Near-Linear Time Approximation Algorithms for Curve Simplification *

Pankaj K. Agarwal[1], Sariel Har-Peled[2], Nabil H. Mustafa[1], and Yusu Wang[1]

[1] Department of Computer Science,
Duke University, Durham, NC 27708-0129, USA.
{pankaj, nabil, wys}@cs.duke.edu
[2] Department of Computer Science, DCL 2111; University of Illinois;
1304 West Springfield Ave.; Urbana, IL 61801; USA;
sariel@cs.uiuc.edu

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 fundamentally different error measures — Hausdorff and Fréchet error measures. 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 $\mathbb{R}^2$ and $\mathbb{R}^3$ for Hausdorff error measure and arbitrary curves in $\mathbb{R}^d$ for the Fréchet error measure. 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 is to compute another polygonal curve that approximates the original curve, according to some predefined error criterion, and whose complexity is as small as possible. Curve simplification has useful applications in various fields, including geographic information systems (GIS), computer vision, graphics, image processing, and data compression. The massive amounts of data available from various sources make efficient processing of this data a challenging task. One of the major applications of this data is for cartographic purposes, where the information has to be visualized and presented as a simple and easily readable map. Since the information is too dense, the maps are usually simplified. To this end, curve simplification

* Research by the first, the third and the fourth authors is supported by NSF grants ITR-333-1050, EIA-98-70724, EIA-01-31905, CCR-97-32787, and CCR-00-86013. Research by second author is partially supported by a NSF CAREER award CCR-0132901.

is used to simplify the representation of rivers, roads, coastlines, and other features when a map at large scale is produced. There are many advantages of the simplification process, such as removing unnecessary cluttering due to excessive detail, saving disk and memory space, and reducing the rendering time.

1.1 Problem definition

Let $P = \langle p_1, \ldots, p_n \rangle$ denote a polygonal curve in $\mathbb{R}^2$ or $\mathbb{R}^3$, where n is the size of P. A polygonal curve $P' = \langle p_{i_1}, \ldots, p_{i_k} \rangle \subseteq P$ *simplifies* P if $1 = i_1 < \cdots < i_k = n$. A curve P is x-monotone if the x-coordinates of p_i are increasing. A curve is *monotone* if there exists a coordinate system for which it is x-monotone. In the rest of the paper, we assume monotonicity to mean x-monotonicity.

Let $d(\cdot, \cdot)$ denote a distance function between points. In this paper we use L_1, L_2, L_∞, and uniform metrics to measure the distance between two points. Uniform metric is defined in $\mathbb{R}^2$ as follows: For two points $a = (a_x, a_y), b = (b_x, b_y)$ in $\mathbb{R}^2$, $d(a, b)$ is $|a_y - b_y|$ if $a_x = b_x$ and ∞ otherwise. The distance between a point p and a segment e is defined as $d(p, e) = \min_{q \in e} d(p, q)$.

Let $\delta_M(p_i p_j, P)$ denote the error of a segment $p_i p_j$ w.r.t. P under error measure M. M can be either Hausdorff (H) or Fréchet (F) error measure and will be defined formally in Section 1.3. The error of a simplification $P' = \langle p_{i_1}, \ldots, p_{i_k} \rangle$ of P is defined as

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

Call P' an *ε-simplification* of P if $\delta_M(P', P) \le \varepsilon$. The *curve-simplification problem* is to compute the smallest size ε-simplification of P, with its size denoted as $\kappa_M(\varepsilon, P)$. We use $\delta_M(\varepsilon)$ and $\kappa_M(\varepsilon)$ to denote $\delta_M(\varepsilon, P)$ and $\kappa_M(\varepsilon, P)$ respectively when P is clear from the context.

If we remove the constraint that the vertices of P' are a subset of the vertices of P, then P' is called a *weak ε-simplification* of P.

1.2 Previous results

The problem of approximating a polygonal curve has been studied extensively during the last two decades, both for computing an ε-simplification and a weak ε-simplification (see [Wei97] for a survey). Imai and Iri [II88] formulated this simplification problem as computing a shortest path between two nodes in a directed acyclic graph. In $\mathbb{R}^2$, under the Hausdorff measure with uniform metric, their algorithm runs in $O(n^2 \log n)$ time. Chin and Chan [CC92], and Melkman and O'Rourke [MO88] improve the running time of their algorithm to quadratic or near quadratic. Agarwal and Varadarajan [AV00] improve the running time to $O(n^{4/3+\delta})$ for L_1 and uniform metric, for $\delta > 0$, by implicitly representing the underlying graph. In dimensions higher than two, Barequet *et al.* compute the optimal ε-simplification under Hausdorff error measure in quadratic time for L_1 and $L_{\inf}$ metrics [?]. For L_2 metric, optimal simplification can be computed in near quadratic time in $\mathbb{R}^3$, and in $O(n^{3-2/(\lfloor \frac{d}{2} \rfloor + 1)} \text{polylog} n)$ in $\mathbb{R}^d$, for $d \ge 4$.

Curve simplification using the Fréchet error measure is first proposed by Godau [God91], who shows that $\kappa_F(\varepsilon)$ is smaller than the size of the optimal weak $\varepsilon/7$-simplification. Alt and Godau [AG95] also propose the first algorithm to compute Fréchet distance between two polygonal curves in $\mathbb{R}^d$ in time $O(mn)$, where m and n are the complexity of the two curves.

The problem of developing a near-linear time algorithm for computing an optimal ε-simplification remains elusive, and several heuristics have been proposed over the years. The most widely used heuristic is the Douglas-Peucker method [DP73] (together with its variants), originally proposed for simplifying curves under the Hausdorff error measure. Its worst case running time is $O(n^2)$ in $\mathbb{R}^d$. In $\mathbb{R}^2$, the running time is improved to $O(n \log n)$ by Snoeyink *et al.* [HS94]. However, the Douglas-Peucker heuristic does not offer any guarantee on the size of the simplified curve.

The second class of simplification algorithms compute a weak ε-simplification of the polygonal curve P. Imai and Iri [II86] give an optimal $O(n)$ time algorithm for finding an optimal weak ε-simplification of a given monotone curve under Hausdorff error measure in $\mathbb{R}^2$. For weak ε-simplification of curves in $\mathbb{R}^2$ under Fréchet distance, Guibas *et al.* [GHMS93] proposed a factor 2 approximation algorithm with $O(n \log n)$ running time, and an $O(n^2)$ exact algorithm using dynamic programming. They also approximate some other variants of weak simplifications for polygonal curves in $\mathbb{R}^2$ in near linear time.

