Curve clustering
hardness and algorithms

Struijs, Martijn

Award date:
2018

EINDHOVEN UNIVERSITY OF TECHNOLOGY

Department of Mathematics and Computer Science
Algorithms Research Group

Curve clustering: hardness and algorithms

Master's Thesis

Martijn Struijs

Supervisors:
dr. J. Nederlof
dr. K.A. Buchin
dr. A. Driemel

Eindhoven, October 2018

Abstract

Clustering point data is a well-studied problem in computer science. Given a set $\mathcal{G}$ of n points in Euclidean space, the Euclidean k-center problem is to compute a set $\mathcal{C}$ of k centers (not necessarily in $\mathcal{G}$) such that the maximum distance between a point in $\mathcal{G}$ and its nearest neighbouring center in $\mathcal{C}$ is minimized. The Euclidean k-median problem is to compute a set $\mathcal{C}$ of k centers such that the sum of distances between the points in $\mathcal{G}$ and their nearest neighbouring center in $\mathcal{C}$ is minimized. In this thesis, we consider these clustering problems for curves under the Fréchet, discrete Fréchet and Dynamic Time Warping (DTW) distance. We show that the k-center problem for curves under the Fréchet and discrete Fréchet distance is NP-hard and even NP-hard to approximate within a factor of $2-\varepsilon$, even for $k = 1$. We then extend our methods to show that computing the k-median under the DTW and squared DTW distance is NP-hard. Finally, we consider algorithms for the related (k, ℓ)-center and (k, ℓ)-median problems, where we additionally require that all center curves in $\mathcal{C}$ have complexity at most ℓ. First, we provide $(1 + \varepsilon)$-approximation algorithms for the $(1, 2)$-center problem for continuous Fréchet distance and the (k, ℓ)-center problem for discrete Fréchet that run in polynomial time for fixed k and ℓ. Then, we give exact algorithms for the (k, ℓ)-center problem for the discrete Fréchet distance in 2D and the $(1, \ell)$-median problem in 1D that run in polynomial time for fixed k and ℓ.

Contents

Contents **v**

1 Introduction **1**
1.1 Problem definition . . . 2
1.2 Related work . . . 5
1.3 Results . . . 6
1.4 Techniques . . . 7

2 Hardness of the MEB problem for Fréchet distance **9**
2.1 NP-hardness . . . 9
2.2 Hardness of approximation . . . 12
2.3 k-wise Fréchet distance . . . 13

3 Hardness of the average curve problem for DTW **15**
3.1 Average-DTW . . . 15
3.2 Squared average-DTW . . . 20
3.3 Multiple sequence alignment and clustering under the DTW distance . . . 23
3.3.1 A reduction for the alignment consensus problem . . . 24
3.3.2 Challenges for a more useful reduction . . . 24

4 Algorithms for (k, ℓ)-center and $(1, \ell)$-median **27**
4.1 Approximating $(1, 2)$-center for the Fréchet distance . . . 27
4.2 Approximating (k, ℓ)-center for the discrete Fréchet distance . . . 29
4.3 Computing (k, ℓ)-center for discrete Fréchet distance in 2D . . . 30
4.4 Computing $(1, \ell)$-center for DTW in 1D . . . 31

5 Conclusions **33**

Bibliography **35**

Chapter 1

Introduction

Many data collected today have a spatial component and are recorded sequentially. The geometry of such data is naturally interpreted as a curve. An important example of this is trajectory data. Trajectory data can be collected with GPS trackers, which can be used to track the routes of taxicabs or the migration patterns of animals such as fish or birds [28].

Clustering is an important tool in the analysis of large datasets. The goal of clustering is to gain insight in the broader patterns available in a dataset by partitioning the dataset into multiple groups that contain curves with high similarity. In this thesis, we consider *centroid-based clustering*, where each cluster is represented by a single center curve. This center is not necessarily a curve in our dataset. The quality of the centroid-based clustering depends on the similarity that each member of a cluster has to the center. If we want to construct an algorithm that produces a clustering of high quality or even analyse such an algorithm, we must first have an objective notion of the 'quality' of a cluster.

A critical part of measuring this quality is the choice of similarity or distance measure for a pair of curves. Most research on clustering represents the data as points in some space. This can be done for curves as well, but there are more specialised similarity measures, which we will consider in this thesis: the Dynamic Time Warping (DTW for short) distance, the (continuous) Fréchet distance and the discrete Fréchet distance.

After we have chosen a distance measure d between curves, we have to decide when the center curve c for a set of curves $\mathcal{G}$ is optimal. One method is to choose a center curve that minimizes $\max_{g\in\mathcal{G}} \mathrm{d}(c, g)$. This is known as the minimum enclosing ball problem w.r.t. the distance d and can be seen as an analogue to the mean of a set of points. Another method is to choose a center curve that minimizes $\sum_{g\in\mathcal{G}} \mathrm{d}(c, g)$. This is called a Steiner point and can be seen as the analogue of the (geometric) median of a set of points.

In this thesis, we focus on the algorithmic problem of finding an optimal center curve for a given set of curves. This can also be seen as the task of clustering a dataset into a single cluster.

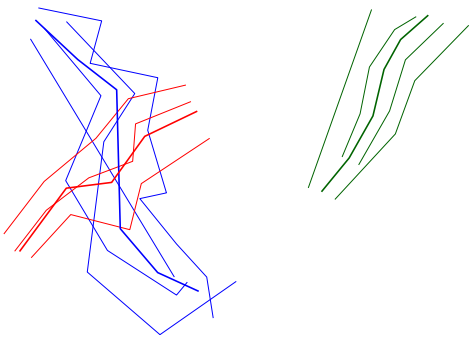

Figure 1.1: An example of a set of curves divided into 3 clusters, with a center curve in bold.

While this problem is interesting on its own, it is directly relevant to the general centroid-based clustering problem in two ways. First, since finding an optimal center for one cluster is a special case of the centroid-based clustering task, showing that this task is NP-hard implies that the clustering task is NP-hard. Secondly, an algorithm for finding an optimal center can be used to construct k clusters by iteratively assigning the curves in the dataset to the nearest of the k centers and updating the centers, similar to the heuristic k-means algorithm.

1.1 Problem definition

To describe the computational problems we consider in this thesis, we first define the distance measures on curves we consider in this thesis, then define criteria for the quality of a center for a set of curves and finally describe the specific problems themselves.

Distance measures on curves

A *polygonal curve* γ is a curve represented by a sequence of vertices $\langle p_1, \ldots, p_m \rangle$, where each contiguous pair of vertices (p_i, p_{i+1}) is connected with the straight-line segment $\overline{p_i p_{i+1}}$. Often, we will simply refer to a polygonal curve as a *curve*. We call the number vertices in a curve the *length* or *complexity* of a curve, denoted by $|\gamma|$.

In this thesis, we consider three methods to measure similarity between a pair of curves γ and γ': the (continuous) Fréchet distance, the discrete Fréchet distance and the Dynamic Time Warping (DTW) distance. Intuitively, the Fréchet distance can be seen as follows: suppose a person walks her dog on a leash, with the person walking on curve γ and her dog on curve γ', both in positive direction without turning back, but possibly at varying speeds. Then the Fréchet distance is the minimum length that leash has to be for both the person and the dog to reach the end of their curve. Formally, the (continuous) *Fréchet distance* is defined using a reparametrisation $f : [0, 1] \to [0, 1]$, a continuous injective function with $f(0) = 0$ and $f(1) = 1$. Let $\mathcal{F}$ be the family of all such reparametrisations, then the continuous Fréchet distance is given by

$$\mathrm{d}_F(\gamma, \gamma') = \inf_{f,g \in \mathcal{F}} \max_{\alpha \in [0,1]} \|\gamma(f(\alpha)) - \gamma'(g(\alpha))\|,$$

where $\|\cdot\|$ denotes the Euclidean norm. The Fréchet distance of a pair of polygonal curves of lengths m and n can be computed in $O(mn \log(mn))$ using the algorithm by Alt and Godau [5].

The *discrete Fréchet* distance is a variant of the Fréchet distance that is more sensitive to the vertices of the curves, but is simpler to compute. To describe this distance, we first define an *alignment* T between the curves γ and γ' as a sequence of pairs of indices $\langle (i_1, j_1), (i_2, j_2) \ldots, (i_t, j_t) \rangle$ with $i_1 = j_1 = 1$, $i_t = |\gamma|$, and $j_t = |\gamma'|$. For each alignment T in the family of alignments $\mathcal{T}$, we require that for all indices $1 < s \leq t$ either

(i) $i_s = i_{s-1} + 1$ and $j_s = j_{s-1}$ or;

(ii) $i_s = i_{s-1}$ and $j_s = j_{s-1} + 1$ or;

(iii) $i_s = i_{s-1} + 1$ and $j_s = j_{s-1} + 1$.

We say that a vertex p_i from γ is *matched* to a vertex p'_j from γ' in an alignment T between those curves if the pair (i, j) occurs in this alignment. Note that in every alignment, each vertex is matched to at least one other vertex, similar to the continuity requirement for the reparametrisation used in the continuous Fréchet distance. Additionally, note that an alignment is *monotone*: if $s \leq s'$, then $i_s \leq i_{s'}$ and $j_s \leq j_{s'}$. This is similar to the injectivity requirement for the reparametrisation.

Using the family $\mathcal{T}$ of all alignments between γ and γ', the discrete Fréchet distance between those curves is defined as

$$\mathrm{d}_{dF}(\gamma, \gamma') = \min_{T \in \mathcal{T}} \max_{(i,j) \in T} \|p_i - p'_j\|.$$

The discrete Fréchet distance of a pair of polygonal curves of length m and n can be computed in $O(mn)$ time with a dynamic programming algorithm based on the following recurrent formula:

$$\mathrm{d}_{dF}(k,l) = \max\left\{\|p_k - p'_l\|, \min\{\mathrm{d}_{dF}(k-1,l), \mathrm{d}_{dF}(k,l-1), \mathrm{d}_{dF}(k-1,l-1)\}\right\},$$

where $\mathrm{d}_{dF}(k,l)$ denotes the discrete Fréchet distance of the curves formed by the first k vertices of γ and the first l of γ'. The base case is $\mathrm{d}_{dF}(1,1) = \|p_1 - p'_1\|$.

The *Dynamic Time Warping* (DTW) distance is also defined using alignments, similar to the discrete Fréchet distance. Instead of taking the maximum distance of all pairs in an alignment, the DTW distance sums over all of them:

$$\mathrm{DTW}(\gamma,\gamma') = \min_{T\in\mathcal{T}} \sum_{(i,j)\in T} \|p_i - p'_j\|.$$

The DTW distance between a pair of curves of length m and n can be computed in $O(mn)$ time in the same manner as the discrete Fréchet distance: with a dynamic programming algorithm based on the following recursion:

$$\mathrm{DTW}(k,l) = \|p_k - p'_l\| + \min\{\mathrm{d}_{dF}(k-1,l), \mathrm{d}_{dF}(k,l-1), \mathrm{d}_{dF}(k-1,l-1)\},$$

where $\mathrm{DTW}(k,l)$ denotes the DTW distance of the curves formed by the first k vertices of γ and the first l vertices of γ'. The base case is $\mathrm{DTW}(1,1) = \|p_1 - p'_1\|$.

Measuring quality of centres

Let d be a distance measure for pairs of curves and G a set of curves. We will discuss two main methods: taking a center of the minimum enclosing ball and taking an average curve.

A *center of a minimum enclosing ball* (MEB) is a center curve c that minimizes the value $\max_{g\in G} \mathrm{d}(c,g)$. This *center curve* is used in the k-center clustering problem: find k curves $c_1,\ldots,c_k$ that minimize $\max_{g\in G} \min_{i=1}^{k} \mathrm{d}(c_i,g)$. So, in this terminology, finding the center of a MEB can be seen as the 1-center clustering problem. See Figure 1.2 for a comparison of a center of a MEB on a point set with one for curves.

The *average curve* is the center curve c that minimizes the value $\sum_{g\in G} \mathrm{d}(g,c)$ (although this is the total sum of distances to c and not the average, c also minimizes $\frac{1}{|G|}\sum_{g\in G} \mathrm{d}(g,c)$, as we scale the total withNote that in case the median lies in between two data points, note that both of these points attain the optimum a factor independent of c). This center is used in the k-median clustering problem: find k curves $c_1,\ldots,c_k$ that minimize $\sum_{g\in G} \min_{i=1}^{k} \mathrm{d}(c_i,g)$. So, in this terminology, finding the average curve can be seen as the 1-median problem. For point sets under the Euclidean distance, a solution to the 1-median problem is known as a geometric median, which coincides with the median of an ordered set of points in 1D. See Figure 1.3 for a comparison of a geometric median of a point set with an average curve.

For the two problems above, the center c does not need to be an input curve from G. This means that an optimal center could have a much larger complexity than that of any curve in the dataset, which can be undesirable when we want c to be representative for (a subset of) G. Therefore, we consider an extension to the two problems above where we additionally require that each center has a complexity of at most ℓ: we call the k-center problem restricted to center curves of complexity at most ℓ the (k,ℓ)-`center` problem and the k-median problem restricted to center curves of complexity at most ℓ the (k,ℓ)-median problem.

Another method to measure the quality of a cluster is to generalise d from a pairwise measure to a *k-wise measure* that assigns a value to k curves for any $k \geq 2$. For example, the discrete Fréchet distance can be generalised by generalising the alignment to a sequence of k-dimensional vectors of indices and taking the maximum of the k pairwise distances as the cost of a single point in the alignment. While this notion of quality does not explicitly use a center, the alignment can be seen as implicitly defining a center and it turns out that the problem of computing the k-wise (discrete) Fréchet distance and finding the center of a MEB under these distances are closely related.

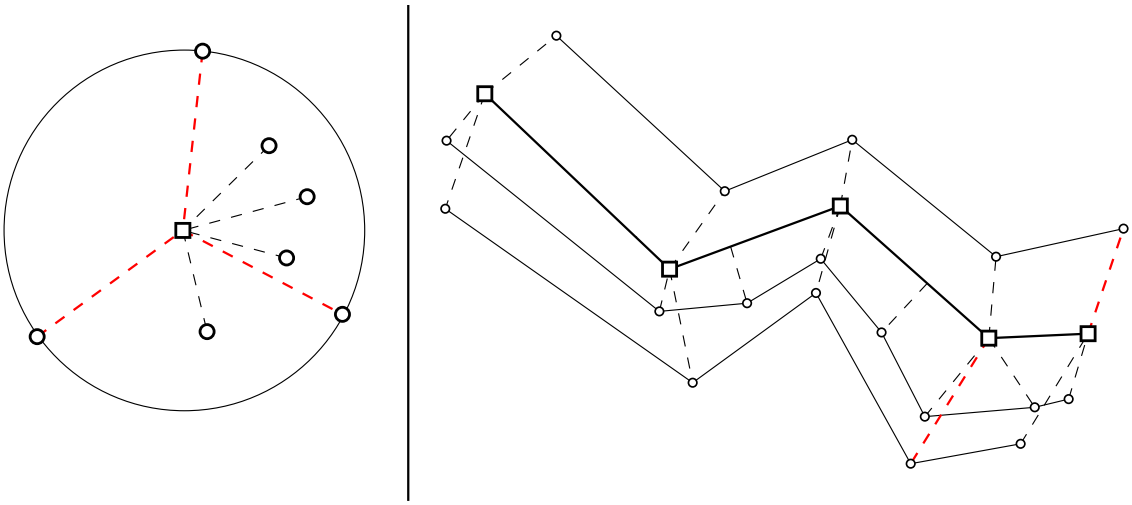

Figure 1.2: A center of a minimum enclosing ball on a point set under Euclidean distance (left) and on a set of curves under Fréchet distance (right). The red lines indicate the maximum distance that determines the cost of the center curve.

