(2 + \epsilon)-ANN for time series under the Fréchet distance

Anne Driemel\(^1\) and Ioannis Psarros\(^1\)

\(^1\)Hausdorff Center for Mathematics, University of Bonn, Germany,
driemel@cs.uni-bonn.de, ipsarros@uni-bonn.de

August 24, 2020

Abstract

We give the first ANN-data structure for time series under the continuous Fréchet distance, where ANN stands for approximate near neighbor. Given a parameter \(\epsilon \in (0, 1]\), the data structure can be used to preprocess \(n\) curves in Euclidean \(\mathbb{R}\) (aka time series), each of complexity \(m\), to answer queries with a curve of complexity \(k\) by either returning a curve that lies within Fréchet distance \(2 + \epsilon\), or answering that there exists no curve in the input within distance 1. In both cases, the answer is correct.

Our data structure uses space in \(n \cdot \mathcal{O}\left(\epsilon^{-1}\right)^k + \mathcal{O}(nm)\) and query time in \(\mathcal{O}(k)\). The data structure is therefore especially useful in the asymmetric setting, where \(k \ll m\). This is well-motivated for the continuous Fréchet distance as the distance measure takes into account the interior points along the edges of the curve.

We show that the approximation factor achieved by our data structure is optimal in the cell-probe model of computation. Concretely, we show that for any data structure which achieves an approximation factor less than 2 and which supports curves of arclength at most \(L\), uses a word size bounded by \(\mathcal{O}(L^{1-\epsilon})\) for some constant \(\epsilon > 0\), and answers the query using only a constant number of probes, the number of words used to store the data structure must be at least \(L^{\Omega(k)}\). Our data structure uses only a constant number of probes per query and does not have any dependency on \(L\).

We apply similar techniques for proving cell-probe lower bounds for the discrete Fréchet distance matching known upper bounds.
1 Introduction

Recent years have seen a raised interest in proximity data structures for trajectory analysis under the Fréchet distance \[15, 24, 33, 7, 20, 12, 2, 18, 6, 19, 22, 23, 16, 5\]. An intuitive definition of the Fréchet distance uses the metaphor of a person walking a dog. Imagine the dog walker being restricted to follow the path defined by the first curve while the dog is restricted to the second curve. In this analogy, the Fréchet distance is the shortest length of a dog leash that makes a dog walk feasible. See Section 1.2 for an exact definition of the distance measure. Despite the popularity of the Fréchet distance it is still an open problem how to build efficient data structures for it. Known results either suffer from a large approximation factor or high complexity bounds with dependency on the arclength of the curve, or only support a very restricted set of queries. See Section 1.1 for a detailed discussion of previous results. In general, for any data structuring problem, one can consider two extremal regimes of the tradeoff between query-time and space. We focus on solutions which are query-time efficient with good approximation factor, while the space complexity can be high. Typically, the ANN data structures in this regime store a set of representative query candidates together with precomputed answers for these queries so that a query can be answered approximately with a lookup table. One example of this approach is the data structure for the \((1 + \epsilon)\)-ANN problem under the discrete Fréchet distance by Filtser et al. [23] which discretizes the query space with a grid and stores representative point sequences on this grid. One can try to follow the same approach in the case of continuous curves. Computing good representatives in this case is more intricate though: two curves may be near but some of their vertices may be far from any other vertex on the other curve. Hence, picking representative curves which are defined by vertices in the proximity of the vertices of the data curve is not sufficient. This issue can be easily resolved if we assume that the arclength of the curve is bounded. In that case, one can enumerate all curves which are defined by grid points and lie within a given distance threshold. Building a data structure which does not need this assumption is the main purpose of this paper. We focus on the important case of time series, which can be seen as continuous curves in \(\mathbb{R}\). In the analogy of the dog walker, both the walker and the dog are restricted to a path that goes back and forth on the same line. For time series, the complexity of a curve can be defined as the number of extrema of the function graph. We present a data structure, which, in addition to storing the input, requires space that is only linear in the number of input curves \(n\) and achieves query times that are independent of the complexity of each input curve \(m\). In fact, the query time only depends on the supported complexity of the query \(k\). As such, our data structure is especially useful in the asymmetric setting, where \(k \ll m\), i.e., when we assume small query complexity. We supplement our upper bounds with an analysis in the cell-probe model indicating that the approximation factor \((2 + \epsilon)\) achieved by our data structure is almost tight. We show that, under reasonable assumptions, any data structure which stores one curve (i.e., \(n = 1\)) and answers queries of complexity \(k\) with approximation factor strictly better than 2 and with only a constant number of memory accesses, needs space proportional to \(L^k\), where \(L\) denotes the maximum arclength of the query curves supported by the data structure. Our data structure uses only a constant number of probes per query and does not have any dependency on \(L\).