The problem of simplifying curves becomes much harder when additional constraints such as topology preservation and non-intersection are introduced. Given a set of non-intersecting curves, the problem of simplifying the curves optimally so that the simplified curves are also non-intersecting is NP-hard – in fact, it is hard to approximate within a factor of $n^{1/5-\delta}$, for any $\delta > 0$, where n is the total number of vertices of the curves [EM01]. Guibas *et al.* [GHMS93] show that the problem of computing an optimal non-intersecting simplification of a simple polygon is NP-hard, and that computing the optimal weak simplification of a set of non-intersecting curves is also NP-hard.

1.3 Our results

In this paper, we study the curve-simplification problem under both the Fréchet and Hausdorff error measures. We present simple near-linear time algorithms for computing an ε-simplification of size at most $\kappa(\varepsilon/c)$, where $c \geq 1$ is a constant. In particular, our contributions are:

Hausdorff Error Measure. Define the *Hausdorff error* of a line segment p_ip_j w.r.t. P, where $p_i, p_j \in P$, $i < j$, to be

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

We prove the following theorem in Section 2.

Theorem 1. *Given a monotone polygonal curve P and a parameter $\varepsilon > 0$, one can compute an ε-simplification with size at most $\kappa_H(\varepsilon/2, P)$ in:*

(i) $O(n)$ *time and space, under the* L_1, L_2, L_∞ *or uniform metrics in* $\mathbb{R}^2$*;*
(ii) $O(n \log n)$ *time and* $O(n)$ *space, under* L_1 *or* L_∞ *metrics in* $\mathbb{R}^3$*.*

We have implemented the algorithm in $\mathbb{R}^2$ and present experimental results.

Fréchet Error Measure. Given two curves $f : [a, a'] \to \mathbb{R}^d$, and $g : [b, b'] \to \mathbb{R}^d$, the Fréchet distance $\mathcal{F}_{\mathrm{D}}(f, g)$ between them is defined as:,

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

where α and β range over continuous and increasing functions with $\alpha(0) = a, \alpha(1) = a', \beta(0) = b$ and $\beta(1) = b'$. The *Fréchet error* of a line segment $p_i p_j$ where $p_i, p_j \in P$, $i < j$, is defined to be

$$\delta_F(p_i p_j, P) = \mathcal{F}_{\mathrm{D}}(p_i p_j, \pi(p_i, p_j)),$$

where $\pi(p, q)$ denotes the subcurve of P from p to q. We prove the following result in Section 3:

Theorem 2. *Given a polygonal curve* P *in* $\mathbb{R}^d$ *and a parameter* $\varepsilon \geq 0$*, an* ε*-simplification of* P *with size at most* $\kappa_F(\varepsilon/2, P)$ *can be constructed in* $O(n \log n)$ *time and* $O(n)$ *space.*

The algorithm is independent of any monotonicity properties. To our knowledge, it is the first efficient, simple approximation algorithm for curve simplification in dimension higher than two under the Fréchet error measure. We provide experimental results for polygonal chains in $\mathbb{R}^3$ to demonstrate the efficiency and quality of our approximation algorithm.

Relations between simplifications. We further analyze the relations between simplification under Hausdorff and Fréchet error measures, and Fréchet and weak Fréchet ε-simplification in Section 4.

2 Hausdorff simplification

Let $P = \langle p_1, \ldots, p_n \rangle$ be a monotone polygonal curve in $\mathbb{R}^2$ or $\mathbb{R}^3$. For a given distance function $d(\cdot, \cdot)$, let $D(p, r) = \{q \mid d(p, q) \leq r\}$ be the disk of radius r centered at p. Let D_i denote $D(p_i, \varepsilon)$. Then $p_i p_j$ is a valid segment, i.e. $\delta_H(p_i p_j) \leq \varepsilon$, if and only if $p_i p_j$ intersects $D_{i+1}, \ldots, D_j$ in order. We now define a general problem, and use it to compute an ε-simplification of polygonal curve under Hausdorff error measure with different distance metrics.

2.1 Segment cover

Let $E = \langle e_1, e_2, \ldots, e_m \rangle$ be a sequence of segments. E is called an *ε-segment cover* of P if there exists a subsequence of vertices $p_{i_1} = p_1, p_{i_2}, \ldots, p_{i_{m+1}} = p_n$, $i_1 < i_2 < \ldots < i_{m+1}$, such that e_j intersects D_{i_j}, D_{i_j+1}, ..., $D_{i_{j+1}}$ in order, and the endpoints of e_j lie in D_{i_j} and $D_{i_{j+1}}$ respectively. An ε-segment cover is optimal if its size is minimum among all ε-segment covers. The following lemma is straightforward, as an ε-simplification of a monotone curve P is an ε-segment cover of P as well.

Lemma 1. *Let $\mu_\varepsilon(P)$ denote the size of an optimal ε-segment cover for a monotone curve P. Then, $\mu_\varepsilon(P) \leq \kappa_H(\varepsilon, P)$.*

Lemma 2. *Let $E = \langle e_1, e_2, \ldots, e_m \rangle$ be an $\varepsilon/2$-segment cover of size m of a monotone polygonal curve P. Then an ε-simplification of size at most m can be computed in $O(m)$ time.*

Proof. By the definition of an $\varepsilon/2$-segment cover, there exists a subsequence of vertices $p_{i_1} = p_1, p_{i_2}, \ldots, p_{i_{m+1}} = p_n$ such that e_j intersects $D(p_{i_j}, \varepsilon/2)$, $D(p_{i_j+1}, \varepsilon/2)$, ..., $D(p_{i_{j+1}}, \varepsilon/2)$, and the endpoints of e_j lie in $D(p_{i_j}, \varepsilon/2)$ and $D(p_{i_{j+1}}, \varepsilon/2)$. See Figure 1 for an example of an optimal ε-segment cover under uniform metric. Define the polygonal curve $P' = \langle p_{i_1} = p_1, \ldots, p_{i_m}, p_{i_{m+1}} = p_n \rangle$. Using the triangle inequality one can easily verify that the segment $p_{i_j} p_{i_{j+1}}$ intersects all the disks $D(p_{i_j}, \varepsilon), D(p_{i_j+1}, \varepsilon), \ldots, D(p_{i_{j+1}}, \varepsilon)$ in order. Hence $p_{i_j} p_{i_{j+1}}$ is a valid segment. Therefore, the polygonal curve P' is an ε-simplification of P, and it can be constructed in $O(m)$ time. □

2.2 An approximation algorithm

In this section, we present near-linear approximation algorithms for computing an ε-simplification of a monotone polygonal curve P in $\mathbb{R}^2$ and $\mathbb{R}^3$ under the Hausdorff error measure.