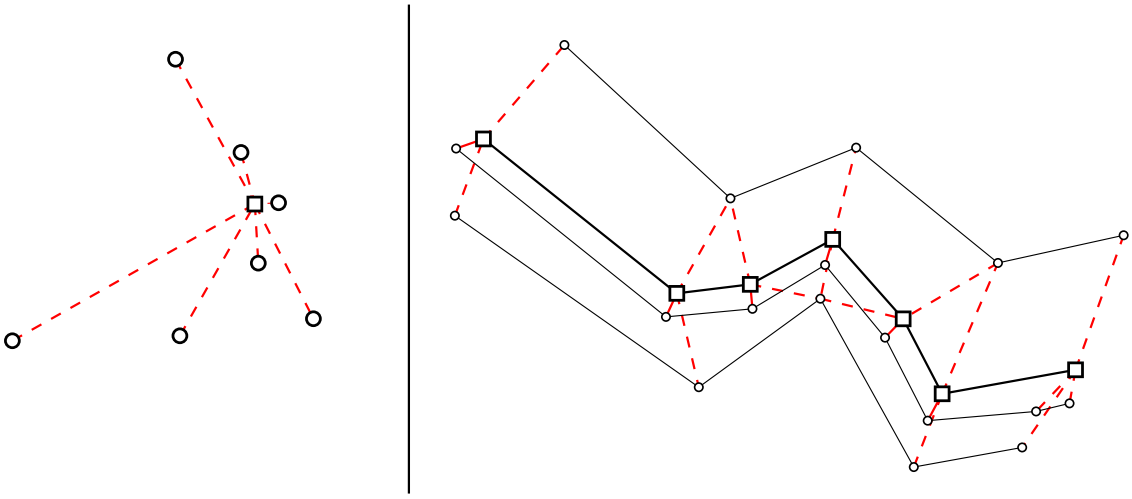

Figure 1.3: A geometric median of a point set under the Euclidean distance (left) and an average curve under the DTW distance (right). The red lines indicate the distances to the center curve, whose sum determines the cost of the center curve.

1.2 Related work

Data clustering is a widely-researched task with various methods and applications [32]. A common usage is exploratory data analysis. Many clustering algorithms interpret the data as point sets in Euclidean space. We can do this for curves as well: we can interpret a 1D curve of complexity k as a vector in a k-dimensional Euclidean space by associating each vertex on the curve with a corresponding entry on the vector.

While this is in fact a common method for e.g. time series [22] (which can be interpreted as 1D curves), this has some downsides. One problem is that we cannot simply compare curves of different complexity. Another is that this distance measures only the distances between the vertices visited at the same time-stamp. This means that the distance of the curves is highly sensitive to their sampling time. This sensitivity has also been shown experimentally [29]. A solution is to allow the distance measure to compare points at different timestamps, while preserving the order. This is done in the three distance measures we consider.

The (continuous) Fréchet distance is useful when comparing curves with different sampling rates, such as in matching GPS tracks from vehicles to a map of the roads they drive on [7]. The discrete Fréchet distance, introduced in [19], is mostly used as an easier to compute approximation to the continuous Fréchet distance, such as in [23]. The DTW distance has been introduced to compare acoustic signals [26]. The DTW distance can be applied in a variety of contexts, such as comparing RNA expression data in biology, synchronising and monotoring batch processes in chemical engineering , comparing handwritten signatures and comparing fingerprints [21]. Petitjean and Gançarski [24] study the problem of finding an average 1D curve under the squared DTW distance and develop a heuristic algorithm that finds average curves of complexity similar to the input curves.

The simple $O(mn)$ algorithm to compute the discrete Fréchet distance for a pair of curves with complexity m and n has been improved to subquadratic with the algorithm by Agarwal et al. [1], which runs in $O(\frac{mn \log\log n}{\log n})$ time when $n \geq m$. While the $O(n^2 \log(n))$ algorithm for curves of complexity n by Alt and Godau for computing the continuous Fréchet distance did not have any improvements on the general computation problem for over 20 years, recently Buchin et al. [11] improved the running time with a randomized algorithm with expected $O(n^2\sqrt{\log n}(\log\log n)^{3/2})$ total running time.

However, Bringmann [9] shows that there can be no strongly subquadratic algorithms for either the discrete or continuous Fréchet distance in 2D unless the SETH (Strong Exponential Time Hypothesis) fails, i.e. no algorithm that runs in $O(n^{2-\delta})$ for any $\delta > 0$. This left the case in 1D open, but Bringmann and Mulzer [10] show that any strongly subquadratic 1.399-approximation algorithm for the discrete Fréchet distance in 1D would violate the SETH. This is again improved by Buchin et al. [13], who show that there exist no strongly subquadratic algorithm that achieves an approximation ratio of 3 or better unless the SETH fails for both the discrete and continuous Fréchet distance in 1D (and higher dimensions). Driemel et al. [16] give an $(1+\varepsilon)$-approximation algorithm that runs in linear time for the realistic input of c-packed curves.

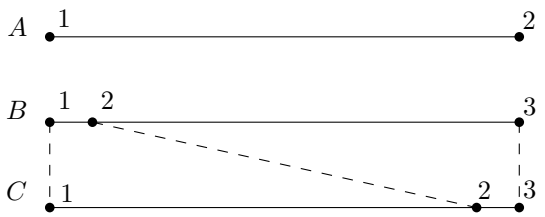

Figure 1.4: An example of similar 1D-curves that differ only in their vertices, yet are dissimilar when interpreted as points in Euclidean space. Curve A cannot even be compared with the other curves in this interpretation, since it has different complexity. Curves B and C have a large distance, as the vertices at time 2 cannot be compared to different points on the curves.

While there is some research on heuristic improvements to compute the DTW distance [27, 4], there are also some more theoretical results. Bringmann and Künnemann [8] show that there are no strongly subquadratic algorithms for the DTW distance unless the SETH fails. Agarwal et al. [2] provide near-linear time algorithms for $(1+\varepsilon)$-approximation algorithms to compute the DTW for specific input classes of curves in $\mathbb{R}^d$.

Har-Peled and Raichel [20] generalise the weak Fréchet distance to simplical complexes and apply that to compute a solution to the 1-center problem for the weak Fréchet distance in $O(m^n)$ time and provide an $(1+\varepsilon)$-approximation for c-packed curves in time $\tilde{O}(n \log n)$, where n is the number of input curves and m their maximum complexity. Ahn et al. [3] provide algorithms for finding a 1-center curve for the Fréchet distance that consists only of vertices of the input curves. Performing clustering with the Fréchet distance in 1D was first considered by Driemel et al. [17]. They provide $(1+\varepsilon)$-approximation algorithms for the (k,ℓ)-center and (k,ℓ)-median that run in $O(mn \log m)$ for constant k, l and ε, where n is the number of input curves and m the maximum complexity among those curves. They also show that (k,ℓ)-center and (k,ℓ)-median problem are NP-hard when ℓ is part of the input. In a recent paper [12], we show that the (k,ℓ)-center problem is NP-hard for the discrete and continuous Fréchet distance, even for $k=1$. Some results of Chapter 2 are included in this paper. We show that the (k,ℓ)-center problem is NP-hard to approximate within $2-\varepsilon$ for the discrete Fréchet distance and $1.5-\varepsilon$ for the continuous Fréchet distance in 1D. Additionally, we show that the (k,ℓ)-center problem for the discrete and continuous Fréchet distance with curves in 2D is NP-hard to approximate within a factor around 2.598 and provide a polynomial time 3-approximation algorithm for those problems.

1.3 Results

The main results of this thesis are to show that some variants of finding optimal centers for curves are NP-hard or even NP-hard to approximate and providing a few algorithms that solve variants in polynomial time. First, we show that finding a minimum enclosing ball under the discrete and continuous Fréchet distance for curves in 1D is NP-hard in Chapter 2. Note that this implies hardness in general dimension, as we can always embed 1D curves on a line for all higher dimensions. This result is included in the paper [12]. We then extend this result to show that finding a minimum enclosing ball under the discrete and continuous Fréchet distance for curves in 1D is NP-hard to approximate within a factor of $2-\varepsilon$ for any $\varepsilon > 0$. Note that selecting any of the input curves gives a 2-approximation, so this result is tight unless P = NP. We use a connection between the MEB problem and computing the k-wise distance for the discrete Fréchet distance to show that computing the k-wise discrete Fréchet distance for curves in 1D is NP-hard to approximate within a factor $2-\varepsilon$. Extending the methods used to show hardness for the MEB problem, we show in Chapter 3 that finding an average curve under the DTW distance and the squared DTW distance is NP-hard.

While these problems are NP-hard in general, in Chapter 4, we provide some polynomial-time exact and approximation algorithms for some cases of a variant of this problem, introduced in [17], where we only consider center curves with at most ℓ vertices. We first give an $(1+\varepsilon)$-approximation algorithm for the $(1,2)$-center problem for the continuous Fréchet distance in polynomial time for any fixed dimension. We then provide an $(1+\varepsilon)$-approximation algorithm for the (k,ℓ)-center problem for the discrete Fréchet distance that runs in $O(mn \log m)$ for fixed dimension, k, ℓ and ε. After that, we provide an algorithm to exactly solve the decision version of the (k,ℓ)-center problem under the discrete Fréchet distance for curves in 2D in polynomial time for fixed k and ℓ. Finally, we provide an exact algorithm for the $(1,\ell)$-median problem under the DTW distance for curves in 1D in polynomial time for fixed ℓ.

1.4 Techniques

We briefly discuss the main techniques and ideas used to obtain the results in this thesis. All proofs for NP-hardness in this thesis are a reduction from the decision version of the shortest common supersequence problem (SCS): Given a set of sequences $\mathcal{S} = \{s_1, \ldots, s_n\}$ over a binary alphabet $\Sigma = \{A, B\}$ and a positive integer r, the problem is to find a string s^* of length at most r that is a supersequence of all strings in $\mathcal{S}$. This problem is NP-hard [25].

The main idea of the reductions is to transform each sequence s_i into a curve $\gamma(s_i)$ such that a curve g^* with a small enough distance to $\gamma(s_i)$ corresponds to a string s^* that is a supersequence of s_i. To achieve this, we construct $\gamma(s_i)$ from s_i by replacing each A and B character with a different subcurve and separate them with 'buffer' subcurves. This requires the potential center curve c to have subcurves corresponding to a A or B character or buffer as well. The subcurves are chosen such that parts corresponding to A and B in $\gamma(s_i)$ can only be matched with the respective part in c. The buffers are used to ensure A or B parts in c are used at most once and that the parts not corresponding to the letters in s_i can be matched to some parts of $\gamma(s_i)$. To ensure that a center curve c has length at most r, we add some additional curves to the input.

This idea can be implemented in a straightforward manner in the hardness proof for the discrete Fréchet distance, as we can directly set an upper bound on the distance between the individual matched vertex pairs. Showing the hardness the average curve under the DTW distance is more complicated, since we can only set an upper bound on the sum of the distances of all matched vertex pairs and have to spend some effort to get an upper bound on the distances between individual matched pairs from that.

The main idea for the $(1 + \varepsilon)$-approximation algorithms in Chapter 4 is to take some center curves from a constant factor approximation algorithm and show that an optimal set of center curves has vertices that lie close to the vertices of the set of approximate center curves. Then, we can divide the area in which the vertices of the optimal solution lie with a grid with a resolution proportional to ε. Since all center curves have complexity at most ℓ, we can iterate over all possible sets of center curves that use vertices of the grid. This yields an $(1 + \varepsilon)$-approximation.

The exact algorithms in Chapter 4 both rely on the fact that there are a bounded number of locations for the vertices of an optimal sequence. In the algorithm for the discrete Fréchet distance, we only have to look at one point for each maximal overlapping region in an arrangement of disks. In the algorithm for the DTW distance, we only have to look at the vertices of our input curves.

Chapter 2

Hardness of the MEB problem for Fréchet distance

2.1 NP-hardness

In this section, we will show that finding the center of a minimum enclosing ball for the discrete and continuous Fréchet distance is NP-hard. We use a reduction from the NP-hard Shortest Common Supersequence (SCS) problem [25]: Given a set of n sequences $S = \{s_1, \ldots, s_n\}$ over a binary alphabet $\Sigma = \{0, 1\}$ and a positive integer r, does there exists a sequence s^* of length at most r such that s^* is a supersequence of all sequences in S?

For our construction, first define the following points in $\mathbb{R}$:

$$g_A := p_{-3}, \qquad g_B := p_3, \qquad g_a := p_{-1}, \qquad g_b := p_1.$$

The distance between these points is the standard Euclidean distance on $\mathbb{R}$: $d(p_i, p_j) := |i - j|$. Given a string $s \in S$ over the binary alphabet $\{A, B\}$, we map each character to a curve over $\mathbb{R}$ as follows

$$A \to (g_a g_b)^t g_A (g_b g_a)^t$$
$$B \to (g_b g_a)^t g_B (g_a g_b)^t,$$

where t is the length of the supersequence sought for the strings in S. The curve $\gamma(s_i)$ is constructed by concatenating the curves resulting from the previous mapping. Here, we call g_A and g_B *letter gadgets* and the subcurves in between, $(g_a g_b)^t$ and $(g_b g_a)^t$, *buffer gadgets*. Call the resulting set of curves G, i.e. $G = \{\gamma(s) \mid s \in S\}$.

To get a reduction for the MEB problem, we use a technique similar to the proof in [30, Theorem 1]. Add the set $R_{i,j} = \{A^i, B^j\}$ with

$$A^i = g_b (g_A g_b)^i$$
$$B^j = g_a (g_B g_a)^j,$$

We will show that the instance (S, t) of SCS is a YES-instance if and only if there exists $(i, j) \in I_t := \{(i, j) \in \mathbb{N}^2 \mid i, j \geq 0, i + j = t\}$ such that $(G \cup R_{i,j}, 1)$ is a YES-instance of the MEB problem for discrete Fréchet. Since the number of choices for i, j is linear in t, the number of different instances of MEB is polynomial in the input size of the SCS instance[1]. This means that there is a truth-table reduction from SCS to MEB for discrete Fréchet[2].

[1] Although the input size of t is $\log t$, SCS on binary alphabets is non-trivial only when $t \leq 2 \max_{s \in S} |s|$ so we can assume that t is at most linear in the input size of the SCS instance

[2] Note that this procedure doesn't provide a many-one reduction, as a many-one reduction is a special (i.e. limited) case of a truth-table reduction

Lemma 1. *If (S,t) is a YES-instance of SCS, then there exists $i,j \in I_t$ such that $(G \cup R_{i,j}, 1)$ is a YES-instance of MEB under the discrete Fréchet distance.*

Proof. If (S,t) is a YES-instance of SCS, then there is a string s^* of size t that is a supersequence of all strings in S.[3]

Construct the curve c^* of length $2t+1$, with for each $i \in \{1,\ldots,2t+1\}$,

$$c_i^* := \begin{cases} p_0 & i \text{ is odd} \\ p_{-2} & i \text{ is even and } s^*_{i/2} = A \\ p_2 & i \text{ is even and } s^*_{i/2} = B \end{cases}.$$

That is, c^* has a vertex at p_{-2} for every A in s^*, a vertex at p_2 for every B in s^* and with p_0 in between and p_0 as the first and last vertex of c^*.

Let $\gamma(s) \in G$. We will create an alignment with c^*. Since s is a subsequence of s^*, every letter gadget in $\gamma(s)$ can be matched with the corresponding letter in c^* within distance 1. All other p_2 and p_{-2} vertices in c^* can be matched with a vertex g_a and g_b in a buffer of $\gamma(s)$ respectively. All other vertices in the buffer gadgets of $\gamma(s)$ can be matched with some p_0 in c^*. So, we have an alignment between $\gamma(s)$ and c^* where all matched vertices have distance at most 1, so $d_{DF}(g,c^*) \leq 1$ for all $g \in G$.