1.1 Previous work

Most previous results on data structures for ANN search of curves, concern the discrete Fréchet distance (see Section 1.4 for the exact definition). These results are very relevant in our setting, since the continuous Fréchet distance can be approximated using the discrete Fréchet distance.

The first non-trivial ANN-data structure for the discrete Fréchet distance from 2002 by Indyk [28] achieved approximation factor \(O((\log m + \log \log n)^{\epsilon^{-1}})\), where \(m\) is the maximum length of
a sequence, and \( t > 1 \) is a trade-off parameter. More recently, in 2017, Driemel and Silvestri designed a data structure which achieves approximation factor \( O(k) \), where \( k \) is the length of the query sequence. They show how to improve the approximation factor to \( O(d^{3/2}) \) at the expense of additional space usage, and a follow-up result by Emiris and Psarros achieves a \((1 + \epsilon)\) approximation, at the expense of further increasing space usage. Recently, Filtser et al. showed how to build a \((1 + \epsilon)\)-approximate data structure using space in \( n \cdot O(1/\epsilon)^{kd} \) and with query time in \( O(kd) \). When using the discrete Fréchet to approximate the continuous Fréchet distance, all known transformations introduce a dependency on the arclength of the curves (resp. the maximum length of an edge), either in the complexity bounds or in the approximation factor. It is not at all obvious how to avoid this. We next discuss results on the related problem of range searching that manage to circumvent this problem. The subset of input curves, that lie within the search radius of the query curve is called the range of the query. A range searching query should return all input curves inside the range, and a range counting query should return the number thereof. In this terminology, an ANN-query should return at least one element of the range, if the range is non-empty. Interestingly, there are some data structures for range searching, which are especially tailored to the case of the continuous Fréchet distance and which do not have a dependency on the arclength. The work of de Berg et al. focuses on preprocessing a single polygonal curve into a data structure to support range counting queries among its subcurves. Here, a query curve is restricted to be a line segment. Gudmundsson and Smid consider the problem of preprocessing a tree embedded in the plane so that given a query polygonal curve, one can decide if there is a path in the tree which is within Fréchet distance \( \delta \) for some threshold \( \delta > 0 \). Driemel and Afshani consider the exact range searching problem in the general case. For \( n \) curves of complexity \( m \) in \( \mathbb{R}^2 \), their data structure uses space in \( \mathcal{O}(n (\log \log n)^{\mathcal{O}(m^2)}) \) and the query time costs \( \mathcal{O}(\sqrt{n \log \mathcal{O}(m^2) n}) \), assuming that the complexity of the query curves is at most \( \log \mathcal{O}(1) \) \( n \). More importantly, they show lower bounds in the pointer model of computation that match the number of log factors used in the upper bounds asymptotically. Our lower bounds also hold for the case of range searching, but we assume a different computational model, namely the cell-probe model.

From a practical perspective, range searching has drawn a lot of attention, due to the many applications. The 2017 SIGSPATIAL Cup asked for practical solutions for the exact range searching problem under the Fréchet distance. The top solutions use heuristics for lower-bounding the distance to filter the candidates and reduce the amount of time needed to answer a query.

