.

Output size. We compare the quality (size) of simplifications produced by our algorithm (`FS`) and optimal algorithm (`Exact`) in Figure 8 for curves from the above two families respectively. The simplifications produced by our algorithm are almost always within 5% of the optimal.

For monotone curves, the simplifications produced by algorithm `HS` (using Hausdorff error measure) and `FS` (using Fréchet error measure) are similar, as shown in Figure 9. Note that `HS` is using uniform metric and `FS` is under L_2 metric, which accounts for the smaller size of the simplifications computed by `FS`.

To provide a visual picture of the simplification produced by various (commonly used) algorithms for curves in $\mathbb{R}^3$, Figure 14 shows the simplifications of protein A computed by the `FS`, exact

Output size				
	ProteinA (327)		*ProteinB (9,777)*	
ε	FS	Exact	FS	Exact
0.05	327	327	6786	6431
0.12	327	327	1537	651
1.20	254	249	178	168
1.60	220	214	140	132
2.00	134	124	115	88
5.00	37	36	41	39
10.0	22	22	24	20
20.0	10	8	8	6
50.0	2	2	2	2

(a)

Output size				
	Trans-Tel (7,057)		*Tel-Bio (1,559)*	
ε	FS	Exact	FS	Exact
0.05	6882	6880	1558	1558
0.50	4601	4469	1473	1471
1.20	2811	2637	1292	1279
3.00	1396	1228	974	942
5.00	890	732	772	720
10.0	414	329	490	402
20.0	168	124	243	200
50.0	47	35	94	73

(b)

Figure 8. The sizes of Fréchet simplifications on (a) protein data and (b) stock-index data.

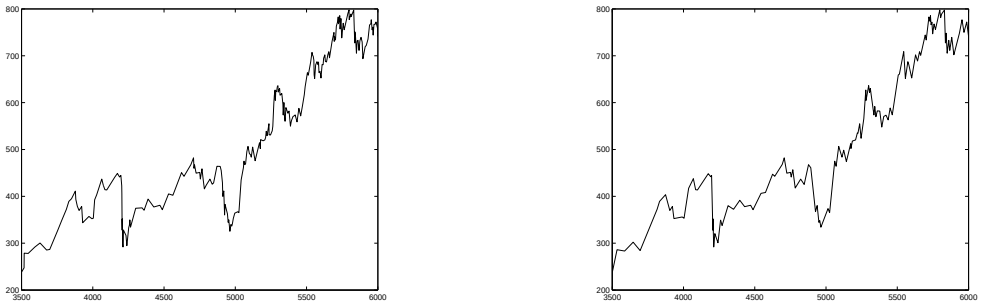


Figure 9. Simplify a portion of the composite-index curve (with $2,500$ vertices) using algorithm HS(left) and FS (right) for error $\varepsilon = 10.0$. The simplified curve has 224 and 125 vertices respectively.

(i.e., optimal) Fréchet simplification algorithm, and the Douglas-Peucker heuristic (using Hausdorff error measure under L_2 metric).

Running time. As the running time for the optimal algorithm is orders of magnitude slower than our algorithm, we compare the efficiency of our algorithm (FS) with the widely used Douglas-Peucker simplification algorithm under the Hausdorff measure – we can extend the Douglas-Peucker algorithm to simplify curves under the Fréchet error measure; however, such an extension is inefficient and can take $\Omega(n^3)$ time in the worst case.

Figures 10 illustrates the running time of the two algorithms. Note that as ε increases, resulting in small simplified curves, the running time of Douglas-Peucker decreases. This phenomenon is further illustrated in Figure 11, which compares the running time of our algorithm with the Douglas-Peucker on Protein B (with artificial noise added) with $9,777$ vertices. This phenomenon is due

Running time (ms.)				
	ProteinA (327)		*ProteinB (9,777)*	
ε	FS	DP	FS	DP
0.05	3	16	146	772
0.50	3	16	171	524
1.20	4	16	176	488
1.60	5	12	202	394
2.00	5	11	210	354
5.00	5	11	209	356
10.0	5	10	222	329
20.0	5	8	233	263
50.0	2	1	87	50

(a)