Algorithm. The approximation algorithm to compute an ε-simplification for a monotone curve P, denoted `HausdorffSimp`, in fact computes an optimal $\varepsilon/2$-segment cover of P. It then follows from Lemma 2 that the vertex set of an optimal $\varepsilon/2$-segment cover forms an ε-simplification P' of P. The size of P' is at most $\kappa_H(\varepsilon/2, P)$ by Lemma 1.

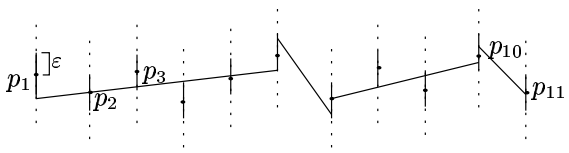


Fig. 1. Covering the vertical segments of length 2ε with maximal set of stabbing lines.

An ordered set of disks $\mathcal{D}$ has a line transversal if there exists a line intersecting all the disks $D_i \in \mathcal{D}$ in order. We use the greedy method of Guibas *et al.* [GHMS93] to compute an optimal ε-segment cover: start with the set $\mathcal{D} = \langle D_1 \rangle$. Now iteratively add each disk D_k, $k = 2, 3, \ldots$ to $\mathcal{D}$. If there does not exist a line transversal for $\mathcal{D}$ after adding disk D_k, then add the vertex $p_{i_1} = p_k$ to our segment-cover, set $\mathcal{D} = \langle D_k \rangle$, and continue. Let $S = \langle p_{i_1}, \ldots, p_{i_m} \rangle$ be the polygonal curve computed by algorithm `HausdorffSimp`. Clearly, the segments $\mathcal{C}(S) = \langle e_j = p_{i_j} p_{i_j+1}, j = 1, \ldots, (m-1) \rangle$ form a $\varepsilon/2$-segment cover. It follows from the convexity property of line transversals (i.e., if there exists a line transversal for disks $D_i \cdots D_j$, then there also must exist a line transversal for disks $D_{i'} \cdots D_{j'}$ for all $i \leq i' < j' \leq j$) that the resulting set $\mathcal{C}(S)$ computed is an optimal $\varepsilon/2$-cover.

Analysis. Given a set of i disks $\mathcal{D} = \langle D_1, \ldots, D_i \rangle$, it takes linear time to compute a line that stabs $\mathcal{D}$ in order in $\mathbb{R}^2$ under L_1, L_2, $L\infty$, and uniform metrics [GHMS93]. The algorithm is incremental, and we can use a data structure so that it only takes constant time to update the data structure while adding a new disk. Thus our greedy approach uses $O(n)$ time and space overall in $\mathbb{R}^2$. In $\mathbb{R}^3$, the line transversal under L_1 and L_∞ metrics can be computed in $O(i)$ time using linear programming [Ame92], and one needs to update it efficiently when a new disk is added. (Of course, we can use techniques for dynamic linear programming [Ram00], but we describe a faster and simpler approach.) We use an exponential-binary search method using the linear programming algorithm as a black-box, and obtain an $O(n \log n)$ running time. The exponential-binary search method will be described in more detail in the next section. To compute a line that stabs a set of balls under L_2 norm in $\mathbb{R}^3$ seems to be hard, with no near-linear time algorithm currently known — the best algorithm computes a transversal of n disks in time $O(n^{3+\varepsilon})$ [AAS99], yielding a $O(n^{3+\varepsilon} \log n)$ time algorithm for computing a simplification of size $\kappa_H(\varepsilon/2, P)$. Nevertheless, a simplification under L_1 or L_∞ norm is an approximation for the simplification under L_2 norm. Putting everything together proves Theorem 1.

Remark. For monotone curves, the size of the Hausdorff simplification is in fact equal to the size of the Fréchet simplification. In the next section, we extend this approach (with an extra logarithmic overhead) to work for simplifying arbitrary curves under the Fréchet error measure.

Remark 2. Segment cover is not well-defined for non-monotone curves, as the stabbing order is not well-defined. One variant of the simplification problem that has been investigated for non-monotone curves [GHMS93] is that the simplifying curve stabs each of the disks of the vertices *in order* in the following sense: For a stabbing directed line l and a set of disks $D_k, D_{k+1}, \ldots, D_m$, there exist a set of points $q_k, q_{k+1}, \ldots, q_m \in l$, such that $q_i \in D_i$, and q_i appears before q_j on l, for $k \leq i < j \leq m$. We can modify the definition of segment cover based on this new order, and Lemma 1 and Lemma 2 hold for the new concept. A result by Guibas et al. [GHMS93] states that such an optimal segment cover can be computed

Points along a Sine curve		
$\varepsilon = 0.6$	*Approx.*	*DP*
100	28	29
200	48	58
500	148	120
1000	284	247
2000	552	459
5000	1336	1073
10000	2700	2173
20000	5360	4391
40000	10720	9237
100000	26800	22995

(a)

	Composite Index		*Industry Index*	
Size:	*7,289*		*7,289*	
ε	*Approx.*	*DP*	*Approx.*	*DP*
0.05	6989	6991	6912	6920
0.10	6721	6718	6572	6570
0.50	5148	5119	4748	4692
1.00	4048	4011	3633	3582
3.00	2319	2292	1964	1946
5.00	1669	1667	1373	1365
10.00	996	994	767	772
20.00	544	547	384	387

(b)

Fig. 2. (a) ε-simplification sizes on varying sized synthetic input, (b) ε-simplification size on two stock index curves.

efficiently in $O(n \log n)$ time in $\mathbb{R}^2$ (this definition of simplification corresponds to *Definition 4* in their paper). Unfortunately, it is not clear yet how to compute it efficiently in $\mathbb{R}^3$.