Set i to the number of occurrences of A in s^* and j to the number of occurrences of B in s^*. So, $i+j=t$ and $i,j \geq 0$, so $(i,j) \in I_t$. Since c^* contains a vertex at p_{-2} exactly i times, we can match those characters to g_A in A^i and all vertices in c^*, at points p_0 or p_2, with g_a. Since $d(p_{-2},g_A) = d(p_0,g_a) = d(p_2,g_a) = 1$, the discrete Fréchet distance between A^i and c^* is at most 1. Analogously, we get that the discrete Fréchet distance between B^j and c^* is at most one. So, $d_{DF}(g,c^*) \leq 1$ for all $g \in G \cup R_{i,j}$ for some $i,j \in I_t$. □

Lemma 2. *If there exists $i,j \in I_t$ such that $(G \cup R_{i,j}, 1)$ is a YES-instance of MEB under the discrete Fréchet distance, then (S,t) is a YES-instance of SCS.*

Proof. If there exists $i,j \in I_t$ such that $(G \cup R_{i,j}, 1)$ is a YES-instance, then there exist a center curve c^* such that $d_{DF}(g,c^*) \leq 1$ for all $g \in G \cup R_{i,j}$. So, there exists a alignment between c^* and A^i in which all matched pairs have distance at most 1. Note that there exists no point p such that $d(p,g_b) \leq 1$ and $d(p,g_A) \leq 1$. This means that for every point in c^*, it is either matched only to some points at g_A or some points at g_b. Since every pair of points at g_A has at least one point at g_b in between and the matching is monotone, a point in c^* matched to some points g_A can be matched to at most 1 point in A^i. The same holds for the points matched to g_b. So, every point in c^* is matched to exactly one point in A^i.

This means we can partition the points of c^* into $2i+1$ parts, where the k-th part contains all points of c^* matched to the k-th point of A^i. We can do the same with the matching between c^* and B^j to get a partition of $2j+1$ parts. A point in c^* cannot be both in a part corresponding to an g_A and in another corresponding to g_B. We can 'combine' these two partitions into one partition of at most $2(i+j)+1$ parts (see Figure 2.1), as follows: if a point in c^* is matched to g_A, put it in the part corresponding to that point, we call this an A-part. If a point in c^* is matched to g_B, put it in the part corresponding to that point, we call such a part an B-part. If a point is matched to neither g_A nor g_B match it to the 'buffer' between the part of the nearest point of smaller index in c^* in an A/B-part and the nearest point of greater index in an A/B-part. (If such an index doesn't exist, identify it by only one A/B-part. There are at most two such parts) We call such a part a buffer-part. Note that by construction, there are exactly $i+j=t$ A/B-parts and at most $i+j+1$ buffer-parts.

Construct the string s^* by removing the buffer parts from the partition of c^*, replacing the A-parts with the A character and the B-parts with B character. As there are exactly $i+j=t$ A/B-parts, $|s^*| = t$. We complete the proof by showing that s^* is a supersequence of all sequences in S.

[3]While we have a YES-instance for size at most t, we can always add 'dummy' characters to get one of size t. So, we can assume that s^* has size exactly t without loss of generality.

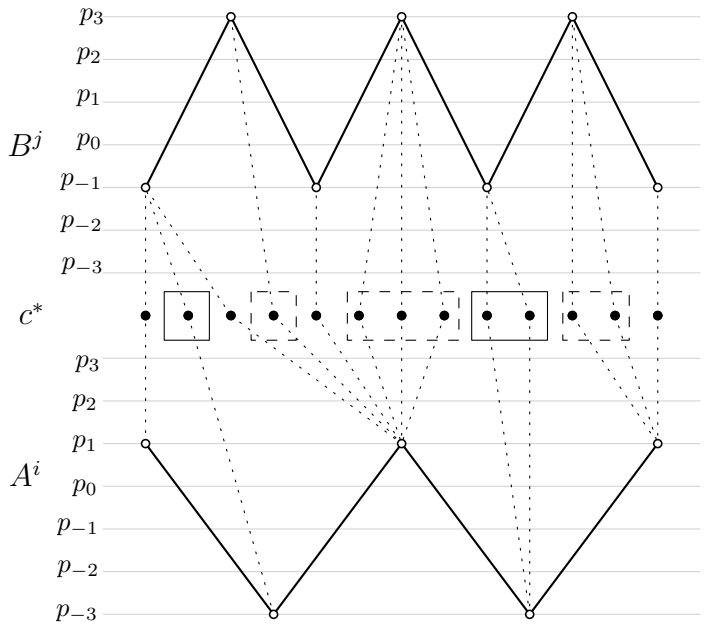

Figure 2.1: The partition of a center curve c^* from Lemma 2, based on matchings between the center curve c^* and curves A^i and B^j. Dashed boxes indicate an A-part, normal boxes indicate a B-part, and the points without boxes are in a buffer part.

Let $\gamma(s) \in G$. As $d_{DF}(\gamma(s), c^*) \leq 1$, there exists a matching between $\gamma(s)$ and c^* such that all matched pairs have distance at most 1. This means an A-part cannot be matched to multiple letter gadgets, since the matching is monotone and the buffer separating the letter gadgets contains the point g_b. Analogously, the B-parts cannot be matched to multiple letter gadgets. Furthermore, each letter gadget must be matched to an A/B-part with the corresponding letter. This means that the sequence of letter gadgets in $\gamma(s)$ is matched one-to-one with a subsequence of the sequence of A/B-parts in c^* with corresponding letters and therefore s is a subsequence of s^*. □

So, we have a polynomial time reduction from SCS to MEB under discrete Fréchet distance and we can conclude with the following theorem.

Theorem 3. *The minimum enclosing ball problem with discrete Fréchet distance is NP-hard.*

The MEB problem with *continuous* Fréchet distance can be shown to be NP-hard with the same construction as above. Denote the continuous Fréchet distance between curves a and b by $d_{CF}(a, b)$. First, note that for the center curve c^* constructed in Lemma 1, we have $d_{CF}(c^*, g) \leq d_{DF}(c^*, g) \leq 1$ for all $g \in (G \cup R_{i,j})$, since the continuous Fréchet distance is always smaller than the discrete Fréchet distance on polygonal curves. So, the result of Lemma 1 also holds for the continuous Fréchet distance.

The continuous version of Lemma 2 also holds. First note that for every A-part, there must be a point on the center curve c^* at p_{-2}, since it it is matched to some g_A and a point on curve B^j. This point is at the largest distance from 0 on c^* and the neighbouring parts are either an B- or a buffer part, which are strictly closer to 0. This means there is a vertex at p_{-2} on c^* for every A-part. Analogously, there is a vertex at p_2 on c^* for every B-part. We again can create the sequence s^* of length t by replacing the A/B-parts with A/B characters. As the letter gadgets in the curves $\gamma(s)$ must be matched to some vertex at p_2 or p_{-2} representing the A/B-part, we get that the sequence $s \in S$ is a subsequence of s^*.

Theorem 4. *The minimum enclosing ball problem with continuous Fréchet distance is NP-hard.*

Note that in [12], a slightly different construction is required to show NP-hardness for the center problem with continuous Fréchet distance. The reason for this is that they use the total length of the center curve is to determine the length of the supersequence and some buffer vertices in the center curve can be omitted in the continuous case, unlike in the discrete case.

2.2 Hardness of approximation

In this section, we show that the minimum enclosing ball problem is NP-hard to approximate within factor $2-\varepsilon$ for both the discrete and continuous Fréchet distance, for any constant $\varepsilon > 0$. To show this, we use the same construction as before.

Lemma 5. *If there is a curve within discrete Fréchet distance δ of all curves in $G \cup R_{i,j}$, for some $\delta < 2$, then there exists a curve within discrete Fréchet distance 1 of all curves in $G \cup R_{i,j}$.*

Proof. Let c be a curve that is within discrete Fréchet distance δ of all curves in $G \cup R_{i,j}$ for some $\delta < 2$. Let p be a vertex in c and consider the matchings between c and the curves in $G \cup R_{i,j}$. Since these matchings attain a discrete Fréchet distance of at most δ, p is within distance δ of any vertex in $G \cup R_{i,j}$ it is matched to. Note that all vertices in the curves in $G \cup R_{i,j}$ lie at g_a, g_b, g_A or g_B. Furthermore, no single vertex can be within distance δ of both vertices in the pairs (g_a, g_B) (g_b, g_A) or (g_A, g_B). This means that the set of vertices p is matched with is a subset of either $\{g_b, g_B\}$, $\{g_a, g_A\}$ or $\{g_a, g_b\}$. In case p is matched only to $\{g_b, g_B\}$, replacing p with a point at p_2 results in p being at distance 1 from all vertices that it is matched with. Similarly, replacing p with p_{-2} if p is matched only with $\{g_a, g_A\}$ and replacing p with p_0 if p is matched only with $\{g_a, g_b\}$ results in p being at distance 1 from all vertices that it is matched with.

So, p can be replaced by a vertex such that it has distance 1 to all vertices from curves in $G \cup R_{i,j}$ it is matched with. We can replace all vertices in c by such a vertex to obtain a curve of which all vertices are matched with vertices from $G \cup R_{i,j}$ at distance 1. Hence, this curve lies within distance 1 of all curves in $G \cup R_{i,j}$. □

Lemma 5 tells us that if we have an algorithm that guarantees an approximation for the MEB problem with factor at least $2-\varepsilon$ under the discrete Fréchet distance, then we can decide whether an instance $(G \cup R_{i,j}, 1)$ of the MEB problem is a YES-instance. Using the reduction to SCS above, this results in the following theorem.

Theorem 6. *The MEB problem for the discrete Fréchet distance is NP-hard to approximate within approximation factor $2-\varepsilon$ for any $\varepsilon > 0$.*

For the continuous Fréchet distance, we can use the same approach.

Lemma 7. *If there is a curve within continuous Fréchet distance δ of all curves in $G \cup R_{i,j}$, for some $\delta < 2$, then there exists a curve within discrete Fréchet distance 1 of all curves in $G \cup R_{i,j}$.*

Proof. Let c be a curve that is within Fréchet distance δ of all curves in $G \cup R_{i,j}$ for some $\delta < 2$. Let $g \in G \cup R_{i,j}$. Note that there exist no points that are within distance δ of the two points g_A and g_b or within distance δ of the two points g_B and g_a. This means each g_A vertex on g can only be matched to points on c with coordinate < -1 and that points on c with coordinate < -1 cannot completely match with a g_B vertex, g_b vertex or a $(g_a g_b)^t$ subcurve. Symmetrically, each g_B vertex on g can only be matched to points on c with coordinate > 1 on c and points on c with coordinate > 1 cannot completely match a g_A vertex, g_a vertex or $(g_a g_b)^t$ subcurve.

Construct the curve c' that starts and ends with p_0, has a vertex p_{-2} when there is a maximal subcurve of vertices with coordinate < -1 in c, a vertex p_2 when there is a maximal subcurve of vertices with coordinate > 1 in c and vertices p_0 in between. We will show that c' lies within *discrete* Fréchet distance 1 of all curves in $G \cup R_{i,j}$.

First, $\mathrm{d}_{dF}(c', A^i) \leq 1$, since each vertex g_A on A^i was matched to point at coordinate < -1 that has been replaced by g_{-2} on c' and all other vertices on c' can be matched to the g_b vertices on A^i. By symmetry, $\mathrm{d}_{dF}(c', B^j) \leq 1$. Similarly, for $g \in G$, then each vertex g_A in g is matched

to a p_{-2} vertex on c', each g_B vertex to a p_2 vertex on c'. Each g_a that was matched to some point with coordinate < -1 in c can be matched to a vertex p_{-2} and each g_b that was matched to some point with coordinate > 1 on c can be matched to a p_2 vertex on c'. The remaining g_a and g_b vertices on g can be matched to a vertex at p_0 and so $\mathrm{d}_{dF}(c', g) \leq 1$.

Since the discrete Fréchet distance is an upper bound of the continuous Fréchet distance, $\mathrm{d}_F(c', g) \leq \mathrm{d}_{dF}(c', g) \leq 1$ for all $g \in G \cup R_{i,j}$. □

Similar to the discrete Fréchet case, we get the following theorem as a consequence of Lemma 7.

Theorem 8. *The MEB problem for the discrete Fréchet distance on curves in 1D is NP-hard to approximate within approximation factor $2 - \varepsilon$ for any $\varepsilon > 0$.*

2.3 k-wise Fréchet distance

The MEB problem is related to computing the discrete Fréchet distance generalised to multiple curves. The k-wise discrete Fréchet distance on a set $G = \{\gamma_1, \ldots, \gamma_k\}$ is defined using the alignment $T = \langle t_1, \ldots, t_m \rangle$ of k-dimensional vectors, such that $t_1 = (1, 1, \ldots, 1)$, $t_m = (|\gamma_1|, |\gamma_2|, \ldots, |\gamma_k|)$ and $t_{i+1} - t_i \in \{0, 1\}^k \setminus \{\vec{\mathbf{0}}\}$ for all $1 \leq i < m$. This alignment can be seen as an increasing path through a k-dimensional matrix where each step goes to a cell that touches the previous one. Using the set $\mathcal{T}$ of all such alignments on k-dimensional vectors, the k-wise discrete Fréchet distance is defined by

$$\mathrm{d}_{dF}(G) = \min_{T \in \mathcal{T}} \max_{t_l \in T} \max_{1 \leq i,j \leq k} \|\gamma_i(t_{li}) - \gamma_j(t_{lj})\|.$$

This generalisation is similar to the generalisation to the k-wise continuous Fréchet distance in [18], which uses reparameterisations instead of alignments. To recall, a reparameterisation $f : [0, 1] \to [0, 1]$ is a continuous injective function with $f(0) = 0$ and $f(1) = 1$. Call the family of all such reparameterisations $\mathcal{F}$, then

$$\mathrm{d}_F(G) = \inf_{f_1, f_2, \ldots, f_k \in \mathcal{F}} \max_{t \in [0,1]} \max_{1 \leq i,j \leq k} \|\gamma_i(f_i(t)) - \gamma_j(f_j(t))\|.$$

Now, we will show that the computing the k-wise discrete Fréchet distance is related to computing the center of a MEB under the discrete Fréchet distance.

Lemma 9. *Let be G be a finite set of curves in 1D and $x \in \mathbb{R}$. There exists a 1D curve c such that $\mathrm{d}_{dF}(c, g) \leq x$ for all $g \in G$ if and only if $\mathrm{d}_{dF}(G) \leq 2x$.*

Proof. Suppose there exist a curve c be such that $\mathrm{d}_{dF}(g, c) \leq x$ for all $g \in G$. Take a vertex p in c. Since the distance $\|\cdot\|$ for the points in the curves in G satisfies the triangle inequality, we have that for all points on the curves in G that are matched to p have distance at most $2x$ to each-other. It is possible that p is matched to k points on the same curve from G for some $k > 1$. In that case, we can replace the vertex p in c^* by the subcurve consisting of p copied k times, to ensure each point on the center curve is matched with exactly one point on every curve in G. If we would not do this, then when at least two curves both have at least two vertices matched to the same vertex in c^*, we cannot match all those vertices to each-other in a valid warping path, as this breaks monotonicity. Since the maximum distance over the matchings doesn't change by this copying operation, the discrete Fréchet distance remains the same. So we can construct a center curve c' from c such that $\mathrm{d}_{dF}(g, c') \leq x$ and there is a matching achieving this distance between each $g \in G$ and c' such that each point in c' is matched to exactly one point in g.

From c', we can construct an alignment for the curves in G by matching all points with each-other that are matched to the same point in c'. Every matching in this path has maximum distance among the matched points of at most $2x$. So, we have that $\mathrm{d}_{dF}(G) \leq 2x$.

Suppose $\mathrm{d}_{dF}(G) \leq 2x$. Note that in 1D, if we have a finite set $S \subset \mathbb{R}$ such that $|a - b| \leq 2x$ for all $a, b \in S$, then we can take $m = \min(S)$ and $M = \max(S)$, set $d = \frac{1}{2}(m + M)$ and get $|d - s| \leq \max(|d - M|, |d - m|) = |m - M|/2 \leq x$ for all $s \in S$. So, as each group of vertices

matched in the alignment of $\mathrm{d}_{dF}(G)$ has maximum pairwise distance $2x$, there exists a vertex with distance at most x to all aligned vertices. So, we can construct a curve c from the center point of every subset in the alignment for G, such that $\mathrm{d}_{dF}(c, g) \leq x$ for all $g \in G$. □

Note that Lemma 9 does not extend to curves in higher dimensions. In particular, when $\mathrm{d}_{dF}(G) \leq 2x$ for curves in 2D, there does not necessarily exist a c such that $\mathrm{d}_{dF}(c, g) \leq x$ for all $g \in G$. Consider $G = g_1, g_2, g_3$ where g_1, g_2, g_3 are curves of a single point, each forming one vertex of an equilateral triangle with side length 2 and the pointwise distance is the 2D Euclidean distance. Then we have that $\mathrm{d}_{dF}(G) \leq 2$. But there exists no single point that has distance at most 1 to g_1, g_2, g_3: if a point a has distance at most 1 to both g_1 and g_2, then a lies on the line between g_1 and g_2, which means $d(a, g_3) > 1$.