1.2 Problem Statement

**Definition 1** (Fréchet distance). Given two curves \( \pi, \tau : [0, 1] \to \mathbb{R} \), their Fréchet distance is:

\[
d_F(\pi, \tau) = \min_{f : [0,1] \to [0,1]} \max_{g : [0,1] \to [0,1]} \| \pi(f(\alpha)) - \tau(g(\alpha)) \|_2,
\]

where \( f \) and \( g \) range over all continuous, non-decreasing functions with \( f(0) = g(0) = 0 \), and \( f(1) = g(1) = 1 \).

**Definition 2** (c-ANN problem). The input consists of \( n \) curves \( \Pi \) in \( \mathbb{R}^d \). Given a distance threshold \( r > 0 \), an approximation factor \( c > 1 \), preprocess \( \Pi \) into a data structure such that for any query \( \tau \), the data structure reports as follows:

- if there exists a \( \pi \in \Pi \) s.t. \( d_F(\pi, \tau) \leq r \), then it returns \( \pi' \in \mathcal{P} \) s.t. \( d_{dF}(\pi, \tau) \leq cr \),
- if \( \forall \pi \in \Pi, d_F(\pi, \tau) \geq cr \) then it returns “no”,
- otherwise, it either replies with a curve \( \pi \in \Pi \) s.t. \( d_F(\pi, \tau) \leq cr \), or with “no”.
The approximate nearest neighbor problem is known \cite{indyk1998approximate} to reduce to a sequence of ANN problems.

### 1.3 Our contribution

We study the $c$-ANN problem for time series under the continuous Fréchet distance. Our data structures operate in the real-RAM model, enhanced with floor function operations in constant time. For a more detailed discussion on the computational models used in this paper we refer to Appendix A.

**Theorem 3 (Main Theorem).** Let $\epsilon \in (0, 7]$. There is a data structure for the $(2 + \epsilon)$-ANN problem, which stores $n$ curves in $\mathbb{R}$ and supports query curves of complexity $k$ in $\mathbb{R}$, which uses space in $n \cdot O\left(\frac{1}{\epsilon}\right)^k + O(nm)$, needs $n \cdot O\left(\frac{1}{\epsilon}\right)^k + O(nmk^3)$ preprocessing time and answers a query in $O(k)$ time.

The proof of the above theorem can be found in Section 3. To achieve this result we generate a discrete approximation of the set of queries that have non-empty ranges. To this end, we employ the concept of signatures, previously introduced in \cite{babenko2020approximate}. The signature of a time series provides us with a selection of the local extrema of the function graph, which we use to generate candidate curves.

We first apply this technique in Section 2 in combination with the triangle inequality. This does not yield the desired approximation factor, yet. A more careful analysis of the involved matchings and a more intricate algorithm to build the candidate set which we present in Section 3 yields the above theorem.

Our second main result is a lower bound in the cell-probe model of computation.

**Theorem 4.** Consider any Fréchet distance oracle with approximation factor $2 - \gamma$, for any $\gamma \in (0, 1]$, in the cell-probe model, which supports curves in $\mathbb{R}$ as follows: it stores any polygonal curve of arclength at most $L$, for $L \geq 6$, it supports queries of arclength at most $L$ and complexity $k$, where $k \leq L/6$, and it achieves performance parameters $t, w, s$. There exist

$$w_0 = \Omega\left(\frac{k}{t} \left(\frac{L}{k}\right)^{1-\epsilon}\right), \quad s_0 = 2^{\Omega\left(\frac{k \log(k/k)}{t}\right)}$$

such that if $w < w_0$ then $s \geq s_0$, for any constant $\epsilon > 0$.

The above theorem implies that the approximation factor of $(2 + \epsilon)$ in Theorem 3 cannot be significantly improved, unless we restrict ourselves to curves of bounded arclength or increase the number of probes. The proof can be found in Section 4.

To achieve this result we observe that a technique first introduced by Miltersen \cite{miltersen2000communication} can be applied here. Miltersen shows that lower bounds for communication problems can be translated into lower bounds for cell-probe data structures. In particular, we use a reduction from the lopsided disjointness problem. Such reductions are not new. A similar reduction was devised by Meintrup et al. \cite{meintrup2014lower} in order to lower bound the bit complexity required for sketching, but it works for polygonal curves in $\mathbb{R}^2$, only. To the best of our knowledge the lower bound of the above theorem is new.