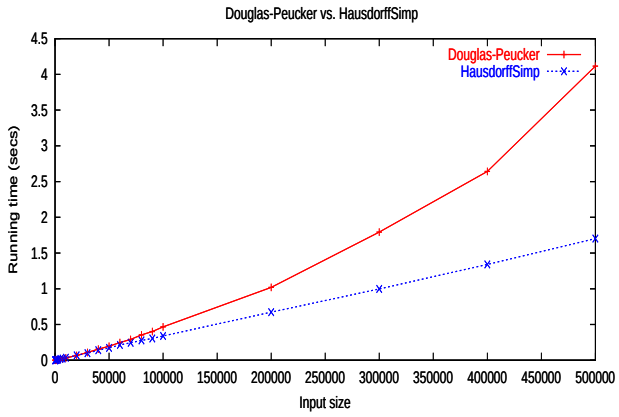


Fig. 3. Comparing running times of `HausdorffSimp` and Douglas-Peucker for $\varepsilon = 0.6$.

2.3 Experiments

In this section, we compare the results of our approximation algorithm for curve simplification under the Hausdorff error measure with the Douglas-Peucker heuristic. For our experiments, there are two input parameters — the size of the

input polygonal curve, and the error threshold ε for simplification. Similarly, there are two output parameters — the size of the simplified curve for a particular ε, and the running time for the simplification algorithm. All our experiments were run on a Sun Blade-100 machine running SunOS 5.8 with 256MB RAM.

Quality comparison. We implemented algorithm `HausdorffSimp` for planar x-monotone curves under the uniform metric. Figure 2 shows the sizes of ε-simplifications produced on (a) synthetic input where $\varepsilon = 0.6$, and the curves consist of points sampled from a sinusoidal curve, and (b) data derived from NASDAQ stock indices, where curve *Composite* (*Industry*) is obtained by plotting the daily NASDAQ composite (industry) index over the years January 1971 till January 2003.

Time comparison. We compare `HausdorffSimp` with Douglas-Peucker on inputs that are most favorable for Douglas-Peucker, where the curve is always partitioned at the middle vertex, and then recursively simplified. Figure 3 compares the running time of the two algorithms, where the curve consists of point sets of varying sizes sampled from a sinusoidal curve. As expected, `HausdorffSimp` exhibits empirically linear running time, outperforming the Douglas-Peucker heuristic.

3 Fréchet simplification

We now present algorithms for simplification under the Fréchet error measure. The exact algorithm for computing $\delta_F(\varepsilon)$ follows the approach of Imai and Iri [II88], used for simplifying a curve under the Hausdorff error measure. For a polygonal curve P and an error ε, we construct a graph $G = (V, E)$. The node set V is the same as the vertex set of P. An edge $(v_i, v_j) \in E$ if and only if $\delta_F(p_ip_j) = \mathcal{F}_{\mathrm{D}}(\pi(p_i, p_j), p_ip_j) \leq \varepsilon$. It is easy to see that the shortest path from p_1 to p_n in G provides the optimal simplification under error bound ε.

The decision version of the Fréchet distance between two polygonal curves A and B can be decided in time $O(|A||B|)$ [AG95]. Thus for each pair of nodes p_i and p_j in graph G, it takes at most $O(n)$ time to decide if $\delta_F(\pi(p_i, p_j), p_ip_j) \leq \varepsilon$. There are $O(n^2)$ pairs to be checked to construct G explicitly. Finding the shortest path for a pair of vertices in G takes $O(|V| + |E|)$ time. Therefore the overall time complexity of the algorithm is $O(n^3)$. We focus on approximation algorithms below.

Lemma 3. *Given two directed segments uv and xy in $\mathbb{R}^d$,*

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

where $d(\cdot, \cdot)$ represents the L_1, L_2 or L_∞ norm.

Proof. Let δ denote the maximum of $d(u, x)$ and $d(v, y)$. Note that $\mathcal{F}_{\mathrm{D}}(uv, xy) \geq \delta$, since u (resp. v) has to be matched to x (resp. y). Assume the natural parameterization for segment uv, $A(t) : [0, 1] \to uv$, such that $A(t) = u + t(v - u)$.

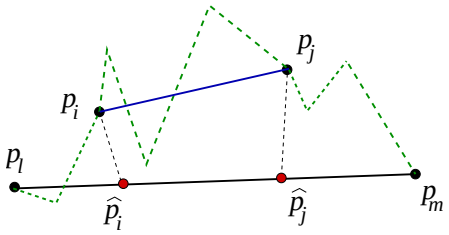


Fig. 4. Bold dashed curve is $\pi(p_l, p_m)$, p_i and p_j are matched to $\hat{p}_i$ and $\hat{p}_j$ respectively.

Similarly, define $B(t) : [0,1] \to xy$ 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 $\mathcal{F}_{\mathrm{D}}(uv, xy) \leq \delta$. □

Lemma 4. *Given a polygonal curve P and two directed segments uv and xy,*

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

Proof. The optimal matching between polygonal curve P and segment uv is composed of two parameterizations of P and uv, $P(t)$ and $A(t)$, $t \in [0,1]$, respectively. Assume that $A(t) : [0,1] \to uv$ is the natural parameterization of uv, such that $A(t) = u + t(v - u)$. As in the proof of Lemma 3, let $B(t) : [0,1] \to xy$ be such that $B(t) = x + t(y - x)$. Note that by triangle inequality, for any $t \in [0,1]$,

$$\begin{aligned} d(P(t), A(t)) - d(A(t), B(t)) &\leq d(P(t), B(t)) \\ &\leq d(P(t), A(t)) + d(A(t), B(t)), \end{aligned}$$

$$\Rightarrow \mathcal{F}_{\mathrm{D}}(uv, P) - \mathcal{F}_{\mathrm{D}}(uv, xy) \leq \mathcal{F}_{\mathrm{D}}(P, xy) \leq \mathcal{F}_{\mathrm{D}}(P, uv) + \mathcal{F}_{\mathrm{D}}(uv, xy).$$

□

Lemma 5. *Let $P = \langle p_1, p_2, \ldots, p_n \rangle$ be a polygonal curve. For $l \leq i \leq j \leq m$, $\delta_F(p_i p_j, P) \leq 2 \cdot \delta_F(p_l p_m, P)$.*