Of course, we can still use this lemma to show that the k-wise discrete Fréchet distance is hard, as all input curves in our hardness proofs are in 1D. In particular, Lemma 9 implies that the decision problems for computing the k-wise distance and the center of the MEB are equivalent for the discrete Fréchet distance on 1D curves, which leads to the following theorems.

Theorem 10. *Computing the k-wise discrete Fréchet distance of a set of 1D curves is NP-hard.*

Theorem 11. *Approximating the k-wise discrete Fréchet distance of a set of 1D curves is within a factor of $2 - \varepsilon$ for any $\varepsilon > 0$ is NP-hard.*

Chapter 3

Hardness of the average curve problem for DTW

A construction similar to that of the previous section can be used to show hardness of the average-DTW problem and its variant that minimizes $\sum_{i=1}^{n} \mathrm{DTW}(c^*, s_i)^2$, the squared average-DTW problem. Note that when the vertices of the curves are in an Euclidean space of dimension at least 2, the average-DTW problem on curves containing only one point is equivalent to the Fermat-Weber problem, of which the solution in general cannot be expressed in radicals over the rationals [6] and therefore cannot be exactly computed in computational models where the root of an algebraic equation is obtained using arithmetic operations and the extraction of kth roots.

However, finding an average curve for the DTW distance over points in 1-dimensional Euclidean space and an average curve for squared DTW distance (for any dimension) are not hard on curves of only one point, so our argument here extends the hardness of the problem to include these cases.

3.1 Average-DTW

We again make a truth-table reduction from the SCS instance (S, t). First, we redefine the following points in $\mathbb{R}$:

$$g_A := p_{-2}, \qquad g_B := p_2, \qquad g_a := p_{-1}, \qquad g_b := p_1 \qquad g_0 := p_0.$$

Given a string $s \in S$ over the binary alphabet $\{A, B\}$, we map each character to a curve over $\mathbb{R}$:

$$A \to (g_a g_b)^t g_A (g_b g_a)^t$$
$$B \to (g_b g_a)^t g_B (g_a g_b)^t.$$

The curve $\gamma(s)$ is constructed by concatenating the curves from the previous mapping. The resulting set of curves is again called G, i.e. $G = \{\gamma(s) \mid s \in S\}$.

Now, we add the set $R_{i,j} = \{A_k^i \mid k \in \{1, \dots \alpha\}\} \cup \{B_k^j \mid k \in \{1, \dots \alpha\}\}$ with $A_k^i = A^i$ and $B_k^j = B^j$ for all k and

$$A^i := g_0^\beta (g_A^\beta g_0^\beta)^i$$
$$B^j := g_0^\beta (g_B^\beta g_0^\beta)^j.$$

Here, we take $\alpha = 4t\|S\|/\varepsilon + 1$, $\beta = r$, $r = 4t\|S\| + 2t\alpha$ and $\varepsilon = \frac{1}{8t\|S\|}$. We claim that (S, t) is a YES-instance of SCS if and only if there exists $i, j \in I_t$ such that $(G \cup R_{i,j}, r)$ is a YES-instance of average-DTW.

Before we prove this, we first look at an example of this construction.

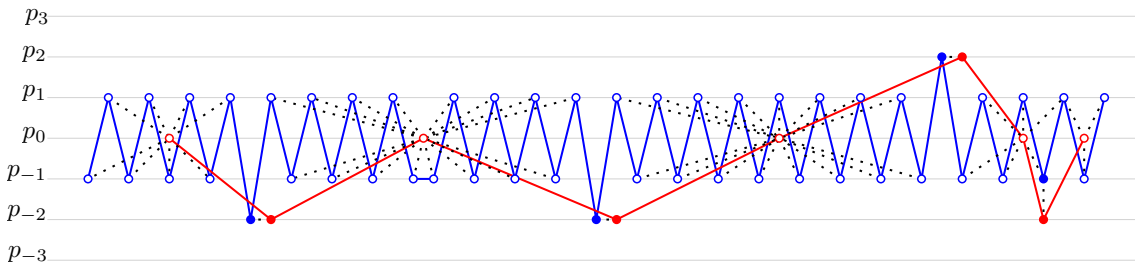

Figure 3.1: A matching between the curve c^* (in red) and the curve $\gamma(s_1)$ (in blue). Matched vertices are connected by dotted lines.

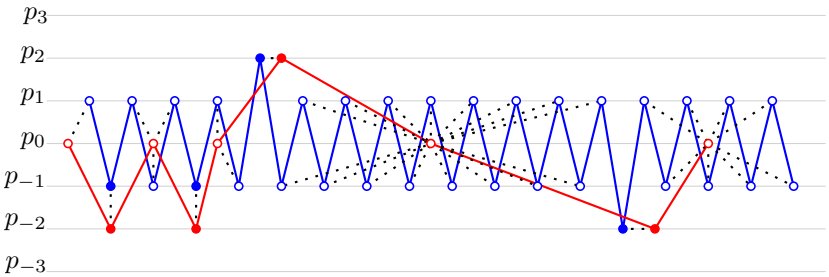

Figure 3.2: A matching between the curve c^* (in red) and the curve $\gamma(s_2)$ (in blue). Matched vertices are connected by dotted lines.

Example 1. Suppose $S = \{s_1, s_2\}$ with $s_1 = AAB$ and $s_2 = BA$. Then the sequence $s^* = AABA$ is a supersequence of S of length 4, so (S, t) with $t = 4$ is a YES-instance of SCS. Since s^* has 3 A characters and 1 B character, we can take $i = 3$ and $j = 1$. The curves in $G \cup R_{i,j}$ are now:

$$\begin{aligned}
\gamma(s_1) &= (g_a g_b)^t g_A (g_b g_a)^t (g_a g_b)^t g_A (g_b g_a)^t (g_b g_a)^t g_B (g_a g_b)^t \\
\gamma(s_2) &= (g_b g_a)^t g_B (g_a g_b)^t (g_a g_b)^t g_A (g_b g_a)^t \\
\left.\begin{aligned} A^i &= g_0^\beta g_A^\beta g_0^\beta g_A^\beta g_0^\beta g_A^\beta g_0^\beta \\ B^j &= g_0^\beta g_B^\beta g_0^\beta \end{aligned}\right\} &\quad \alpha \text{ copies of these curves.}
\end{aligned}$$

We will now show $(G \cup R_{i,j}, r)$, with $r = 4t\|S\| + 2\alpha t$, is a YES-instance of average-DTW. From the supersequence s^*, we construct the curve c^* by interleaving with g_0:

$$c^* = g_0 g_A g_0 g_A g_0 g_B g_0 g_A.$$

For the curves in $\gamma(s) \in G$, we get $\mathrm{DTW}(c^*, \gamma(s)) \leq 4t|s|$ by matching letter gadgets in $\gamma(s)$ with the corresponding letter-gadget in c^* at distance 0 and matching vertices in buffer gadgets in $\gamma(s)$ to vertices in c^* at distance 1. See Figure 3.1 for a matching between c^* and $\gamma(s_1)$, Figure 3.2 for a matching between c^* and $\gamma(s_2)$ and see Figure 3.3 for a more abstract view of the matching that emphasises the cost of the matching.

For the curves in $R_{i,j}$, we get $\mathrm{DTW}(c^*, A^i) \leq 2j = 2$ by matching all g_B vertices from c^* to a single g_0 vertex with distance 2 and matching the remaining vertices in A^i to a corresponding vertex in c^* at distance 0. Similarly, we get $\mathrm{DTW}(c^*, B^j) \leq 2i = 6$ by matching all g_A vertices in c^* to a single g_0 vertex at distance 2 and matching all other vertices in B^j to a corresponding vertex in c^*. See Figure 3.4.

This means that $\sum_{g \in G \cup R_{i,j}} \mathrm{DTW}(g, c^*) = \sum_{s \in S} \mathrm{DTW}(\gamma(s), c^*) + \alpha(\mathrm{DTW}(A^i, c^*) + \mathrm{DTW}(B^j, c^*)) \leq \sum_{s \in S} 4t|s| + \alpha(2i + 2j) = 4t\|S\| + 2\alpha t = r$ and so $(G \cup R_{i,j}, r)$ is indeed a YES-instance of average-DTW.

The following Lemma shows we can do the construction from Example 1 in general.

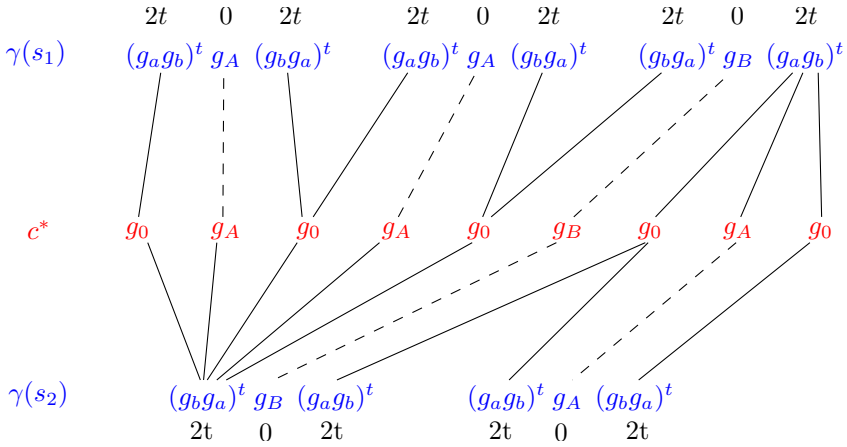

Figure 3.3: An abstract view of the matching between the curve c^* and the curves $\gamma(s_1)$ and $\gamma(s_2)$. Dashed lines indicate a matched pair with cost 0, normal lines indicate matched pairs with cost 1. See Figures 3.1 and 3.2 for a more concrete view of these matchings.

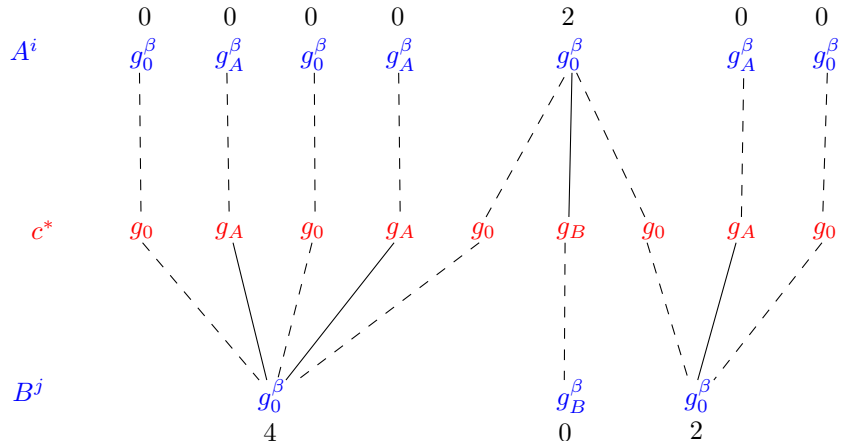

Figure 3.4: A matching between the curve c^* and the curves A^i and B^j. Dashed lines indicate a matched pair with cost 0, normal lines indicate matched pairs with cost 2.

Lemma 12. *If (S, t) is a YES-instance of SCS, then there exist $(i, j) \in I_t$ such that $(G \cup R_{i,j}, r)$ is a YES-instance of average-DTW.*

Proof. If (S, t) is a YES-instance of SCS, then there is a string s^* of length t that is a supersequence of all strings in S. Construct the curve c^* of length $2t + 1$, with for each $i \in \{1, \ldots, 2t + 1\}$,

$$c_i^* := \begin{cases} g_0 & i \text{ is odd} \\ g_A & i \text{ is even and } s^*_{i/2} = A \\ g_B & i \text{ is even and } s^*_{i/2} = B \end{cases} .$$

Let $\gamma(s) \in G$. Since s is a subsequence of s^*, all A/B-gadgets in $\gamma(s)$ can be matched to the corresponding A/B-gadget in c^*. All other A/B-gadgets in c^* can be matched to a vertex in the buffer of $\gamma(s)$ at distance 1. All other buffer vertices in $\gamma(s)$ can be matched to a g_0 vertex in c^*, also at distance 1. So $\mathrm{DTW}(\gamma(s), c^*) \leq 4t|s|$, i.e. the total cost is at most the number of vertices in buffer gadgets in $\gamma(s)$. This means $\sum_{g \in G} \mathrm{DTW}(g, c^*) \leq 4t\|S\|$.

Set i and j to the number of occurrences of A and B in s^*, respectively. Since s^* has length t, we have $(i, j) \in I_t$. Consider the matching between A^i and c^*. We can match each of the i subcurves g_A^r in A^i to each of the i g_A vertices in c^*, with total cost 0. All subcurves g_0 vertices in both curves can be matched to a g_0 vertex in the other curve, so also with cost 0. The remaining j g_B vertices in c^* can be matched to a g_0 vertex, with cost 2. So, $\mathrm{DTW}(A^i, c^*) \leq 2j$. Analogously, $\mathrm{DTW}(B^j, c^*) \leq 2i$ and therefore $\sum_{r \in R_{i,j}} \mathrm{DTW}(r, c^*) = \alpha(\mathrm{DTW}(A^i, c^*) + \mathrm{DTW}(B^j, c^*)) \leq \alpha(2i + 2j) = 2t\alpha$. So, in total, $\sum_{g \in G \cup R_{i,j}} \mathrm{DTW}(g, c^*) \leq 4t\|S\| + 2t\alpha = r$. □

To prove the converse of Lemma 12, we will show that any center curve satisfying the $(G \cup R_{i,j}, r)$ instance must be 'very similar' to the one constructed in Lemma 12 with some supporting lemmas.

Given a true instance of $(G \cup R_{i,j}, r)$ for some i, j, let c^* be a curve such that $\sum_{g \in G \cup R_{i,j}} \mathrm{DTW}(g, c^*) \leq r$. So, there exists an alignment $\mathcal{A}_A$ between c^* and A^i and $\mathcal{A}_B$ between c^* and B^j such that aligning c^* with all α copies of A^i and B^j is able to satisfy the previous inequality.

Definition 1. A vertex p in the curve c^* is called a *A-signal vertex* if

1. p is matched to a vertex g_A in the alignment $\mathcal{A}_A$, and
2. $d(p, g_A) \leq 1$.

Similarly, a vertex p in the curve c^* is called a *B-signal vertex* if

1. p is matched to a vertex g_B in the alignment $\mathcal{A}_B$, and
2. $d(p, g_B) \leq 1$.

A *signal vertex* is either an A-signal vertex or a B-signal vertex. (since there exists no vertex p such that $d(p, g_A) \leq 1$ and $d(p, g_B) \leq 1$, a vertex cannot be both)

Lemma 13. *c^* contains at most t signal vertices.*

Proof. First, note that $\alpha(\mathrm{DTW}(c^*, A^i) + \mathrm{DTW}(c^*, B^j)) \leq r = 2\alpha t + 4t\|s\|$, so $\mathrm{DTW}(c^*, A^i) + \mathrm{DTW}(c^*, B^j) \leq 2t + \frac{4t\|S\|}{\alpha} < 2t + \varepsilon$. Since an A-signal vertex p is matched to some vertex g_A in A^j and some vertex in B^j, p contributes at least $d(p, g_A) + d(p, g_0) \geq 2$ to the total cost of $\mathrm{DTW}(c^*, A^i) + \mathrm{DTW}(c^*, B^j)$. Analogously, a B-signal vertex contributes at least 2 as well. Therefore, if s' is the total number of signal vertices, $2s' \leq \mathrm{DTW}(c^*, A^i) + \mathrm{DTW}(c^*, B^j) < 2t + \varepsilon < 2t + 1$ and so $s' \leq t$. □

Lemma 14. *For all subcurves g_A^β in A^i, there exists an A-signal vertex matched to a vertex in it. For all subcurves g_B^β in B^j, there exists a B-signal vertex matched to a vertex in it.*

Proof. If there is no A-signal vertex matched to some g_A^β subcurve in A^i, then for every vertex p in c^* matched to it, $d(p, g_A) > 1$. This means the cost of matching with this subcurve is strictly more than $\beta = r$, so $r < \mathrm{DTW}(c^*, A^i) \leq r$, a contradiction, which proves the claim for A-signal vertices.