Running time (ms.)				
	Trans-Tel (7,057)		*Tel-Bio (1,559)*	
ε	FS	DP	FS	DP
0.05	82	599	16	113
0.50	103	580	17	114
1.20	113	559	19	113
3.00	119	510	22	109
5.00	121	472	24	109
10.0	127	411	25	96
20.0	146	360	27	85
50.0	162	271	27	71

(b)

Figure 10. The running time of FS and Douglas-Peucker algorithm on (a) protein data and (b) stock-index data.

to the fact that, at each step, the DP algorithm determines whether a line segment p_ip_j simplifies $P[p_i, p_j]$. The algorithm recursively solves two subproblems only if $\delta_H(p_ip_j, P) > \varepsilon$. Thus, as ε increases, it needs to make fewer recursive calls. Our algorithm, however, proceeds in a linear fashion from the first vertex to the last vertex using exponential and binary search. Suppose the algorithm returns $P' = \langle p_{j_1}, \ldots, p_{j_k} \rangle$ for an input polygonal curve $P = \langle p_1, \ldots, p_n \rangle$. The exponential search takes $O(n)$ time, while the binary search takes $\sum_{i=1}^{k-1} O(n_i \log n_i)$ where $n_i = j_{i+1} - j_i$ and k is the number of vertices of P'. Thus as ε increases, n_i increases, and therefore the time for binary search increases, as Figure 11 illustrates. Note however that if ε is so large that $k = 2$, i.e., the simplification is just one line segment connecting p_1 to p_n, the algorithm does not perform any binary search and is much faster, as the case for $\varepsilon = 50.0$ illustrates in Figure 11.

4 Comparisons

In this section, we prove relationships between various error measures.

Hausdorff vs. Fréchet. One natural question is to compare the quality of simplifications produced under the Hausdorff- and the Fréchet- error measures. Given a curve $P = \langle p_1, \ldots, p_n \rangle$, it is not too hard to show that $\delta_H(p_ip_j, P) \leq \delta_F(p_ip_j, P)$ under any L_p-metrics, which implies that $\kappa_H(\varepsilon, P) \leq \kappa_F(\varepsilon, P)$. The inverse however does not hold, and there are polygonal curves P and values of ε for which $\kappa_H(\varepsilon, P) = O(1)$ and $\kappa_F(\varepsilon, P) = \Omega(n)$.

The Fréchet error measure takes the order along the curve into account, and hence is more useful in some cases especially when the order of the curve is important (such as curves derived from protein backbones). Figure 12 illustrates a substructure of a protein backbone, where the ε-

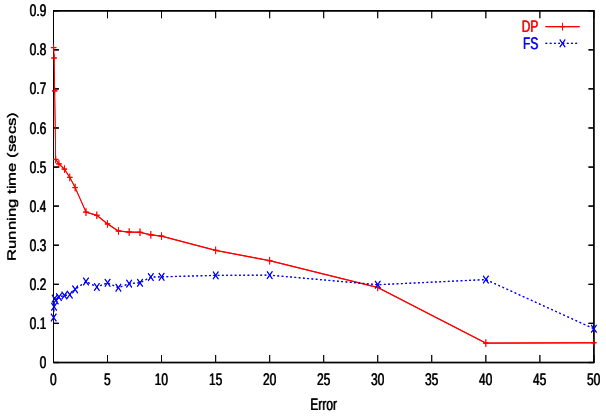
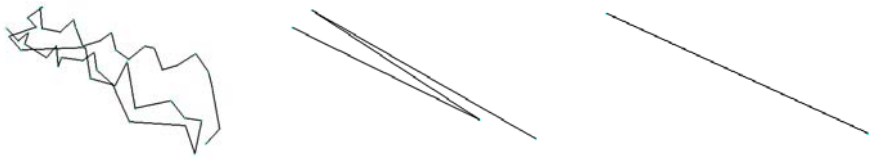


Figure 11. Comparison of running time of FS and DP algorithms for varying ε on Protein B.