Proof. Let $\delta^* = \delta_F(p_l p_m)$. Suppose under the optimal matching between $\pi(p_l, p_m)$ and $p_l p_m$, p_i and p_j are matched to $\hat{p}_i$ and $\hat{p}_j \in p_l p_m$ respectively (see Figure 4 for an illustration). Then by definition, $\mathcal{F}_{\mathrm{D}}(\hat{p}_i \hat{p}_j, \pi(p_i, p_j)) \leq \delta^*$. In particular, we have that $d(p_i, \hat{p}_i) \leq \delta^*$, and $d(p_j, \hat{p}_j) \leq \delta^*$. By Lemma 3, $\mathcal{F}_{\mathrm{D}}(p_i p_j, \hat{p}_i \hat{p}_j) \leq \delta^*$. It then follows from Lemma 4 that

$$\delta_F(p_i p_j) = \mathcal{F}_{\mathrm{D}}(\pi(p_i, p_j), p_i p_j) \leq \mathcal{F}_{\mathrm{D}}(\pi(p_i, p_j), \hat{p}_i \hat{p}_j) + \delta^* \leq 2\delta^*.$$

□

3.1 An $O(n \log n)$ Algorithm

The algorithm (denoted `FrechetSimp`) computes an ε-simplification P' of P in a greedy manner: set the initial simplification as $P' = \langle p_{i_1} = p_1 \rangle$, and iteratively add vertices to P' as follows. Assume $P' = \langle p_{i_1}, \ldots, p_{i_j} \rangle$. The algorithm finds an index $k > i_j$ such that (i) $\delta_F(p_{i_j}, p_k) \leq \varepsilon$ and (ii) $\delta_F(p_{i_j}, p_{k+1}) > \varepsilon$. Set $i_{j+1} = k$, output $p_{i_{j+1}}$ to P', and repeat the above procedure till the last vertex of P.

Lemma 6. `FrechetSimp` *computes* P' *such that* $\delta_F(P', P) \leq \varepsilon$, *and* $|P'| \leq \kappa_F(\varepsilon/2)$.

PROOF. It is clear that the algorithm computes a curve that is an ε-simplification of P. It remains to show that the size of P' is bounded by $\kappa_F(\varepsilon/2)$.

Let $Q = \langle p_{j_1} = p_1, \ldots, p_{j_l} = p_n \rangle$ be the optimal $\varepsilon/2$-simplification of P, where $1 \leq j_m \leq n$ for $1 \leq m \leq l$. Let $P' = \langle p_{i_1} = p_1, \ldots, p_{i_k} = p_n \rangle$, where $1 \leq i_m \leq n$ for $1 \leq m \leq k$.

The proof proceeds by induction. The following invariant will always be true: $i_m \geq j_m$, for all m. This implies that $k \leq l$, and therefore $k \leq \kappa_F(\varepsilon/2)$.

Assume $i_{m-1} \geq j_{m-1}$. Let $i' = i_m + 1$. By construction $\delta_F(p_{i_{m-1}}, p_{i'}) > \varepsilon$, and $\delta_F(p_{i_{m-1}}, p_{i'-1}) \leq \varepsilon$. By the inductive step, $i' > j_{m-1}$. If $i' > j_m$, we are done. So assume $i' \leq j_m$. Since Q is an $\varepsilon/2$-simplification, $\delta_F(p_{j_{m-1}} p_{j_m}) \leq \varepsilon/2$. Lemma 5 implies that for all $j_{m-1} \leq i_{m-1} \leq j' \leq j_m$, $\delta_F(p_{i_{m-1}} p_{j'}) \leq \varepsilon$. But since $\delta_F(p_{i_{m-1}}, p_{i'}) > \varepsilon$, $i' > j_m$ and hence $i_m \geq j_m$.

□

After computing vertex $p_{i_m} \in P'$, we find the next vertex $p_{i_{m+1}}$ as follows: let b_ρ be a bit that is one if $\delta_F(p_{i_m} p_{i_m+\rho}) > \varepsilon$ and zero otherwise. b_ρ can be computed in $O(\rho)$ time by the algorithm proposed in [AG95]. Recall our goal: finding two consecutive bits b_Δ and $b_{\Delta+1}$ such that $b_\Delta = 0$ and $b_{\Delta+1} = 1$. Clearly, the index of the next vertex is then $i_{m+1} = i + \Delta$. Δ can be computed by performing an exponential search, followed by a binary search. First find the smallest j such that $b_{2^{j-1}} = 0$ and $b_{2^j} = 1$ by computing the bits $b_{2^{j'}}$, $j' \leq j$. Computing each bit takes time $O(i_{m+1} - i_m)$, since $i_{m+1} \geq 2^{j-1}$. It then follows by simple calculation that the total time is $O(i_{m+1} - i_m)$. Next, use binary search to find any two consecutive bits which are 0-1 combination, in the range $b_{2^{j-1}}, \ldots, b_{2^j}$. Note that this is not strictly a binary search, as the bits which are ones are not necessarily consecutive. Nevertheless, it is straightfoward to verify that the same divide and conquer approach works. This requires computing $O(j-1) = O(\log(i_{m+1} - i_m))$ bits in this range, and it takes $O(i_{m+1} - i_m)$ time to compute each of them. Therefore computing $p_{i_{m+1}}$ takes $O((i_{m+1} - i_m) \log(i_{m+1} - i_m))$ time. Summing over all i_j's yields the running time of $O(n \log n)$, proving Theorem 2.

3.2 Experiments

We now present experiments comparing our $O(n \log n)$ algorithm `FrechetSimp` with (i) the optimal $O(n^3)$ time Fréchet simplification algorithm for quality; and

Curves	*Aprx.*	*Exact*	*Aprx.*	*Exact*	*Aprx.*	*Exact*	*Aprx.*	*Exact*	*Aprx.*	*Exact*
Protein ε:	*0.05*		*0.12*		*1.2*		*1.6*		*2.0*	
ProteinA (327)	327	327	327	327	254	249	220	214	134	124
ProteinB (1,998)	201	201	134	134	42	42	36	36	32	32
ProteinC (9,777)	6786	6431	1537	651	178	168	140	132	115	88
Trajectory ε:	*0.05*		*0.12*		*1.2*		*1.6*		*2.0*	
Traj1 (516)	223	166	161	130	53	49	34	32	13	13
Traj2 (493)	161	112	100	85	41	39	30	27	10	8
Traj3 (498)	214	156	148	114	53	48	32	29	14	14
Stock index ε:	*0.05*		*0.5*		*1.2*		*3*		*5*	
Tel-Trans 1 (7,057)	6882	6880	4601	4469	2811	2637	1396	1228	890	732
Tel-Bio 2 (1,559)	1558	1558	1473	1471	1292	1279	974	942	772	720

Table 1. Comparing the size of simplifications produced by `FrechetSimp` with the optimal algorithm.