The proof for B-signal vertices is analogous. □

Lemma 15. *Each g_A^β subcurve is matched to an A-signal vertex not matched to any other g_A^β subcurve. Each g_B^β subcurve is matched to an B-signal vertex not matched to any other g_B^β subcurve.*

Proof. Fix a subcurve g_A^β. By Lemma 14, there is an A-signal vertex p matched to this subcurve. Suppose p is matched to some vertex in another g_A^β subcurve. Since the matching between c^* and A^i is monotone, p must be matched to all vertices in some g_0^β subcurve lying between the two g_A^β subcurves. Since $d(p, g_A) \leq 1$ and therefore $d(p, g_0) \geq 1$, matching p with the subcurve costs at least β. This means that $\alpha r = \alpha\beta \leq \alpha\,\mathrm{DTW}(A^i, c^*) \leq r$, a contradiction. So, the A-signal vertex p is not matched to any other g_A^β subcurve.

The proof for B-signal vertices is analogous. □

Lemma 16. *There are exactly t signal vertices.*

Proof. There are i g_A^β subcurves in A^i and j g_B^β subcurves in B^j. By Lemma 15, this means that there are at least i A-signal vertices and at least j B-signal vertices. So, there are at least $i + j = t$ signal vertices. By Lemma 13, there are at most t signal vertices. □

Lemma 17. *Let p be a vertex in c^*. If p is an A-signal vertex, $d(p, g_A) < \varepsilon$. If p is a B-signal vertex, $d(p, g_B) < \varepsilon$. Otherwise, $d(p, g_0) < \varepsilon$.*

Proof. Recall that $\mathrm{DTW}(c^*, A^i) + \mathrm{DTW}(c^*, B^j) < 2t + \varepsilon$. By Lemma 16, the total cost of all signal vertices in $\mathrm{DTW}(c^*, A^i) + \mathrm{DTW}(c^*, B^j)$ is at least $2t$, as each signal vertex costs at least 2. This means that a vertex p in c^* that is not a signal vertex must contribute a cost strictly less than ε. Such a vertex p cannot be matched to a g_A or g_B vertex, as this would cost at least 1 since p is not a signal vertex. Since the curves A^i and B^j only contain the vertices g_A, g_B and g_0 and p must be matched to some vertex on those curves, p must be matched to a g_0 vertex, with $d(p, g_0) < \varepsilon$.

A signal vertex must contribute a cost of less than $2 + \varepsilon$, since all other $t - 1$ signal vertices have a cost of at least $2(t-1)$. Let p be an A-signal vertex. Note that vertices in c^* that are not an A-signal vertex cannot be matched with an g_A vertex, as this has a too large cost. From Lemma 16 and Lemma 15, it follows that the g_A^β subcurve p is matched to is matched to no other A-signal vertex. This means p is matched to all vertices of that g_A^β subcurve. As p is also matched to some vertex of B^j, the cost contributed by p is at least $\beta d(p, g_A) + d(p, g_0)$, so $\beta d(p, g_A) + d(p, g_0) < 2 + \varepsilon$ and therefore $d(p, g_A) < \varepsilon/(\beta - 1) < \varepsilon$ as $d(p, g_A) + d(p, g_0) \geq 2$.

The proof for B-signal vertices is analogous. □

Lemma 18. *Let $\gamma(s)$ be a curve in G. Each vertex g_A in $\gamma(s)$ is matched only to A-signal vertices. Each vertex g_B in $\gamma(s)$ is matched only to B-signal vertices. All signal vertices are matched to at most one g_A or g_B vertex in $\gamma(s)$.*

Proof. Since there are t signal vertices (Lemma 16) that contribute cost of at least 2 to the matchings with A^i and B^j, we have $\mathrm{DTW}(c^*, A^i) + \mathrm{DTW}(c^*, B^j) \geq 2t$. Combining this with the definition of c^* gives $\sum_{g \in G} \mathrm{DTW}(c^*, g) + 2\alpha t \leq \sum_{g \in G} \mathrm{DTW}(c^*, g) + \alpha(\mathrm{DTW}(c^*, A^i) + \mathrm{DTW}(c^*, B^j)) \leq r = 4t\|S\| + 2\alpha t$, so $\sum_{g \in G} \mathrm{DTW}(c^*, g) \leq 4t\|S\|$. By Lemma 17, we have that for any vertex p in c^*, $d(p, g_a) > 1 - \varepsilon$ and $d(p, g_b) > 1 - \varepsilon$. There are $4t\|S\|$ vertices g_a, g_b in G. Matching these vertices to c^* costs at least $(1 - \varepsilon)4t\|S\| = 4t\|S\| - \frac{1}{2}$. Therefore, any additional cost to $\mathrm{DTW}(\gamma(s), c^*)$ (i.e. any cost other than counting $1 - \varepsilon$ per vertex g_a and g_b in $\gamma(s)$) can be at most $\frac{1}{2}$.

This means that a vertex g_A in $\gamma(s)$ can only be matched to an A-signal vertex in c^*, since all other vertices on c^* have a distance of $> \frac{1}{2}$ to g_A (Lemma 17). Analogously, a vertex g_B in $\gamma(s)$ can only be matched to an B-signal vertex.

Finally, note that if an A-signal vertex p would be matched to two distinct vertices g_A in $\gamma(s)$, p must be matched to all vertices in the subcurve $(g_b g_a)^t$ that lies in between the g_A vertices, since the matching is monotone. However, matching p to g_b costs at least $3 - 2\varepsilon$ (Lemma 17), which exceeds the minimal cost for matching g_b of $1 - \varepsilon$ by more than $\frac{1}{2}$. Therefore, p cannot be matched to g_b and so p cannot be matched to two distinct vertices g_A in $\gamma(s)$. Analogously, B-signal vertices cannot be matched to two distinct vertices g_B in $\gamma(s)$. □

Lemma 19. *(S, t) is a YES-instance of SCS.*

Proof. Construct the sequence s^* from c^* by mapping A-signal vertices to A and B-signal vertices to B and concatenating the results (other vertices in c^* are ignored). Since there are t signal vertices (Lemma 16), $|s^*| = t$.

Let $s \in S$ and consider the matching between $\gamma(s)$ and c^*. By Lemma 18 and noting that each vertex in $\gamma(s)$ is matched to at least one vertex in c^*, each g_A vertex in $\gamma(s)$ is matched to an A-signal vertex in c^* that is not matched to any other g_A vertex in $\gamma(s)$ and each g_B vertex in $\gamma(s)$ is matched to an B-signal vertex in c^* that is not matched to any other g_B vertex in $\gamma(s)$. Since the matching is monotone as well, if follows that the sequence of g_A and g_B vertices in $\gamma(s)$ is a subsequence of the sequence of A-signal vertices and B-signal vertices in c^* if we identify g_A vertices with A-signal vertices and g_B vertices with B-signal vertices. Since the sequence s can be mapped to the g_A and g_B vertices in $\gamma(s)$ and the sequence of signal vertices in c^* can be mapped to s^*, it follows that s is a subsequence of s^*. So, s^* is a supersequence with length t of all sequences in S. □

So, we have a polynomial time reduction from SCS to average-DTW and can conclude with the following theorem.

Theorem 20. *The average-DTW problem is NP-hard.*

3.2 Squared average-DTW

The construction for the squared case, where we seek the curve c^* that minimizes $\sum_{i=1}^{n} \mathrm{DTW}(c^*, s_i)^2$ instead of $\sum_{i=1}^{n} \mathrm{DTW}(c^*, s_i)$, is the same as the previous section, except for the following constants: set $r = \sum_{s \in S} (4t|s|)^2 + \alpha((2i)^2 + (2j)^2)$, $\alpha = \frac{1}{\varepsilon} \sum_{s \in S} (4t|s|)^2$, $\beta = r/\varepsilon$ and $\varepsilon = (1 - \frac{1}{2}\sqrt{2})/(\sum_{s \in S} (4t|s|)^2)$. Given an instance (S, t) of SCS, we again reduce to the instances $(G \cup R_{i,j}, r)$ of *squared* average-DTW.

Lemma 21. *If (S, t) is a YES-instance of SCS, then there exists $i, j \in I_t$ such that $(G \cup R_{i,j}, r)$ is a YES-instance of squared average-DTW.*

Proof. Since (S, t) is a YES-instance of SCS, there exist a string s^* of length t that is a supersequence of all strings in S. Construct the curve c of length $2t+1$, with for each $i \in \{1, \ldots, 2t+1\}$:

$$c_i := \begin{cases} g_0 & i \text{ is odd} \\ g_A & i \text{ is even and } s^*_{i/2} = A \\ g_B & i \text{ is even and } s^*_{i/2} = B \end{cases}.$$

Set i and j to the number of occurrences of A and B in s^*, respectively. Since s^* has length t, we have $(i, j) \in I_t$. Analogously to Lemma 12, we have that $\mathrm{DTW}(c, \gamma(s)) \leq 4t|s|$ for all $s \in S$, $\mathrm{DTW}(c, A^i) \leq 2i$ and $\mathrm{DTW}(c, B^j) \leq 2j$. So, $\sum_{g \in G \cup R_{i,j}} \mathrm{DTW}(c, g)^2 \leq \sum_{s \in S} (4t|s|)^2 + \alpha((2i)^2 + (2j)^2) = r$. □

The main idea to prove the converse of Lemma 21 remains the same as in the non-squared case from Section 3.1: we will show that any center curve c^* that attains the required cost must have a matching with the curves in G from which we can construct a short supersequence of S. However, the usage of a sum of squares instead of an ordinary sum of DTW adds some technicalities to the

proof. In particular, proving the inequality $\mathrm{DTW}(c^*, A^i)^2 + \mathrm{DTW}(c^*, B^j)^2 \geq (2i)^2 + (2j)^2$ is less straightforward than the inequality $\mathrm{DTW}(c^*, A^i) + \mathrm{DTW}(c^*, B^j) \geq 2i + 2j$.

Suppose there exists a pair $(i, j) \in I_t$ such that $(G \cup R_{i,j}, r)$ is a YES-instance of squared average-DTW. Then, there exists a curve c^* such that $\sum_{g \in G \cup R_{i,j}} \mathrm{DTW}(c^*, g)^2 \leq r$. We use the signal vertices from definition 1 in the context of this c^* to prove some auxiliary lemmas to conclude with (S, t) is a YES-instance of SCS.

Lemma 22. *An A-signal vertex is matched to exactly one g_A^β subcurve in A^i. An B-signal vertex is matched to exactly one g_B^β subcurve in B^j.*

Proof. We will only show the proof for the A-signal vertices, the case of the B-signal vertices is analogous. By definition, an A-signal vertex is matched to at least one g_A^β subcurve. Suppose an A-signal vertex p is matched to two g_A^β subcurves. Since a DTW matching is monotone, this means p is matched to all vertices in some g_0^β subcurve in between the two g_A^β subcurves. However, as $d(p, g_0) \geq 1$ (since $d(p, g_A) \leq 1$), this means $\mathrm{DTW}(c^*, A^i) \geq \beta d(p, g_0) \geq \beta > r$, a contradiction. Therefore, an A-signal vertex is matched to exactly one g_A^β subcurve. □

Lemma 23. *The contribution of matching signal vertices to the cost of* $\mathrm{DTW}(c^*, A^i)^2 + \mathrm{DTW}(c^*, B^j)^2$ *is at least* $(2i)^2 + (2j)^2$*. If there are at least $i+1$ A-signal vertices or at least $j+1$ B-signal vertices, then the contribution of matching signal vertices to the cost of* $\mathrm{DTW}(c^*, A^i)^2 + \mathrm{DTW}(c^*, B^j)^2$ *is at least* $(2i)^2 + (2j)^2 + 1$.

Proof. By Lemma 22, each g_A^β subcurve in A^i has some A-signal vertices matched to only this subcurve. Since the sum of costs of all vertices in c^* matched to g_A^β is at most r, there must be at least $\beta - r$ vertices from this subcurve that are matched to an A-signal vertex. Let p be a A-signal vertex that has the least distance to g_A. Observe that the A-signal vertices matched to this g_A^β subcurve contribute at least $(\beta - r) \cdot d(p, g_A)$ to $\mathrm{DTW}(c^*, A^i)$ and at least $d(p, g_0)$ to $\mathrm{DTW}(c^*, B^j)$. Since $d(p, g_A) + d(p, g_0) \geq d(g_0, g_A) = 2$ by the triangle inequality, the A-signal vertices contribute at least $(\beta - r) \cdot x^A$ to $\mathrm{DTW}(c^*, A^i)$ and at least $2 - x^A$ to $\mathrm{DTW}(c^*, B^j)$ for some $x^A \in [0, 2]$.

Analogously, the B-signal vertices matched to a single g_B^β subcurve in B^j contribute at least $(\beta - r) \cdot x^B$ to $\mathrm{DTW}(c^*, B^j)$ and at least $2 - x^B$ to $\mathrm{DTW}(c^*, A^i)$ for some $x^B \in [0, 2]$. Since there are i g_A^β subcurves and i g_B^β subcurves, these signal vertices contribute at least $\sum_{k=1}^{i} (\beta - r) x_k^A + \sum_{k=1}^{j} (2 - x_k^B)$ to $\mathrm{DTW}(c^*, A^i)$ and at least $\sum_{k=1}^{j} (\beta - r) x_k^B + \sum_{k=1}^{i} (2 - x_k^A)$ to $\mathrm{DTW}(c^*, B^j)$ for some $x_1^A, \ldots, x_i^A \in [0, 2]$ and $x_1^B, \ldots, x_j^B \in [0, 2]$.

So, the total contribution of signal vertices to the cost of $\mathrm{DTW}(c^*, A^i)^2 + \mathrm{DTW}(c^*, B^j)^2$ is at least

$$\left(\sum_{k=1}^{i} (\beta - r) x_k^A + \sum_{k=1}^{j} (2 - x_k^B) \right)^2 + \left(\sum_{k=1}^{j} (\beta - r) x_k^B + \sum_{k=1}^{i} (2 - x_k^A) \right)^2.$$

Setting $A := \sum_{k=1}^{i} x_k^A$ and $B := \sum_{k=1}^{j} x_k^B$ and $y := \beta - r$, we get

$$\begin{aligned}
&\left(\sum_{k=1}^{i} (\beta - r) x_k^A + \sum_{k=1}^{j} (2 - x_k^B) \right)^2 + \left(\sum_{k=1}^{j} (\beta - r) x_k^B + \sum_{k=1}^{i} (2 - x_k^A) \right)^2 \\
&= \left(2j + (\beta - r) \sum_{k=1}^{i} x_k^A - \sum_{k=1}^{j} x_k^B \right)^2 + \left(2i + (\beta - r) \sum_{k=1}^{j} x_k^B - \sum_{k=1}^{i} x_k^A \right)^2 \\
&= (2j + yA - B)^2 + (2i + yB - A)^2 \\
&= (2i)^2 + (2j)^2 + A^2 + B^2 \\
&\quad + 4A(yj - i) + 4B(yi - j) \\
&\quad + (Ay)^2 + (By)^2 - 4ABy \\
&\geq (2i)^2 + (2j)^2,
\end{aligned}$$

where $A \in [0, 2i]$ and $B \in [0, 2j]$. To show the inequality above holds, it is sufficient to show that the terms other than $(2i)^2 + (2j)^2$ are greater than 0. $A^2 + B^2 \geq 0$ is obvious. $4A(yj - i) + 4B(yi - j) \geq 0$, since $A, B \geq 0$ and $y = \beta - r = (\frac{1}{\varepsilon} - 1)r \geq \max\{i, j\}$. Finally, $(Ay)^2 + (By)^2 - 4ABy \geq (Ay)^2 + (By)^2 - 2ABy^2 = (Ay - By)^2 \geq 0$, since $y \geq 2$.