(a) (b) (c)

Figure 12. (a) Polygonal chain (a piece of protein backbone) composed of three alpha-helices, (b) its Fréchet ε-simplification and (c) its Hausdorff ε-simplification using DP algorithm.

simplification under Fréchet error measure preserves the overall structure, while the ε-simplification under Hausdorff error measure is unable to preserve it. Note that the Douglas-Peucker algorithm is also based on Hausdorff error measure. Therefore the above discussion holds for it as well.

Weak-Fréchet vs. Fréchet. In Section 3 we described a fast approximation algorithm for computing an ε-simplification of P under the Fréchet error measure, where we used the Fréchet measure in a local manner: we restrict the curve $\langle p_i, \ldots, p_j \rangle$ to match to the line segment $p_i p_j$. We can remove this restriction to make the measure more global by considering weak ε-simplification. More precisely, given P and $S = \langle s_1, s_2, \ldots, s_t \rangle$, where s_i does not necessarily lie on P, S is a *weak ε-simplification* under Fréchet error measure if $\mathcal{F}(P, S) \leq \varepsilon$. The following lemma shows that for Fréchet error measure, the size of the optimal ε-simplification can be bounded in terms of the size of the optimal weak ε-simplification:

Theorem 4.1 *Given a polygonal curve P,*

$$\kappa_F(\varepsilon) \leq \hat{\kappa}_F(\varepsilon/4) \leq \kappa_F(\varepsilon/4).$$

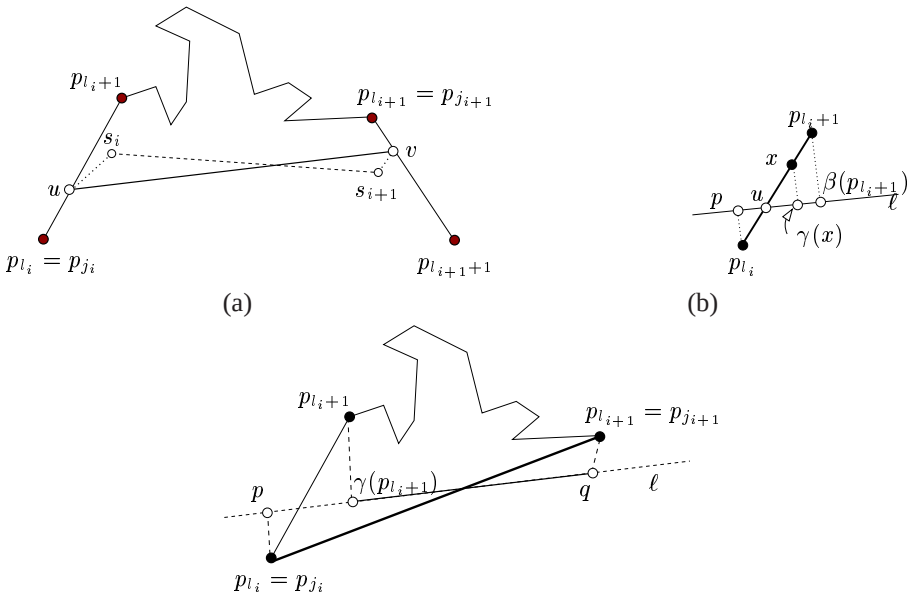


Figure 13. Relationship between $\hat{\kappa}_F(\varepsilon, P)$ and $\kappa_F(\varepsilon, P)$: (a) $u = \alpha(s_i)$ and $v = \alpha(s_{i+1})$; (b) depicts the map γ, $\beta(p_{l_i+1}) = \gamma(p_{l_i+1})$, and $p = \gamma(p_{l_i})$. (c) the case for $j_i = l_i$ and $j_{i+1} = l_{i+1}$: $q = \beta(p_{l_{i+1}}) = \eta(p_{l_{i+1}})$.

PROOF. The second inequality is immediate, so we prove the first one. Let $S = \langle s_1, \ldots, s_t \rangle$ be an optimal weak $(\varepsilon/4)$-simplification of P, and let $\alpha : S \to P$ be a Fréchet map so that $d(x, \alpha(x)) \leq \varepsilon/4$ for all $x \in S$ (see Section 1.1 for the definition). For $1 \leq i \leq t$, let $e_i = p_{l_i} p_{l_i+1}$ be the edge of P that contains $\alpha(s_i)$, and let p_{j_i} be the endpoint of e_i that is closer to $\alpha(s_i)$. We set $P' = \langle p_{j_1} \ldots, p_{j_t} \rangle$; we remove a vertex from this sequence if it is the same as its predecessor. Clearly, $j_i < j_{i+1}$, so P' is a simplification of P. Next we show that $\delta_F(p_{j_i} p_{j_{i+1}}, P) \leq \varepsilon$, for all $1 \leq i < t$.