In addition, we extend these lower bound results to the case of the discrete Fréchet distance. Here, our reduction is more intricate. We adapt a reduction by Bringmann and Mulzer \cite{bringmann2012efficient}, which was used for showing lower bounds for computing the Fréchet distance. Our results show that for the corresponding data structure problem an exponential dependence on $k$ for the space is necessary, when the number of probes is constant. This exponential dependence on $k$ appears, e.g., in the upper bound by Filtser et al. \cite{filtser2019approximate}.
1.4 Preliminaries

Throughout this paper, a curve is a function $[0,1] \mapsto \mathbb{R}$ and we may refer to such a curve as a time series. We can define a curve $\pi$ as $\pi := \langle x_1, \ldots, x_m \rangle$, which means that $\pi$ is obtained by linearly interpolating $x_1, \ldots, x_m$. The vertices of $\pi : [0,1] \mapsto \mathbb{R}$ are those points which are local extrema in $\pi$. So if $v_1, \ldots, v_\ell$ are the vertices of some curve $\pi$, then $\pi = \langle v_1, \ldots, v_\ell \rangle$. For any curve $\pi$, $V(\pi)$ denotes the ordered set of vertices of $\pi$. The number of vertices $|V(\pi)|$ is called the complexity of $\pi$ and it is also denoted by $|\pi|$. For any two points $x, y$, $\overrightarrow{x,y}$ denotes the directed line segment connecting $x$ with $y$ in the direction from $x$ to $y$. The segment defined by any two consecutive vertices is called an edge. For any two $0 \leq p_a < p_b \leq 1$ and any curve $\pi$, we denote by $\pi[p_a,p_b]$ the subcurve $\{\pi(x) \mid x \in [p_a,p_b]\}$. For any two curves $\pi_1, \pi_2$, with vertices $x_1, \ldots, x_k$ and $x_k, \ldots, x_m$ respectively, $\pi_1 \circ \pi_2$ denotes the curve $\langle x_1, \ldots, x_k, \ldots, x_m \rangle$, that is the concatenation of $\pi_1$ and $\pi_2$. We define the arclength $\lambda(\tau)$ of a curve $\tau$ as the total sum of lengths of the edges of $\tau$.

We refer to a pair of continuous, non-decreasing functions $f : [0,1] \mapsto [0,1]$, $g : [0,1] \mapsto [0,1]$ such that $\{f(0) = g(0), f(1) = g(1)\}$, as a matching. We will also use the following concept introduced by Alt and Godau [3].

Definition 5 ($\delta$-free space). Given two curves $\pi : [0,1] \mapsto \mathbb{R}$, $\tau : [0,1] \mapsto \mathbb{R}$. The $\delta$-free space is the subset of the parametric space defined as $\{(x,y) \in [0,1]^2 \mid |\pi(x) - \tau(y)| \leq \delta\}$.

The standard algorithm by Alt and Godau [4] for computing the Fréchet distance between two curves $\pi, \tau$, finds a matching in the parametric space of the two curves, where a matching is realized by a monotone path which starts at $(0,0)$ and ends at $(1,1)$. If such a path is entirely contained in the $\delta$-free space, then $d_F(\pi,\tau) \leq \delta$. The Fréchet distance is known to satisfy the triangular inequality. We use the following two observations repeatedly in the paper.

(i) For any curves $\tau_1, \tau_2, \pi_1, \pi_2$, which satisfy the property that the last vertex of $\tau_1$ is the first vertex of $\tau_2$ and the last vertex of $\pi_1$ is the first vertex of $\pi_2$, it holds that $d_F(\pi_1 \circ \tau_2, \pi_1 \circ \tau_2) \leq \max\{d_F(\tau_1,\tau_2), d_F(\pi_1,\pi_2)\}$.