If there are at least $i+1$ A-signal vertices, then there is a g_A^β subcurve that is matched with at least two A-signal vertices. This means that there is an additional matching of an A-signal vertex with some vertex in B^j, which costs at least 1. So, in this case, the signal vertices contribute at least

$$\left(\sum_{k=1}^{i}(\beta - r)x_k^A + \sum_{k=1}^{j}(2 - x_k^B)\right)^2 + \left(\sum_{k=1}^{j}(\beta - r)x_k^B + \sum_{k=1}^{i}(2 - x_k^A) + 1\right)^2 \geq (2i)^2 + (2j)^2 + 1$$

to the cost of $\mathrm{DTW}(c^*, A^i)^2 + \mathrm{DTW}(c^*, B^j)^2$, using the inequality above. Analogously, if there are at least $j+1$ B-signal vertices, then the signal vertices contribute at least $(2i)^2 + (2j)^2 + 1$ to the cost of $\mathrm{DTW}(c^*, A^i)^2 + \mathrm{DTW}(c^*, B^j)^2$. □

Lemma 24. *There are exactly i A-signal vertices and exactly j B-signal vertices.*

Proof. We will only provide the proof for A-signal vertices, as the proof for B-signal vertices is analogous. By Lemma 22, there are at least i A-signal vertices.

Suppose there are at least $i + 1$ A-signal vertices. Then, Lemma 23 implies

$$\mathrm{DTW}(c^*, A^i)^2 + \mathrm{DTW}(c^*, B^j)^2 \geq (2i)^2 + (2j)^2 + 1.$$

Since $\alpha(\mathrm{DTW}(c^*, A^i)^2 + \mathrm{DTW}(c^*, B^j)^2) \leq r = \sum_{s \in S}(4t|s|)^2 + \alpha((2i)^2 + (2j)^2)$, we have

$$\mathrm{DTW}(c^*, A^i)^2 + \mathrm{DTW}(c^*, B^j)^2 \leq (2i)^2 + (2j)^2 + \frac{1}{\alpha}\sum_{s \in S}(4t|s|)^2 < (2i)^2 + (2j)^2 + \varepsilon.$$

Combined with the inequality above, we get $1 < \varepsilon$, a contradiction, which means there are at most i A-signal vertices. □

Lemma 25. *Let p be a vertex in c^*. If p is an A-signal vertex, $d(p, g_A) < \varepsilon$. If p is an B-signal vertex, $d(p, g_B) < \varepsilon$. Otherwise, $d(p, g_0) < \varepsilon$.*

Proof. Let p be a vertex in c^* that is not a signal vertex. Since $\mathrm{DTW}(c^*, A^i)^2 + \mathrm{DTW}(c^*, B^j)^2 < (2i)^2 + (2j)^2 + \varepsilon$ and matching signal vertices with A^i and B^j contributes at least $(2i)^2 + (2j)^2$ to $\mathrm{DTW}(c^*, A^i)^2 + \mathrm{DTW}(c^*, B^j)^2$ (see Lemma 23), the cost of matching p with A^i and B^j is at most ε. This means that $d(p, q_A) + d(p, q_B) < \varepsilon$ for some vertex q_A in A^i and q_B in B^j, since p must be matched to at least one vertex from A^i and B^j. By the triangle inequality, it follows that $d(q_A, q_B) < \varepsilon$, which means q_A and q_B must be vertices at g_0. So, $d(p, g_0) < \varepsilon$ and p can only be matched to vertices at g_0 in A^i and B^j.

Let p be an A-signal vertex. By the reasoning above, no non-signal vertices are matched with the g_A^β subcurve p is matched to. If there would be a B-signal vertex that is matched to this g_A^β subcurve, then this increases the cost of matching the B-signal vertex with A^i by 2 compared to matching it with a vertex at g_0 and will contradict the upper bound of $(2i)^2 + (2j)^2 + \varepsilon$ for $\mathrm{DTW}(c^*, A^i)^2 + \mathrm{DTW}(c^*, B^j)^2$. So, only A-signal vertices are matched to a vertex in a g_A^β subcurve. Additionally p is the only A-signal vertex matched to the g_A^β subcurve it is matched to by Lemma 24, so p is matched to all β vertices in the g_A^β subcurve. This means that $(d(p, g_A)\beta)^2 \leq r$, so $d(p, g_A) \leq \frac{r}{\beta^2} = \frac{\varepsilon}{\sqrt{r}} < \varepsilon$.

The proof for the B-signal vertices is analogous. □

Lemma 26. *Let $\gamma(s)$ be a curve in G. Each vertex g_A in $\gamma(s)$ is matched only to A-signal vertices. Each vertex g_B in $\gamma(s)$ is matched only to B-signal vertices. All signal vertices are matched to at most one g_A or g_B vertex in $\gamma(s)$.*

Proof. From the definition of c^* and using Lemma 23, we get $\sum_{s\in S}\mathrm{DTW}(c^*,\gamma(s))^2 \leq r - \alpha(\mathrm{DTW}(c^*,A^i)^2 + \mathrm{DTW}(c^*,B^j)^2) \leq r - \alpha((2i)^2+(2j)^2) = \sum_{s\in S}(4t|s|)^2$. From Lemma 25, it follows that for any vertex p in c^*, $d(p,g_a) > 1-\varepsilon$ and $d(p,g_b) > 1-\varepsilon$. For any $s \in S$, the curve $\gamma(s)$ contains exactly $4t|s|$ vertices at g_a or g_b, so matching these vertices with c^* contributes at least $4t|s|(1-\varepsilon)$ to $\mathrm{DTW}(c^*,\gamma(s))$. Define $X(s') := \mathrm{DTW}(c^*,\gamma(s')) - 4t|s|(1-\varepsilon)$ for all $s' \in S$, i.e. $X(s')$ is the cost of matching c^* with $\gamma(s')$ that has not yet been counted. Therefore, for any $s' \in S$,

$$\begin{aligned} X(s') + (1-\varepsilon)^2\sum_{s\in S}(4t|s|)^2 &< \sum_{s\in S}(X(s) + 4t|s|(1-\varepsilon))^2 \\ &= \sum_{s\in S}\mathrm{DTW}(c^*,\gamma(s))^2 \\ &\leq \sum_{s\in S}(4t|s|)^2, \end{aligned}$$

so $X(s') < (1-(1-\varepsilon)^2)\sum_{s\in S}(4t|s|)^2 = \frac{1}{2}$.

The remainder of the proof is analogous to Lemma 18. □

Lemma 27. *(S,t) is a YES-instance of SCS.*

Proof. This proof is follows from Lemma 24 (note that $i+j=t$) and Lemma 26 analogously to Lemma 19 following from Lemma 16 and Lemma 18. □

So, we have a polynomial time reduction from SCS to average-DTW and we can conclude with the following theorem.

Theorem 28. *The squared average-DTW problem is NP-hard.*

3.3 Multiple sequence alignment and clustering under the DTW distance

The paper by Petitjean and Garcarski [24] introduces a heuristic algorithm for finding an average curve under the squared DTW distance. Their method is based on connecting the notion of the average curve for the DTW distance with the multiple sequence alignment (MSA) problem. They suggest that the intractability of the problem of finding the average DTW distance is related to the intractability of finding an optimal multiple sequence alignment under SP-score (Sum of Pairs score). This led us to investigate whether we can formally use this connection to show that finding an average curve under the (squared) DTW distance is NP-hard based on the NP-hardness proof of an MSA problem in [30]. Unfortunately, this approach did not lead to a simple NP-hardness proof.

In this section, we consider Problem 1: a variant of the average curve problem under DTW that is more closely related to the MSA problem. We show that Problem 1 is NP-hard by a relatively simple reduction from the shortest common supersequence problem. We then use some examples to show that there are obstacles to a similar reduction for finding an average sequence under DTW.

An *alignment of a set of sequences* $\mathcal{S}$ over Σ is a set of sequences $\mathcal{A} = \{a_1,\ldots,a_n\}$ of equal length over $\Sigma\cup\{\Delta\}$, such that for all i, the subsequence of a_i obtained by removing the Δ symbol is equal to s_i. $\Delta \notin \Sigma$ represents a *gap* in the alignment. An alignment is often presented as a matrix where the i-th row corresponds to the sequence a_i. $\|\mathcal{S}\|$ denotes the total length of all sequences in $\mathcal{S}$.

Problem 1 (Alignment consensus sequence)**.** Given are a set $\mathcal{S} = \{s_1,\ldots,s_n\}$ of n sequences with length at most m, where the elements of those sequences are from some set Σ equipped with a distance metric $\delta : \Sigma\cup\{\Delta\}\times\Sigma\cup\{\Delta\} \to \mathbb{R}^+$. Then, the Alignment-consensus problem is

the problem of finding a *consensus sequence* c^* such that there exists an alignment $\mathcal{A}$ of $\mathcal{S}$ such that the SP-Score $\sum_{i=1}^{n}\sum_{j=1}^{m}\delta(a_i[j], c^*[j])$ is minimized.

Note that the SP-score cost of a consensus sequence c^* in a sequence alignment $\mathcal{A}$ is the same as the cost of this sequence interpreted as an average curve under the DTW distance, if c^* is aligned in the pairwise DTW distance as in $\mathcal{A}$.

3.3.1 A reduction for the alignment consensus problem

We can show that the decision version of Problem 1, which is to decide if an instance with SP-score at most t exists, is NP-hard.

Lemma 29. *The decision version of* Alignment-consensus *is NP-hard.*

Proof. We use a reduction from the Shortest Common Supersequence (SCS) problem, which is NP-hard [25]. The reduction is as follows: Given an instance $(\mathcal{S}, r)$ of SCS, we reduce to the instance $(\mathcal{S}, t)$ of Alignment-consensus, with $t := n \cdot r - \|\mathcal{S}\|$ and distance δ such that $\delta(0,1) = +\infty$, $d(x, \Delta) = 1$ for all $x \in \Sigma$, $d(y, y) = 0$ for all $y \in \Sigma \cup \{\Delta\}$ and δ is symmetric. We claim that $(\mathcal{S}, r)$ is a YES-instance of SCS if and only if $(\mathcal{S}, t)$ is a YES-instance of Alignment-consensus.

If: Suppose $(\mathcal{S}, r)$ is a YES-instance of SCS. Then, there exists a sequence s^* of length at most r such that s_i is a subsequence of s^* for all i. We create an alignment $\mathcal{A}$ of $\mathcal{S}$ as follows: for all i, take a_i such that $a_i[j] = s^*[j]$ if j is an index of the subsequence in s^* corresponding to s_i and $a_i[j] = \Delta$, otherwise. In other words, we fill in gaps to match the entries of s_i to the corresponding index in s^*. Note that $\mathcal{A}$ is indeed an alignment of $\mathcal{S}$. Take s^* as the consensus sequence c^*. For all i, note that $\sum_{j=1}^{m}\delta(a_i[j], c^*[j]) = |s^*| - |s_i|$, as there are $|s^*| - |s_i|$ matches of a Δ in a_i with a character from Σ in s^* and all other matches are with the same character. So $\sum_{i=1}^{n}\sum_{j=1}^{m}\delta(a_i[j], c^*[j]) = \sum_{i=1}^{n}|s^*| - |s_i| \leq n \cdot r - \|\mathcal{S}\| = t$ and $(\mathcal{S}, t)$ is therefore a YES-instance of Alignment-consensus.

Only if: Suppose $(\mathcal{S}, t)$ is a YES-instance of Alignment-consensus, then there exist a consensus sequence c^* and alignment $\mathcal{A}$ such that $\sum_{i=1}^{n}\sum_{j=1}^{m}\delta(a_i[j], c^*[j]) \leq t < \infty$. This means that $\delta(a_i[j], c^*[j]) < \infty$ for all i, j and hence no 0 is matched with a 1. This means that s_i, which is equal to the sequence obtained after removing all gaps from a_i, is a subsequence of c^*. Similar to above, this means that $\sum_{j=1}^{m}\delta(a_i[j], c^*[j]) = |c^*| - |s_i|$ for all i. Therefore, we get $n \cdot |c^*| - \|\mathcal{S}\| = \sum_{i=1}^{n}|c^*| - |s_i| = \sum_{i=1}^{n}\sum_{j=1}^{m}\delta(a_i[j], c^*[j]) \leq t = n \cdot r - \|\mathcal{S}\|$. Therefore, we get $|c^*| \leq r$ and therefore $(\mathcal{S}, r)$ is a YES-instance of SCS. □

Note that for simplicity, we have chosen a non-Euclidean distance in this reduction. When we set $d(1,0) = 2$, we do get an Euclidean distance.

3.3.2 Challenges for a more useful reduction

Applying the same strategy to finding an average curve under DTW directly fails, as an alignment does not count the length of the supersequence:

Example 2. Consider $\mathcal{S} = \{s_1, s_2, s_3\}$ with $s_1 = \langle 1, 0, 1, 1\rangle$, $s_2 = \langle 0, 1, 1, 1\rangle$ and $s_3 = \langle 1, 0, 0, 1\rangle$. Figure 3.5 shows optimal pairwise alignments these sequences have with a shortest common supersequence c^*. Note that if we interpret the duplicates as 'gaps', we get a gap cost that is dependent of the previous row, since we actually copy the row. This difference in gap costs is the main difference between DTW and alignments. Since we cannot use the notion of gaps similarly to that of an alignment, applying the strategy of the previous reduction directly fails.

One idea to simulate the 'gap insertion' behaviour from sequence alignments within DTW is to interleave all sequences with a gap: given a sequence s of length m, construct t of length $2m+1$ such that $t[i] = \begin{cases} s[i/2] & \text{if } i \text{ is even} \\ \Delta & \text{otherwise} \end{cases}$ (indices start at 1). In this construction, we can create a DTW alignment similar to an alignment from a supersequence. However, there can be a strictly better average sequence that is not a shortest supersequence:

$s_1:$	1	0	$\underline{0}$	1	1	$\underline{1}$
$s_2:$	0	$\underline{0}$	$\underline{0}$	1	1	1
$s_3:$	1	0	0	1	$\underline{1}$	$\underline{1}$
$c^*:$	1	0	0	1	1	1

Figure 3.5: The sequences in $\mathcal{S}$ of Example 2 and a shortest common super-sequence c^*. The underlined characters are 'duplicates' of the previous character in an optimal alignment with c^*.

Example 3. Take again $\mathcal{S}$ from Example 2. Construct t_1, t_2, t_3 by interleaving with gaps, as described above. Construct t^* from interleaving the supersequence c^* with gaps. The following matrix shows optimal warps (given t^*) such that the total cost of c^* here is 6, as there are 6 gaps. (gaps are are denoted by dashes.)

$t_1:$	-	1	0	-	-	-	1	-	1	-	-	-
$t_2:$	-	-	0	-	-	-	1	-	1	-	1	-
$t_3:$	-	1	0	-	0	-	1	-	-	-	-	-
$t^*:$	-	1	0	-	0	-	1	-	1	-	1	-

However, the sequence $t' = \langle \Delta, 1, \Delta, 1, \Delta, 1, \Delta \rangle$ achieves a total cost of 5:

$t_1:$	-	1	-	0	-	1	-	1	-
$t':$	-	1	-	-	-	1	-	1	-

, so $\mathrm{DTW}(t_1, t') \leq 1$.

$t_2:$	-	0	-	1	-	1	-	1	-
$t':$	-	-	-	1	-	1	-	1	-

, so $\mathrm{DTW}(t_2, t') \leq 1$.

$t_3:$	-	1	-	0	-	0	-	1	-	-	-
$t':$	-	1	-	-	-	-	-	1	-	1	-

, so $\mathrm{DTW}(t_3, t') \leq 3$.

Clearly, t' doesn't correspond to a supersequence of $\mathcal{S}$. One problem is that, unlike the alignment consensus sequence, the average sequence under DTW is 'symmetric' in the sense that it can either be larger or smaller than the input sequences, while the consensus sequence must be larger than the sequences from $\mathcal{S}$ by definition of alignment.

Another problem is that for the alignment consensus problem, the consensus sequence must have a single alignment that is consistent with all sequences from $\mathcal{S}$ simultaneously. This is not required for the average sequence, as we need only pairwise DTW alignments.