Let $u = \alpha(s_i)$ and $v = \alpha(s_{i+1})$. See Figure 13 (a) for an illustration.

Claim A $\mathcal{F}(uv, P[u, v]) \leq \varepsilon/2$.

PROOF. By construction, $d(s_i, u), d(s_{i+1}, v) \leq \varepsilon/4$. Therefore, $\mathcal{F}(uv, s_i s_{i+1}) \leq \varepsilon/4$ (by Lemma 3.1). On the other hand, since S is a weak $(\varepsilon/4)$-simplification of P, $\mathcal{F}(s_i s_{i+1}, P[u, v]) \leq \varepsilon/4$. The claim then follows from Lemma 3.2. ∎

Let $\beta : P[u, v] \to uv$ be a Fréchet map such that $d(x, \beta(x)) \leq \varepsilon/2$, for all $x \in P[u, v]$. Let ℓ be the line containing uv. We define a map $\gamma : e_i \to \ell$ that maps a point $x \in e_i$ to the intersection point of ℓ with the line through x and parallel to $p_{l_i+1}\beta(p_{l_i+1})$. See Figure 13 (b) for an illustration. If $p_{j_i} = p_{l_i}$, then $d(p_{j_i}, \gamma(p_{j_i})) \leq \varepsilon/2$ since $d(u, p_{j_i}) \leq d(u, p_{l_i+1})$: This is the case depicted in Figure 13. Hence, we can infer that

$$\mathcal{F}(p_{j_i} p_{l_i+1}, \gamma(p_{j_i})\gamma(p_{l_i+1})) \leq \varepsilon/2. \tag{2}$$

Similarly, we define a map $\eta : e_{i+1} \to \ell$ that maps a point $x \in e_{i+1}$ to the intersection point of ℓ with the line through x and parallel to $p_{l_{i+1}}\beta(p_{l_{i+1}})$. As above,

$$\mathcal{F}(p_{l_{i+1}}p_{j_{i+1}}, \eta(p_{l_{i+1}})\eta(p_{j_{i+1}})) \leq \varepsilon/2. \tag{3}$$

Let p (resp. q) denote the point $\gamma(p_{j_i})$ (resp. $\eta(p_{j_{i+1}})$); see Figure 13 (c).

Claim B $\mathcal{F}(pq, p_{j_i}p_{j_{i+1}}) \leq \varepsilon/2$.

PROOF. Since $d(p_{j_i}, p), d(p_{j_{i+1}}, q) \leq \varepsilon/2$, the claim follows from Lemma 3.1. □

Claim C $\mathcal{F}(pq, P[p_{j_i}, p_{j_{i+1}}]) \leq \varepsilon/2$.

PROOF. By definition of Fréchet distance,

$$\begin{aligned}\mathcal{F}(pq, P[p_{j_i}, p_{j_{i+1}}]) \;\leq\; \max\{\; & \mathcal{F}(p\gamma(p_{l_i+1}), p_{j_i}p_{l_i+1}),\\ & \mathcal{F}(\gamma(p_{l_i+1})\eta(p_{l_{i+1}}), P[p_{l_i+1}, p_{l_{i+1}}]),\\ & \mathcal{F}(p_{l_{i+1}}p_{j_{i+1}}, \eta(p_{l_{i+1}})q)\;\}.\end{aligned}$$

By (2) and (3), the first and the third terms are at most $\varepsilon/2$. To bound the second term, observe that $\gamma(p_{l_i+1}) = \beta(p_{l_i+1})$ and $\eta(p_{l_{i+1}}) = \beta(p_{l_{i+1}})$. It then follows from the definition of the map β and Claim A that

$$\mathcal{F}(\beta(p_{l_i+1})\beta(p_{l_{i+1}}), P[p_{l_i+1}, p_{l_{i+1}}]) \leq \mathcal{F}(uv, P[u, v]) \leq \varepsilon/2,$$

which implies that the second term is at most $\varepsilon/2$ as well. Thus proves the claim. □