(*ii*) with the Douglas-Peucker algorithm under Hausdorff error measure (with L_2 metric) to demonstrate its efficiency. Our experiments were run on a Sun Blade-100 machine with 256 RAM.

Data Sets. We test our algorithms on three types of three dimensional datasets. The first set of curves are derived from *protein backbones* by adding suitably small random "noise" vertices along the edges of the backbone. The second set of curves are trajectories of moving vehicles obtained from the *moving objects database* [MOD]. The last set of curves are index curves formed by taking daily NASDAQ index values (in particular, telecommunication index and biotechnology index varying over time in one case, and telecommunication and transportation in the second) spanning the years January 1971 till January 2003. In each of the above cases, simplification helps identify certain motifs, patterns, and trends in the data.

Quality Comparison. The running time of the optimal simplification algorithm under Fréchet distance is independent of the input curve, or the error measure ε — it is always $\Theta(n^3)$. Therefore, it is *orders* of magnitude slower than our approximation algorithm, and hence we omit its empirical running time. Instead we focus on comparing the quality (size) of simplifications produced by `FrechetSimp` and the optimal Fréchet simplification algorithm. The results are presented in Table 1, divided according to the three types of input curves — protein backbones, point trajectories, and stock index changes. As seen from Table 1, the size of the simplifications produced by our approximation algorithm is always close to the optimal sized simplification.

Running time Comparison. We compare the efficiency of `FrechetSimp` with the widely-used Douglas-Peucker heuristic under Hausdorff error measure[1]. Table 2

[1] We can extend the Douglas-Peucker approach to simplify curves under Fréchet error measure. However, the implementation of such an algorithm is inefficient, with an $O(n^3)$ worst-case time complexity.

illustrates the running times of the two algorithms. Note that as ε increases, resulting in small simplified curves, the running time of Douglas-Peucker decreases.

	ProteinD		*ProteinC*		*ProteinE*		*Tel-Trans*		*Tel-Bio*	
Size:	*49,633*		*9,777*		*2,847*		*7,057*		*1,559*	
ε	*Fréchet*	*DP*	*Fréchet*	*DP*	*Fréchet*	*DP*	*Fréchet*	*DP*	*Fréchet*	*DP*
0.01	0.64	5.06	0.11	0.80	0.03	0.19	0.08	0.61	0.02	0.12
0.05	0.75	4.95	0.15	0.78	0.04	0.19	0.09	0.61	0.02	0.12
0.10	0.91	4.58	0.18	0.71	0.05	0.16	0.09	0.60	0.02	0.12
0.50	1.01	2.74	0.17	0.52	0.05	0.15	0.11	0.58	0.02	0.12
1.00	1.05	2.68	0.18	0.50	0.05	0.13	0.12	0.56	0.02	0.11
2.00	1.15	2.40	0.19	0.46	0.05	0.13	0.13	0.53	0.02	0.11
5.00	1.40	1.88	0.22	0.37	0.06	0.10	0.12	0.48	0.02	0.10
10.00	1.49	1.72	0.23	0.34	0.06	0.10	0.14	0.42	0.02	0.10
13.00	1.48	1.61	0.24	0.32	0.06	0.09	0.13	0.41	0.02	0.10
17.00	1.76	1.43	0.25	0.31	0.07	0.08	0.15	0.38	0.03	0.09
20.00	1.77	1.38	0.24	0.28	0.06	0.06	0.15	0.38	0.03	0.13

Table 2. Comparing the running time of simplifications produced by `FrechetSimp` with the Douglas-Peucker Heuristic.

This phenomenon is illustrated in Figure 5, which compares the running time of our approximation algorithm with Douglas-Peucker for a protein backbone (with artificial noise added) with $49,633$ vertices. This phenomenon is due to the fact that Douglas-Peucker tries to find a line segment that simplifies a curve, and recurses into subproblems only if that fails. Thus, as ε decreases, it needs to make more recursive calls. Our approximation algorithm, however, proceeds in a linear fashion from the first vertex to the last vertex, and hence it is more stable towards changes in ε. For an infinite value of ε, the running time of the two algorithms would be the same, i.e. $O(n)$.

4 Comparisons

In this section, we compare the output under two different error measures, and we relate two different Fréchet simplifications.

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 $\delta_H(p_i p_j) \leq \delta_F(p_i p_j)$. The converse however does not hold.

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

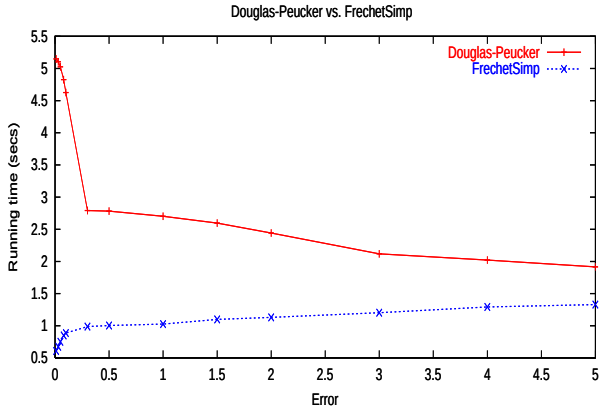
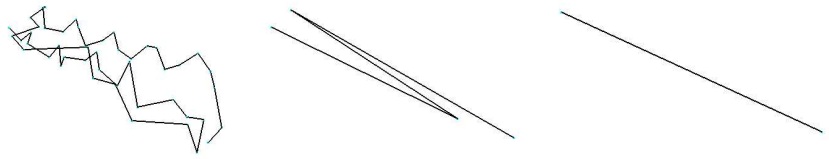


Fig. 5. Comparing running times of `FrechetSimp` and Douglas-Peucker for varying ε for a curve with 49633 vertices.