Chapter 4

Algorithms for (k, ℓ)-center and $(1, \ell)$-median

4.1 Approximating $(1, 2)$-center for the Fréchet distance

In this section, we will give a polynomial $1 + \varepsilon$ approximation algorithm for the $(1, 2)$-center clustering problem under the (discrete or continuous) Fréchet distance d_F, i.e. the problem of finding the the minimum cost segment as a center for a given set of curves. More precisely, given n curves $Z_1, Z_2, \dots Z_n$ with vertices in $\mathbb{R}^d$ equipped with the Euclidean norm $\|\cdot\|$, find a pair of points $p^*, q^* \in \mathbb{R}^d$ that minimizes $\max_{i=1}^n \mathrm{d}_F(Z_i, p^*q^*)$, where p^*q^* denotes the segment connecting p^* and q^*.

Before we give the algorithm, we first derive some useful inequalities. Denote the first vertex of Z_i by u_i and the last vertex by v_i and define $u^* = \frac{1}{n}\sum_{i=1}^n u_i$ and $v^* = \frac{1}{n}\sum_{i=1}^n v_i$.

Lemma 30. *For any segment pq,*

(i) $\max_{i=1}^n \mathrm{d}_F(Z_i, pq) \geq \mathrm{d}_F(u^*v^*, pq)$*; and*

(ii) $\max_{i=1}^n \mathrm{d}_F(Z_i, pq) \geq \frac{1}{2}\max_{i=1}^n \mathrm{d}_F(Z_i, u^*v^*)$*;*

Proof. (i) First, note that the matching between curves for the Fréchet distance must match the start and end vertices to each-other, so $\mathrm{d}_F(Z_i, pq) \geq \max(\|u_i - p\|, \|v_i - q\|)$ for all i. So, $\max_{i=1}^n \mathrm{d}_F(Z_i, pq) \geq \max_{i=1}^n \max(\|u_i - p\|, \|v_i - q\|) \geq \max(\max_{i=1}^n \|u_i - p\|, \max_{i=1}^n \|v_i - p\|)$. Note that $\max_{i=1}^n \|u_i - p\| \geq \|u^* - p\|$, since

$$\|u^* - p\| = \|\frac{1}{n}\sum_{i=1}^n (u_i - p)\| \leq \frac{1}{n}\sum_{i=1}^n \|u_i - p\| \leq \frac{1}{n}\sum_{i=1}^n \max_{j=1}^n \|u_j - p\| = \max_{i=1}^n \|u_i - p\|,$$

using the triangle inequality for $\|\cdot\|$. Analogously, $\max_{i=1}^n \|v_i - q\| \geq \|v^* - q\|$. This means $\max_{i=1}^n \mathrm{d}_F(Z_i, pq) \geq \max(\max_{i=1}^n \|u_i - p\|, \max_{i=1}^n \|v_i - p\|) \geq \max(\|u^* - p\|, \|v^* - q\|) = \mathrm{d}_F(u^*v^*, pq)$.

(ii) Using the triangle inequality over d_F and (i), we get

$$\max_{i=1}^n \mathrm{d}_F(Z_i, u^*v^*) \leq \max_{i=1}^n \mathrm{d}_F(Z_i, pq) + \mathrm{d}_F(pq, u^*v^*) \leq 2\max_{i=1}^n \mathrm{d}_F(Z_i, pq). \qquad \square$$

This lemma shows that the segment u^*v^* already gives a 2-approximation to the $(1, 2)$-center problem. To obtain an $1 + \varepsilon$-approximation, we construct grids around the points u^* and v^* as in Chapter 4.2.2 from [15] and exhaustively test all segments with endpoints in these grids. In particular, we use the following lemma:

Lemma 31. *Given a point $u \in \mathbb{R}^d$ and parameters $0 < \rho \leq 1$ and $\alpha > 0$. Then we can compute a grid $G(u)$ centred at u of $(\lceil \frac{\sqrt{d}}{2\rho} \rceil + 1)^d$ vertices such that for each point p with $\|p - u\| \leq \alpha$, there exists a vertex p' in $G(u)$ such that $\|p' - p\| \leq \rho\alpha$.*

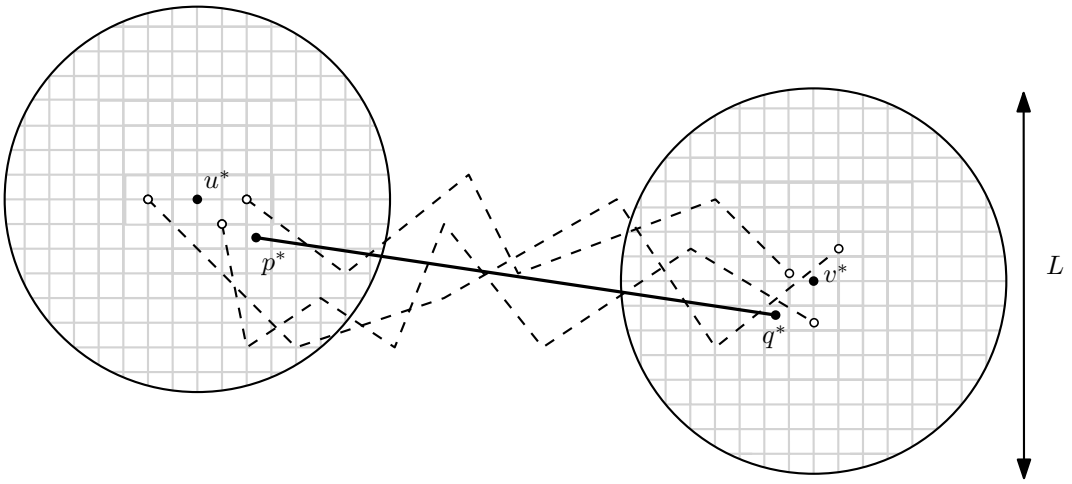

Figure 4.1: The grids around the points u^* and v^* contain the the vertices of an optimal segment center p^*q^* under the Fréchet distance for the input curves (dashed).

Proof. We construct an axis-parallel d-dimensional hypercube centred at u and of side length α. We then divide this hypercube into smaller hypercubes of side length $\frac{2\rho}{\sqrt{d}}\alpha$. The grid $G(u)$ is the set of all vertices of these smaller hypercubes. Since each side of the hypercube is divided in $\lceil \frac{\sqrt{d}}{2\rho} \rceil$ portions, the total number of vertices is $(\lceil \frac{\sqrt{d}}{2\rho} \rceil + 1)^d$.

Let p be a point such that $\|p - u\| \leq \alpha$. Then p lies in the interior of C, one of the small hypercubes. Let p' be the vertex of C that is closest to p. Then, we have

$$\|p - p'\| \leq \frac{\operatorname{diam}(C)}{2} \leq \frac{\sqrt{d}}{2} \cdot \frac{2\rho}{\sqrt{d}}\alpha = \rho\alpha. \qquad \square$$

Algorithm: Given $Z_1, \ldots, Z_n$ with at most m vertices each, first compute the points u^* and v^* as defined above. Then, compute $L = \max_{i=1}^n \mathrm{d}_F(Z_i, u^*v^*)$ and compute the grids $G(u^*)$ and $G(v^*)$ with parameters $\rho = \varepsilon/2$ and $\alpha = L$, as in Lemma 31 (See Figure 4.1). Compute $\max_{i=1}^n \mathrm{d}_F(Z_i, pq)$ for every pair $(p, q) \in G(u^*) \times G(v^*) =: D$ and return a pair p', q' such that $\max_{i=1}^n \mathrm{d}_F(Z_i, p'q') = \min_{(p,q)\in D} \max_{i=1}^n \mathrm{d}_F(Z_i, pq)$.

Theorem 32. *Given n input curves in $\mathbb{R}^d$, each of complexity at most m, and some $0 < \varepsilon \leq 1$, we can compute an $(1+\varepsilon)$-approximation to the $(1, 2)$-center problem for the continuous Fréchet distance in $O(C^2 nm \log m)$ time, with $C = (\frac{\varepsilon}{\sqrt{d}})^{-d}$.*

Proof. Running time: Computing u^* and v^* can be done in $O(n)$ time. We can compute the Fréchet distance between a segment and a curve with m vertices in $O(m \log m)$ time and compute this n times for all $O(((\lfloor \frac{\varepsilon}{\sqrt{d}} \rfloor + 1)^{-d})^2) = O(C^2)$ pairs in the grid. In total, the running time is $O(C^2 nm \log m)$.

Approximation factor: Call the value returned by the algorithm Δ. Let p^*, q^* be a pair that optimizes $\min_{(p,q)\in\mathbb{R}^d} \max_{i=1}^n \mathrm{d}_F(Z_i, pq)$. We claim that $\Delta \leq (1+\varepsilon) \max_{i=1}^n \mathrm{d}_F(p^*q^*, Z_i)$. Let $R = \max(\|p^* - u^*\|, \|q^* - v^*\|)$. We consider two cases: either (i) $R > L$, or (ii) $R \leq L$.

In case (i), $\mathrm{d}_F(p^*q^*, u^*v^*) = R > L = \max_{i=1}^n \mathrm{d}_F(Z_i, u^*v^*) \geq \Delta$ so with Lemma 30(i), we get $\Delta \leq \max_{i=1}^n \mathrm{d}_F(Z_i, p^*q^*)$.

In case (ii), both $\|p^* - u^*\|$ and $\|q^* - v^*\|$ are less than L, so by Lemma 31, there exists points p', q' in the grids such that $\|p^* - p'\| \leq (\varepsilon/2)L$ and $\|q^* - q'\| \leq (\varepsilon/2)L$. So,

$$\mathrm{d}_F(p^*q^*, p'q') = \max(\|p^* - q'\|, \|q^* - q'\|) \leq (\varepsilon/2)L,$$

and using Lemma 30 (ii), we get $\mathrm{d}_F(p^*q^*, p'q') \leq \varepsilon \max_{i=1}^n \mathrm{d}_F(Z_i, p^*q^*)$. Finally, using the triangle inequality, we get

$$\Delta \leq \max_{i=1}^n \mathrm{d}_F(Z_i, p'q') \leq \max_{i=1}^n \mathrm{d}_F(Z_i, p^*q^*) + \mathrm{d}_F(p^*q^*, p'q') \leq (1+\varepsilon) \max_{i=1}^n \mathrm{d}_F(Z_i, p^*q^*). \qquad \square$$

4.2 Approximating (k,ℓ)-center for the discrete Fréchet distance

In this section, we will use the idea from Section 4.1 of creating an approximation by constructing grids as in Lemma 31 around suitable points to get a $(1+\varepsilon)$-approximation for any $\varepsilon > 0$ to the (k,l)-center problem for the discrete Fréchet distance in $O(mn\log(m))$ for any fixed $k, l \in \mathbb{N}$.

Algorithm: Given input curves $Z_1, Z_2, \ldots, Z_n$ of complexity at most m each, we first compute a set $\mathcal{C} := C_1, C_2, \ldots, C_k$ of k curves that forms a 3-approximation for the (k,ℓ)-center problem on the input curves, using the algorithm from the paper [12]: First select an arbitrary input curve Z, compute an ℓ-simplification $\bar{Z}$ of that curve, and add the curve to $\mathcal{C}$. Then, repeatedly select an input curve Z that has the largest discrete Fréchet distance to all curves in $\mathcal{C}$, compute an ℓ-simplification $\bar{Z}$ of that curve, and it to $\mathcal{C}$ until there are k curves in $\mathcal{C}$. See the paper [12] for a more detailed description. Call the cost of the 3-approximation Δ (i.e. the maximum discrete Fréchet distance from an input curve to the nearest curve in $\mathcal{C}$) and let X be the set of vertices of the curves in $\mathcal{C}$.

For each vertex $x \in X$, compute a grid $G(x)$ with the properties from Lemma 31 with parameters $\rho = \varepsilon/6$ and $\alpha = 2\Delta$. Call the union of all grid points U, i.e. $U = \bigcup_{x\in X} G(x)$. Iterate over all sets $\mathcal{C}'$ of k curves of complexity l with vertices in U (including repetitions) and return set of curves with the minimum cost.

This algorithm yields an $(1+\varepsilon)$-approximation.

Theorem 33. *Given n input curves in $\mathbb{R}^d$, each of complexity at most m, and positive integers k, ℓ and some $0 < \varepsilon \leq 1$, we can compute an $(1+\varepsilon)$-approximation to the (k,ℓ)-center problem for the discrete Fréchet distance in $O\left(((Ck\ell)^{k\ell} + \log(\ell+m)) \cdot k\ell \cdot mn\right)$ time, with $C = \left(\frac{6\sqrt{d}}{\varepsilon}\right)^d$.*

Proof. Running time: Computing the initial 3-approximation can be done in $O(kn \cdot \ell m \log(\ell + m))$ time [12]. Note that $|U| = \sum_{x\in X} |G(x)| = |X| \cdot \left(\frac{6\sqrt{d}}{\varepsilon}\right)^d = C \cdot k\ell$. We iterate over $|U|^{k\ell}$ candidate solutions and for each of the n input curves, compute the discrete Fréchet distance of that curve and all k curves in the candidate solution in $O(\ell m)$. So, in total, the algorithm takes $O\left(((Ck\ell)^{k\ell} + \log(\ell+m)) \cdot k\ell \cdot mn\right)$ time.

Approximation factor: Let $\mathcal{C}^*$ be an optimal solution to the (k,ℓ)-center problem that achieves the cost O and let p be a vertex of a curve in $\mathcal{C}^*$. We can assume that each curve in $\mathcal{C}^*$ has at least one input curve in its cluster, since we can replace those an unused curves in $\mathcal{C}^*$ with an input curve without increasing the cost. Since p lies on a curve with discrete Fréchet distance at most O to all input curves in its cluster, there exists a vertex q on an input curve such that $\|p - q\| \leq O \leq \Delta$. Since the vertices in X lie on curves with discrete Fréchet distance at most Δ to the input curves, there exists a vertex $x \in X$ such that $\|q - x\| \leq \Delta$. So, using the triangle inequality, $\|p - x\| \leq \|p - q\| + \|q - x\| \leq 2\Delta$. This means that p lies within a cell of the grid $G(x)$, so by Lemma 31, there exists a grid-point p' such that $\|p - p'\| \leq (\varepsilon/3)\Delta \leq \varepsilon O$, where the latter inequality holds since Δ is the cost of a 3-approximation.

Therefore, there exists a set $\mathcal{C}'$ of k curves of complexity ℓ with vertices in U such that for any $C \in \mathcal{C}^*$, there is a curve $C' \in \mathcal{C}'$ with $\mathrm{d}_{dF}(C, C') \leq \varepsilon O$. This means that for any input curve Z that lies in the cluster with center $C \in \mathcal{C}^*$, there exists a $C' \in \mathcal{C}^*$ such that $\mathrm{d}_{dF}(Z, C') \leq \mathrm{d}_{dF}(Z, C) + \mathrm{d}_{dF}(C, C') \leq O + \varepsilon O = (1+\varepsilon)O$. Since the algorithm has considered the set $\mathcal{C}'$ as a candidate solution, the algorithm indeed gives an $(1+\varepsilon)$-approximation. $\square$

4.3 Computing (k, ℓ)-center for discrete Fréchet distance in 2D

In this section, we give an algorithm that solves the decision version of the (k, ℓ)-center problem for the discrete Fréchet distance in 2D in polynomial time for fixed ℓ. So, we consider the following problem: Given a set of n curves G in the plane with at most m vertices each and a positive real number r, does there exist a set of k center curves $\mathcal{C}$ with at most ℓ vertices each such that $\min_{c \in \mathcal{C}} \mathrm{d}_{dF}(c, g) \leq r$ for all $g \in G$?

The main idea of the algorithm is based on the following observation: $\min_{c \in \mathcal{C}} \mathrm{d}_{dF}(c, g) \leq r$ for all $g \in G$ if and only if each vertex p of a curve in $\mathcal{C}$ lies in the intersection of the disks of radius r around all vertices q from curves in G that p is matched with. Furthermore, it does not matter where the vertex p lies within the intersection region. This means we can select a vertex for each maximal overlapping region (i.e. each region such that the set of disks intersecting the region is not contained in another region) and exhaustively test all sets with k curves of ℓ vertices that can be constructed by using only the selected vertices to determine if there exists a set of curves $\mathcal{C}$ such that $\min_{c \in \mathcal{C}} \mathrm{d}_{dF}(c, g) \leq r$ for all $g \in G$.