Claim B and C along with Lemma 3.2 imply that $\delta_F(p_{j_i}p_{j_{i+1}}, P) \leq \varepsilon$, and therefore $\delta_F(P', P) \leq \varepsilon$. This completes the proof of the theorem. □

5 Conclusions

In this paper we presented near-linear approximation algorithms for curve simplification under Hausdorff and Fréchet error measures. We presented the first efficient approximation algorithm for Fréchet simplifications of a curve in dimension higher than two. Our experimental results demonstrate that our algorithms are efficient.

We conclude by mentioning a few open problems.

(i) Does there exist a near-linear algorithm for computing an ε-simplification of size at most $c\kappa_M(\varepsilon, P)$ for a polygonal curve P, where $c \geq 1$ is a constant?

(ii) Is it possible to compute the optimal ε-simplification under Hausdorff error measure in near-linear time, or under Fréchet error measure in sub-cubic time?

(iii) Is there any provably efficient exact/approximation algorithm for curve simplification in $\mathbb{R}^2$ that returns a simple curve if the input curve is simple.

References

[1] http://www.marketdata.nasdaq.com/mr4b.html.

[2] http://www.cs.uic.edu/wolfson/html/mobile.html.

[3] Protein data bank. http://www.rcsb.org/pdb/.

[4] P. Agarwal and K. R. Varadarajan. Efficient algorithms for approximating polygonal chains. *Discrete Comput. Geom.*, 23:273–291, 2000.

[5] H. Alt and M. Godeau. Computing the frechet distance between two polygonal curves. *International Journal of Computational Geometry*, pages 75–91, 1995.

[6] G. Barequet, D. Z. Chen, O. Daescu, M. T. Goodrich, and J. Snoeyink. Efficiently approximating polygonal paths in three and higher dimensions. *Algorithmica*, 33(2):150 – 167, 2002.

[7] W. S. Chan and F. Chin. Approximation of polygonal curves with minimum number of line segments. In *Proc. 3rd Annual International Symposium on Algorithms and Computation*, pages 378–387, 1992.

[8] D. H. Douglas and T. K. Peucker. Algorithms for the reduction of the number of points required to represent a digitized line or its caricature. *Canadian Cartographer*, 10(2):112–122, 1973.

[9] M. Godau. A natural metric for curves: Computing the distance for polygonal chains and approximation algorithms. In *Proc. of the 8th Annual Symposium on Theoretical Aspects of Computer Science*, pages 127–136, 1991.

[10] L. J. Guibas, J. E. Hershberger, J. B. Mitchell, and J. Snoeyink. Approximating polygons and subdivisions with minimum link paths. *International Journal of Computational Geometry and Applications*, 3(4):383–415, 1993.

[11] P. Heckbert and M. Garland. Survey of polygonal surface simplification algorithms. In *SIGGRAPH 97 Course Notes: Multiresolution Surface Modeling*, 1997.

[12] J. Hershberger and J. Snoeyink. An $O(n \log n)$ implementation of the Douglas-Peucker algorithm for line simplification. In *Proc. 10th Annual ACM Symposium on Computational Geometry*, pages 383–384, 1994.

[13] H. Imai and M. Iri. An optimal algorithm for approximating a piecewise linear function. *Information Processing Letters*, 9(3):159–162, 1986.

[14] H. Imai and M. Iri. Polygonal approximations of a curve-formulations and algorithms. In G. T. Toussaint, editor, *Computational Morphology*, pages 71–86. North-Holland, Amsterdam, Netherlands, 1988.

[15] A. Melkman and J. O'Rourke. On polygonal chain approximation. In G. T. Toussaint, editor, *Computational Morphology*, pages 87–95. North-Holland, Amsterdam, Netherlands, 1988.