(a) (b) (c)

Fig. 6. (a) Polygonal chain composed of three alpha-helices, (b) its Fréchet ε-simplification and (c) its Douglas-Peucker Hausdorff ε-simplification.

important (such as curves derived from protein backbones). Figure 6 illustrates a substructure of a protein backbone, where ε-simplifying under Fréchet error measure preserves the overall structure, while ε-simplifying 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 the previous section we described a fast approximation algorithm for computing an ε-simplification of P under 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 instead looking at the *weak* Fréchet ε-simplification. More precisely, given P and $S = \langle s_1, s_2, \ldots, s_m \rangle$, where it is not necessary that $s_i \in P$, S is a weak ε-simplification under Fréchet error measure if $\mathcal{F}_{\mathrm{D}}(P, S) \leq \varepsilon$. The following lemma shows that the size of the optimal

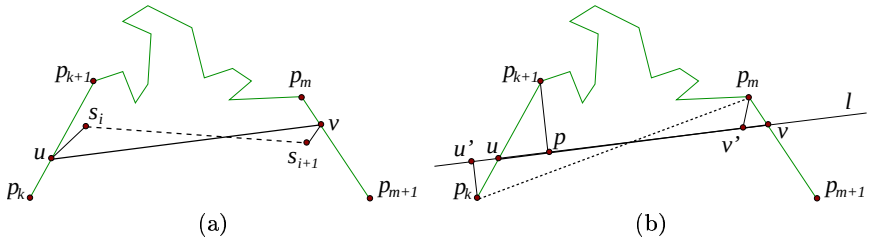


Fig. 7. In (a), u and v are the points that s_i and s_{i+1} are mapped to in the optimal matching between P and S. In (b), $j_i = k$ and $j_{i+1} = m$.

Fréchet simplification can be bounded by the size of the optimal weak Fréchet simplification:

Lemma 7. *Given a polygonal curve P, let $\hat{\kappa}_F(\varepsilon)$ denote the size of the minimum weak ε-simplification of P. Then*

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

Proof. The right side of the inequality is obvious, thus we focus on the left side. Assume $S = \langle s_1, \ldots, s_t \rangle$ is an optimal weak $\varepsilon/4$-simplification of P, i.e., $\mathcal{F}_{\mathrm{D}}(S, P) \leq \varepsilon/4$. For any edge $s_i s_{i+1}$, let $u \in p_k p_{k+1}$ and $v \in p_m p_{m+1}$ denote the points on P that s_i and s_{i+1} are mapped to respectively in the optimal matching between P and S. See Figure 7 (a) for an illustration. Let p_{j_i} (resp. $p_{j_{i+1}}$) denotes the endpoint of $p_k p_{k+1}$ (resp. $p_m p_{m+1}$) that is closer to u (resp. v). In other words, $d(p_{j_i}, u) \leq 1/2 d(p_{k+1}, p_k$, and $d(p_{j_{i+1}}, v) \leq 1/2 d(p_{m+1}, p_m)$. Set $P' = \langle p_{j_1} = p_1, \ldots, p_{j_t} = p_n \rangle$. Clearly, $j_i \leq j_r$ for any $1 \leq i < r \leq t$.

We now show that P' as constructed above is an ε-simplification of P, which will then imply the lemma.

Since $\mathcal{F}_{\mathrm{D}}(\pi(u, v), s_i s_{i+1}) \leq \varepsilon/4$, by Lemma 4, we have $\mathcal{F}_{\mathrm{D}}(\pi(u, v), uv) \leq \varepsilon/2$. Let l denote the line containing segment uv. We construct a segment $u'v' \subset l$ such that $\mathcal{F}_{\mathrm{D}}(\pi(p_{j_i}, p_{j_{i+1}}), u'v') \leq \varepsilon/2$: We describe here how to compute u', and v' can be computed symetrically. Let $p \in uv$ denote the point that p_{k+1} is mapped to in the optimal matching between $\pi(u, v)$ and uv. If $j_i = k+1$, i.e. p_{j_i} is the right endpoint of edge $p_k p_{k+1}$, then set $u' = p$. Otherwise, u' is the point on l such that $p_k u'$ is parallel to $p_{k+1} p$, as illustrated in Figure 7 (b).

By construction, $d(p_{j_i}, u') \leq \varepsilon/2$ (resp. $d(p_{j_{i+1}}, v') \leq \varepsilon/2$), which together with Lemma 3 implies that $\mathcal{F}_{\mathrm{D}}(u'v', p_{j_i} p_{j_{i+1}}) \leq \varepsilon/2$. That is, $\mathcal{F}_{\mathrm{D}}(u'v', p_k p_m) \leq \varepsilon/2$ in Figure 7 (b). On the other hand, we claim that $\mathcal{F}_{\mathrm{D}}(u'v', \pi(p_{j_i}, p_{j_{i+1}})) \leq \varepsilon/2$. Indeed, taking the case depicted in Figure 7 (b) as an example, we can construct a mapping between $u'v'$ and $\pi(p_k, p_m)$ as follows: For any point $x \in p_k u$, map x to $y \in u'u$ such that xy is parallel to $p_k u'$. For points from $\pi(u, p_m)$, we exploit the optimal matching between uv and $\pi(u, v)$ (note that by construction, p_m is mapped to v' in this optimal matching). In the first case, obviously,

$$d(x, y) \leq d(p_k, u') \leq d(p_{k+1}, p) \leq \varepsilon/2.$$

In the second case, as $\mathcal{F}_{\mathrm{D}}(uv, \pi(u,v)) \leq \varepsilon/2$, we have $\mathcal{F}_{\mathrm{D}}(uv', \pi(u,p_m)) \leq \varepsilon/2$. Therefore, $\mathcal{F}_{\mathrm{D}}(u'v', \pi(p_k,p_m)) \leq \varepsilon/2$. Similar arguments prove the claim for other choices of p_{j_i}, $p_{j_{i+1}}$, u', and v'.

It then follows from Lemma 4 that $\mathcal{F}_{\mathrm{D}}(p_{j_i}p_{j_{i+1}}, \pi(p_{j_i},p_{j_{i+1}})) \leq \varepsilon$ for $i = 1 \ldots t$, implying that $\delta_F(P',P) \leq \varepsilon$. □

5 Conclusions

In this paper we gave efficient near linear time approximation algorithms for curve simplification. In particular, we showed that curve simplification under Fréchet error measure can not only produce good simplification, but can also be computed in higher dimensions at no extra effort compared to the two dimensional case, which is not the case for most other simplification algorithms. We also provided experimental results to demonstrate the efficiency of our algorithms.