Algorithm: Let C be the set of all vertices in G. Construct the planar graph $\mathcal{G} = (V, E)$, where V is the set of all intersection points boundaries of disks centered around a vertex in C with radius r and E is the set of arcs on the boundary of those disks determined by two intersection points. This graph has $O((nm)^2)$ vertices and arcs and can be computed in $O((nm)^2)$ time [14]. (We add a vertex at the top and bottom of each disk, so that each disk boundary consists of at least two arcs. See [14] for more details on this construction.) We will store the vertices to base the center curve on in the list S. First, for each pair of disks, if (the boundary of) the disks intersect in exactly one point, then add this point to S and mark the vertex in $\mathcal{G}$. Next, we will find the maximal intersection regions of $\mathcal{G}$ that are larger than a single point. Call an arc on the boundary of a face convex for that face if the disk the arc lies on contains the face. Note that a face f in $\mathcal{G}$ is a maximal intersection region if and only if the arcs on the boundary of f are convex with respect to f and the boundary contains no marked vertices.

Then, for each unmarked arc a in E, traverse the boundary of the face f for which a is convex and mark all arcs for f on this boundary. If all arcs on the boundary of f are convex and the boundary contains no marked vertices, then add a vertex in f to S. Then repeat this until all arcs in E are marked.

Finally, for each set of k curves $\mathcal{C}$ of length ℓ with vertices in S, test whether $\min_{c \in \mathcal{C}} \mathrm{d}_{dF}(c, g) \leq r$ for all $g \in G$. If we have such a set of curves, return yes, otherwise return no.

Running time Since $|C| \leq n$, constructing the graph $\mathcal{G}$ takes $O((nm)^2)$ time. Finding all disks that intersect in exactly one point and traversing all $O((nm)^2)$ edges can be done in $O((nm)^2)$ time. The number of intersection regions in $\mathcal{G}$ is at most $(nm)^2 - (nm) + 2$. This can be shown by induction. Since the boundary of each face consists of at least two unique sides of arcs, there are $O((nm)^2)$ faces in G, so $|S| \leq (nm)^2$. The discrete Fréchet distance between a curve $g \in G$ and $c \in \mathcal{C}$ can be computed in $O(m\ell)$. So, testing all $|S|^{k\ell}$ sets of center curves takes $O(|S|^{k\ell} knm\ell) = O((nm)^{2k\ell} k\ell m)$ time.

In total, we get the following result.

Theorem 34. *Given a set of n curves G in the plane with at most m vertices each and a positive real number r, we can find a set of k center curves $\mathcal{C}$ with at most ℓ vertices each such that $\min_{c \in \mathcal{C}} \mathrm{d}_{dF}(c, g) \leq r$ for all $g \in G$ or determines that such a set of curves does not exist in $O((nm)^{2k\ell} k\ell mn)$ time.*

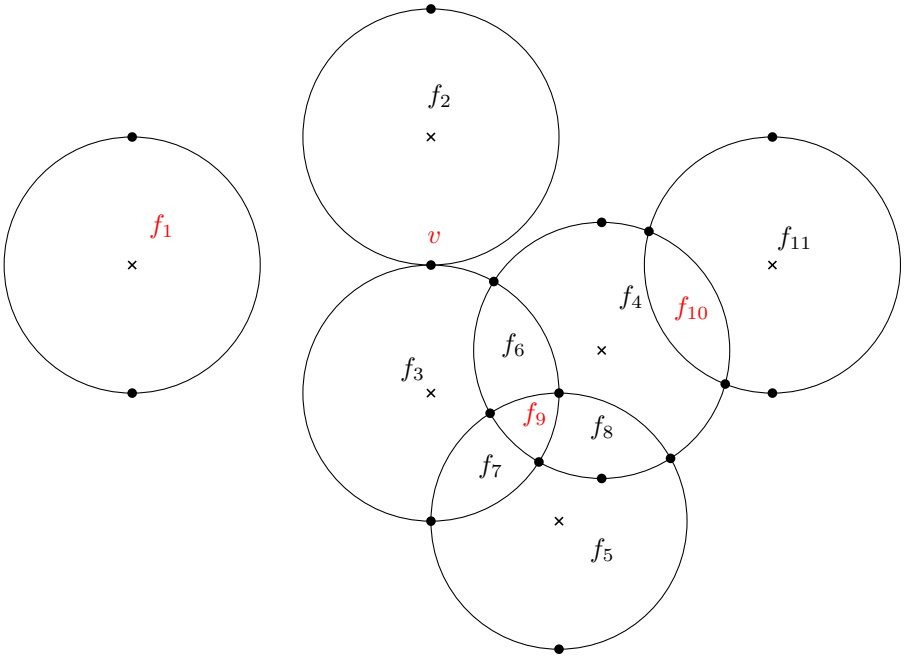

Figure 4.2: An example configuration of $\mathcal{G} = (V, E)$. Crosses indicate the vertices from the curves in G, dots indicate vertices from V and all bounded faces are numbered. The maximal intersection regions are the faces f_1 and f_9 and the vertex v (in red). Note that while all arcs on the boundary of f_2 are convex for that face, f_2 is not maximal, since its boundary intersects the boundary of f_3 only at vertex v.

4.4 Computing $(1, \ell)$-center for DTW in 1D

In this section, we give an exact algorithm to solve the Steiner problem with the center restricted to ℓ vertices for the DTW measure for curves in 1D. Note that for 2D and higher dimensions we are unlikely to solve this problem exactly, even for $\ell = 1$, since the single vertex curves are an instance of the Fermat-Weber problem.

The algorithm is as follows: Given a set of curves G with vertices in $\mathbb{R}$, let S be the list of all distinct vertices on those curves. Iterate over all possible curves c of length at most ℓ that use the vertices from S, compute the distance $\sum_{g \in G} \mathrm{DTW}(g, c)$ and return the curve c that minimizes that distance.

To see that this algorithm provides the optimal center curve, suppose the curve $c^* = \langle c_1^*, \ldots, c_k^* \rangle$ minimizes $\sum_{g \in G} \mathrm{DTW}(g, c^*)$ out of all possible 1D curves of length ℓ. Fix some DTW-alignments that attain this minimum. Define $C(g, i)$ as the set of vertices in g that are aligned with the vertex c_i^* in c^*. By definition of the DTW measure, $\sum_{g \in G} \mathrm{DTW}(g, c^*) = \sum_{g \in G} \sum_{i=1}^{k} \sum_{v \in C(g,i)} |c_i^* - v| = \sum_{i=1}^{k} \sum_{v \in C(i)} |c_i^* - v|$, where $C(i)$ is the multiset containing the vertices from $C(g, i)$ for all $g \in G$. This means that each vertex c_i^* must optimize $\sum_{v \in C(i)} |c_i^* - v|$. This optimum is attained by the median of $C(i)$. Note that in case this median is not a data point in $C(i)$ and lies in between two data points, both of these points attain the optimum. So, for all i either we can choose c_i^* such that $c_i^* \in C(i) \subset S$. This means that it is sufficient to restrict the search to curves with vertices in S and the algorithm indeed finds an optimal solution.

Since we can compute the DTW measure of two curves of size m and ℓ in $O(\ell m)$ and there are $O(|S|^\ell)$ distinct curves with at most ℓ vertices from S, this algorithm takes $O(|S|^\ell \ell m n) = O((mn)^\ell mn\ell)$ time.

Theorem 35. *Given a set of n 1D curves G with at most m vertices each, we can find a center curve c with at most ℓ vertices that minimizes the average-DTW cost for G in $O((nm)^\ell nm\ell)$ time.*

Chapter 5

Conclusions

In this thesis, we have considered the algorithmic problems of computing an optimal k-CENTER for the (discrete) Fréchet distance and k-MEDIAN for the (squared) DTW distance. As we have shown, these problems are all intractable, even when $k = 1$ (assuming P $\neq$ NP). For the k-CENTER problem, even improving an approximation algorithm beyond the basic 2-approximation turns out to be intractable.

These results highlight that obtaining practical algorithms to clustering curves is significantly harder than for clustering points, where the 1-CENTER can be computed in linear time in the number of points for fixed dimension [31], although the dependence on the dimension is exponential. In particular, attacking the k-CENTER and k-MEDIAN problems directly is unlikely to work. So, to nevertheless obtain practical algorithms with theoretical guarantees for these problems, a reasonable direction is to investigate variants of these problems.

In this direction lie the versions where we constrain the complexity of the center to ℓ. The algorithms that exactly solve or give an arbitrarily close approximation to the $(1, \ell)$ version of some problems from Chapter 5 are a first step in that direction. Some of those algorithms are exponential in ℓ. It may not be possible to remove the exponential dependence on ℓ, but it may be possible to at least have algorithms that are fixed parameter tractable in ℓ. It would also be interesting to see if these results can be extended for $k > 1$, to help with the general clustering task.

Another variant, which we have mostly ignored in this thesis, is to consider algorithms that work well when the input is limited to some natural class of curves. In particular, this could be a reasonable approach to approximation algorithms for clustering under the DTW distance, where the lack of a triangle inequality complicates matters. Limiting to certain natural curve classes such as in [2] could lead reasonable approximations in clustering with DTW.

Bibliography

[1] Pankaj K Agarwal, Rinat Ben Avraham, Haim Kaplan, and Micha Sharir. Computing the discrete Fréchet distance in subquadratic time. *SIAM Journal of Computing*, 43(2):429–449, 2014. 5

[2] Pankaj K. Agarwal, Kyle Fox, Jiangwei Pan, and Rex Ying. Approximating dynamic time warping and edit distance for a pair of point sequences. In *Proceedings of the 32nd ACM Symposium on Computational Geometry*, pages 6:1–6:16, 2016. 6, 33

[3] Hee-Kap Ahn, Helmut Alt, Maike Buchin, Eunjin Oh, Ludmila Scharf, and Carola Wenk. A middle curve based on discrete Fréchet distance. In *Proceedings of the 12th Latin American Symposium on Theoretical Informatics*, pages 14–26. Springer, 2016. 6

[4] Ghazi Al-Naymat, Sanjay Chawla, and Javid Taheri. SparseDTW: A novel approach to speed up dynamic time warping. In *Proceedings of the 8th Australasian Data Mining Conference*, volume 101, pages 117–127. Australian Computer Society, Inc., 2009. 6

[5] Helmut Alt and Michael Godau. Computing the Fréchet distance between two polygonal curves. *International Journal of Computational Geometry & Applications*, 5:75–91, 1995. 2

[6] Chanderjit Bajaj. The algebraic degree of geometric optimization problems. *Discrete & Computational Geometry*, 3(2):177–191, 1988. 15

[7] Sotiris Brakatsoulas, Dieter Pfoser, Randall Salas, and Carola Wenk. On map-matching vehicle tracking data. In *Proceedings of the 31st International Conference on Very Large Data Bases*, VLDB '05, pages 853–864. VLDB Endowment, 2005. 5

[8] K. Bringmann and M. Künnemann. Quadratic conditional lower bounds for string problems and dynamic time warping. In *Proceedings of the 56th Annual Symposium on Foundations of Computer Science*, pages 79–97, Oct 2015. 6

[9] Karl Bringmann. Why walking the dog takes time: Fréchet distance has no strongly subquadratic algorithms unless SETH fails. In *Proceedings of the 55th Annual Symposium on Foundations of Computer Science (FOCS)*, pages 661–670. IEEE, 2014. 5

[10] Karl Bringmann and Wolfgang Mulzer. Approximability of the discrete Fréchet distance. *Journal of Computational Geometry*, 7(2):46–76, 2015. 5

[11] Kevin Buchin, Maike Buchin, Wouter Meulemans, and Wolfgang Mulzer. Four Soviets walk the dog: Improved bounds for computing the Fréchet distance. *Discrete & Computational Geometry*, 58(1):180–216, Jul 2017. 5

[12] Kevin Buchin, Anne Driemel, Joachim Gudmundsson, Michael Horton, Irina Kostitsyna, Maarten Löffler, and Martijn Struijs. Approximating (k, ℓ)-center clustering for curves. To appear in Proceedings of the 30th ACM-SIAM Symposium on Discrete Algorithms. Preprint available at `https://arxiv.org/abs/1805.01547`, 2019. 6, 12, 29

[13] Kevin Buchin, Tim Ophelders, and Bettina Speckmann. SETH says: Weak Fréchet distance is faster, but only if it is continuous and in one dimension. To appear in Proceedings of the 30th ACM-SIAM Symposium on Discrete Algorithms, 2019. 5

[14] Bernard Marie Chazelle and Der-Tsai Lee. On a circle placement problem. *Computing*, 36(1-2):1–16, 1986. 30

[15] Anne Driemel. *Realistic analysis for algorithmic problems on geographical data*. PhD thesis, Utrecht University, 2013. 27

[16] Anne Driemel, Sariel Har-Peled, and Carola Wenk. Approximating the Fréchet distance for realistic curves in near linear time. *Discrete & Computational Geometry*, 48(1):94–127, Jul 2012. 5

[17] Anne Driemel, Amer Krivošija, and Christian Sohler. Clustering time series under the Fréchet distance. In *Proceedings of the 27th ACM-SIAM Symposium on Discrete Algorithms*, pages 766–785. Society for Industrial and Applied Mathematics, 2016. 6

[18] Adrian Dumitrescu and Günter Rote. On the Fréchet distance of a set of curves. In *Proceedings of the 16th Canadian Conference on Computational Geometry*, pages 162–165, 2004. 13

[19] Thomas Eiter and Heikki Mannila. Computing discrete Fréchet distance. Technical Report CD-TR 94/64, Christian Doppler Laboratory for Expert Systems, TU Vienna, Austria, 1994. 5

[20] Sariel Har-Peled and Benjamin Raichel. The Fréchet distance revisited and extended. *ACM Transactions on Algorithms (TALG)*, 10(1):3, 2014. 6

[21] Eamonn Keogh and Chotirat Ann Ratanamahatana. Exact indexing of dynamic time warping. *Knowledge and information systems*, 7(3):358–386, 2005. 5

[22] T. Warren Liao. Clustering of time series data—a survey. *Pattern Recognition*, 38(11):1857 – 1874, 2005. 5

[23] Ariane Mascret, Thomas Devogele, Iwan Le Berre, and Alain Hénaff. Coastline matching process based on the discrete Fréchet distance. In *Progress in Spatial Data Handling*, pages 383–400. Springer, 2006. 5

[24] François Petitjean and Pierre Gançarski. Summarizing a set of time series by averaging: From Steiner sequence to compact multiple alignment. *Theoretical Computer Science*, 414(1):76 – 91, 2012. 5, 23

[25] Kari-Jouko Räihä and Esko Ukkonen. The shortest common supersequence problem over binary alphabet is NP-complete. *Theoretical Computer Science*, 16(2):187 – 198, 1981. 7, 9, 24

[26] Hiroaki Sakoe and Seibi Chiba. Dynamic programming algorithm optimization for spoken word recognition. *IEEE transactions on acoustics, speech, and signal processing*, 26(1):43–49, 1978. 5

[27] Stan Salvador and Philip Chan. Toward accurate dynamic time warping in linear time and space. *Intelligent Data Analysis*, 11(5):561–580, 2007. 6

[28] Carlos D. Santos, Stefanie Neupert, Hans-Peter Lipp, Martin Wikelski, and Dina K. N. Dechmann. Temporal and contextual consistency of leadership in homing pigeon flocks. *PLOS ONE*, 9(7):1–5, 07 2014. 1

[29] Haozhou Wang, Han Su, Kai Zheng, Shazia Sadiq, and Xiaofang Zhou. An effectiveness study on trajectory similarity measures. In *Proceedings of the 24th Australasian Database Conference*, volume 137 of *ADC '13*, pages 13–22, Darlinghurst, Australia, Australia, 2013. Australian Computer Society, Inc. 5

[30] Lusheng Wang and Tao Jiang. On the complexity of multiple sequence alignment. *Journal of computational biology*, 1(4):337–348, 1994. 9, 23

[31] Emo Welzl. Smallest enclosing disks (balls and ellipsoids). In *New results and new trends in computer science*, pages 359–370. Springer, 1991. 33

[32] Rui Xu and Donald Wunsch. Survey of clustering algorithms. *IEEE Transactions on neural networks*, 16(3):645–678, 2005. 5