[16] R. Weibel. Generalization of spatial data: principles and selected algorithms. In M. van Kreveld, J. Nievergelt, T. Roos, and P. Widmayer, editors, *Algorithmic Foundations of Geographic Information System*. Springer-Verlag Berlin Heidelberg New York, 1997.

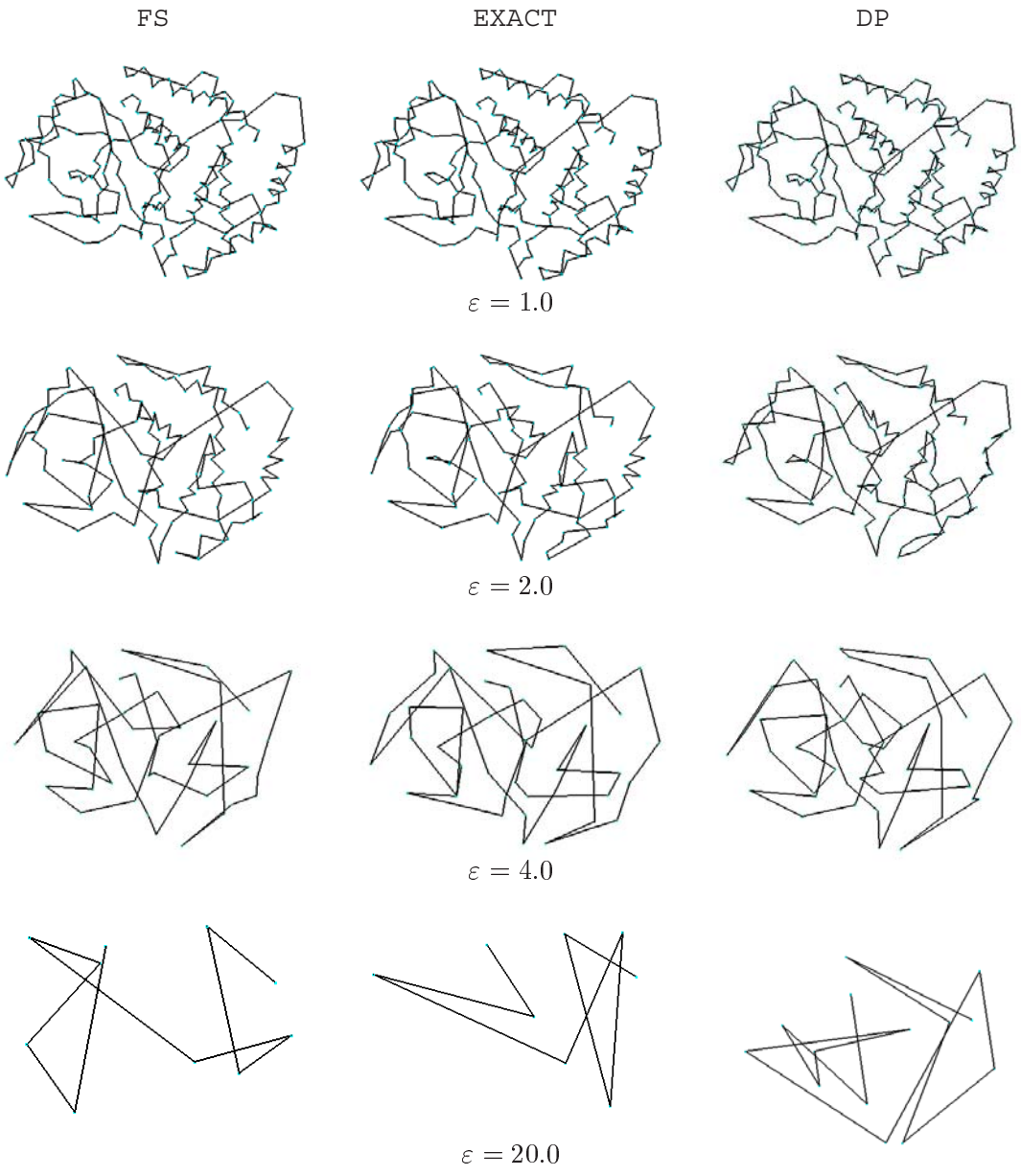


Figure 14. Simplifications of a protein (Protein A) backbone.