A natural question arises whether there exists a near-linear algorithm for computing an ε-simplification of size at most $c\kappa(\varepsilon)$, where $c \geq 1$ is a constant. Other questions include whether it is possible to compute the optimal ε-simplification under Hausdorff error measure in near-linear time, or under Fréchet error measure in sub-cubic time. Another open problem is that whether any provably good approximation algorithms exist for curve simplification that preserve the topological type of the curve.

References

[AAS99] Pankaj K. Agarwal, Boris Aronov, and Micha Sharir. Line transversals of balls and smallest enclosing cylinders in three dimensions. *Discrete & Computational Geometry*, 21(3):373–388, 1999.

[AG95] H. Alt and M. Godeau. Computing the frechet distance between two polygonal curves. *International Journal of Computational Geometry*, pages 75–91, 1995.

[Ame92] N. Amenta. Finding a line transversal of axial objects in three dimensions. In *Proc. 3rd ACM-SIAM Symposium on Discrete Algorithms*, pages 66–71, 1992.

[AV00] P.K. Agarwal and K. R. Varadarajan. Efficient algorithms for approximating polygonal chains. *Discrete Comput. Geom.*, 23:273–291, 2000.

[CC92] 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.

[DP73] 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.

[EM01] Regina Estkowski and Joseph S. B. Mitchell. Simplifying a polygonal subdivision while keeping it simple. In *Proc. 17th ACM Symposium on Computational Geometry*, pages 40–49, 2001.

[GHMS93] L. J. Guibas, J. E. Hershberger, J. B. Mitchell, and J.S. Snoeyink. Approximating polygons and subdivisions with minimum link paths. *International Journal of Computational Geometry and Applications*, 3(4):383–415, 1993.

[God91] 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.

[HS94] 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.

[II86] H. Imai and M. Iri. An optimal algorithm for approximating a piecewise linear function. *Information Processing Letters*, 9(3):159–162, 1986.

[II88] 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.

[MO88] 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.

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

[Ram00] E.A. Ramos. Linear optimization queries revisited. In *Proc. 16th Annual ACM Symposium on Computational Geometry*, pages 176–181, 2000.

[Wei97] Robert Weibel. Generalization of spatial data: principles and selected algorithms. In Marc van Kreveld, Jürg Nievergelt, Thomas Roos, and Peter Widmayer, editors, *Algorithmic Foundations of Geographic Information System*. Springer-Verlag Berlin Heidelberg New York, 1997.

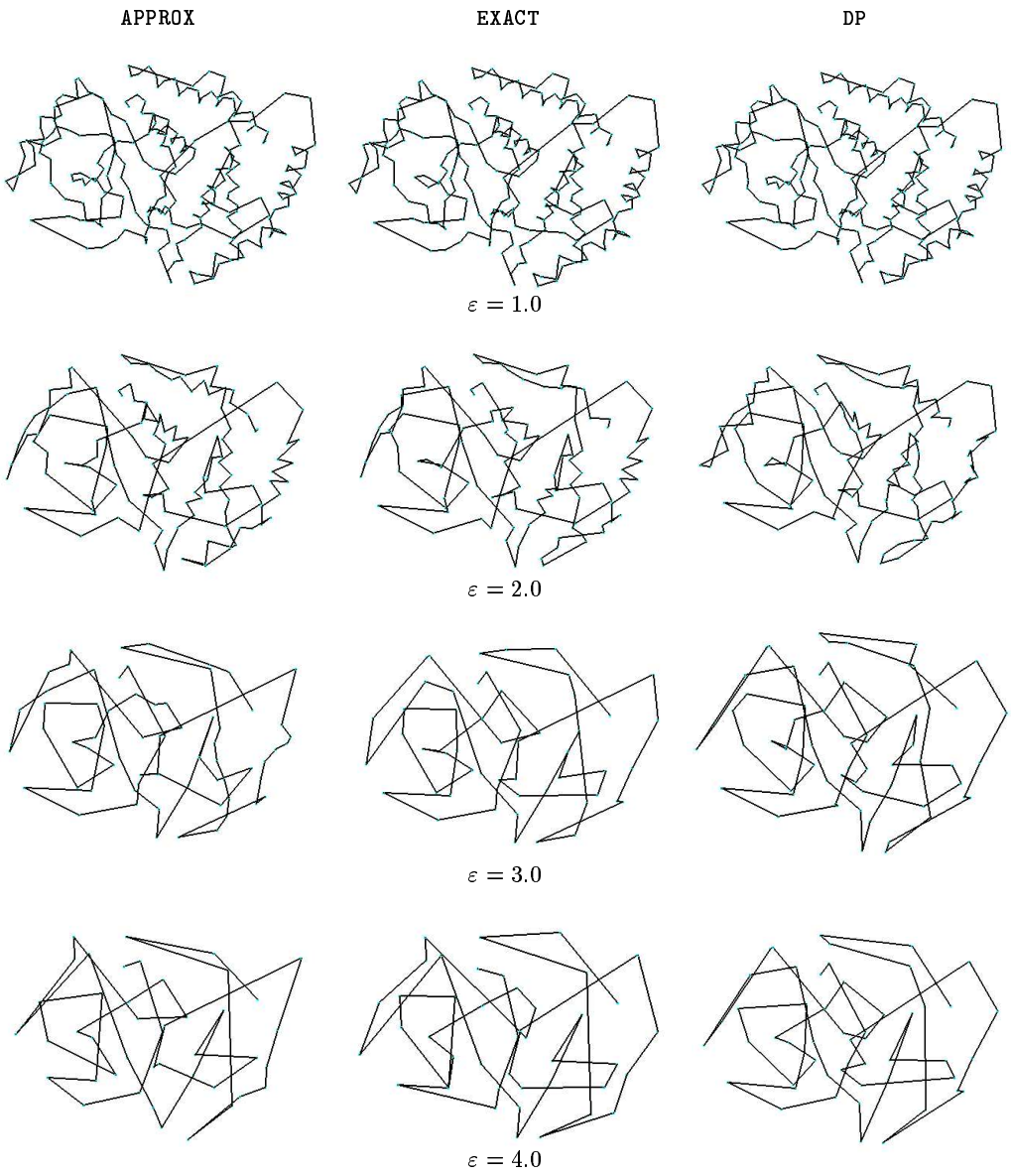


Fig. 8. Simplifications of a protein (1cja) backbone