(ii) For any two edges $a_1 \overrightarrow{a_2}$, $b_1 \overrightarrow{b_2}$, it holds that $d_F(\overrightarrow{a_1a_2}, \overrightarrow{b_1b_2}) = \max\{|a_1 - b_1|, |a_2 - b_2|\}$.

These two facts imply that if $\pi_1 = \langle x_1, \ldots, x_m \rangle$ and $\pi_2 = \langle y_1, \ldots, y_m \rangle$ such that for each $i = 1, \ldots, m$, $|x_i - y_i| \leq \varepsilon$ then $d_F(\pi_1,\pi_2) \leq \varepsilon$. This is a key property that we exploit when we snap vertices of a curve to a grid, since it allows us to bound the distance between the original curve, and the curve defined by the sequence of snapped vertices.

The main ingredient of our algorithms is the notion of signatures which was first introduced in [17] and capture critical points of the input time series. We define signatures as follows [4].

Definition 6 ($\delta$-signatures). We define the $\delta$-signature of any curve $\tau : [0,1] \mapsto \mathbb{R}$ as follows. The signature is a curve $\sigma : [0,1] \mapsto \mathbb{R}$ defined by a series of values $0 = t_1 < \cdots < t_\ell = 1$ as the linear interpolation of $\tau(t_i)$ in the order of the index $i$, and with the following properties. For $1 \leq i \leq \ell - 1$ the following conditions hold:

i) (non-degeneracy) if $i \in [2,\ell - 1]$ then $\tau(t_i) \notin \tau(t_{i-1}), \tau(t_{i+1})$,

ii) (direction-preserving) if $\tau(t_i) < \tau(t_{i+1})$ for $t < t' \in [t_i, t_{i+1}]$: $\tau(t) - \tau(t') \leq 2\delta$ and if $\tau(t_i) > \tau(t_{i+1})$ for $t < t' \in [t_i, t_{i+1}]$: $\tau(t') - \tau(t) \leq 2\delta$,

iii) (minimum edge length) if $i \in [2,\ell - 2]$ then $|\tau(t_i + 1) - \tau(t_i)| > 2\delta$, and if $i \in [1,\ell - 1]$ then $|\tau(t_{i+1}) - \tau(t_i)| > \delta$.

\footnote{Note that the definition in [17] contains a typo, which is corrected here.}
For any $\delta > 0$ and any curve $\pi : [0, 1] \mapsto \mathbb{R}$ of complexity $m$, the $\delta$-signature of $\pi$ can be computed in $O(m)$ time [17]. We now state some basic results about signatures.

**Lemma 7** (Lemma 3.1 [17]). It holds for any $\delta$-signature $\sigma$ of $\tau$ that $d_F(\sigma, \tau) \leq \delta$.

**Lemma 8** (Lemma 3.2 [17]). Let $\sigma$ with vertices $v_1, \ldots, v_\ell$, be a $\delta$-signature of $\pi$ with vertices $u_1, \ldots, u_m$. Let $r_i = [v_i - \delta, v_i + \delta]$, for $1 \leq i \leq \ell$, be ranges centered at the vertices of $\sigma$ ordered along $\sigma$. It holds for any time series $\tau$ if $d_F(\pi, \tau) \leq \delta$, then $\tau$ has a vertex in each range $r_i$, and such that these vertices appear on $\tau$ in the order of $i$.

We end this section with the standard definition of the discrete Fréchet distance.

**Definition 9** (Traversals). Given $P = p_1, \ldots, p_m \in (\mathbb{R}^d)^m$ and $Q = q_1, \ldots, q_k \in (\mathbb{R}^d)^k$, a traversal $T = (i_1, j_1), \ldots, (i_\ell, j_\ell)$ of $P$ and $Q$ is a sequence of pairs of indices referring to a pairing of points from the two sequences such that:

(i) $i_1, j_1 = 1$, $i_\ell = m$, $j_\ell = k$.

(ii) $\forall (i_u, j_u) \in T : i_{u+1} - i_u \in \{0, 1\}$ and $j_{u+1} - j_u \in \{0, 1\}$.

(iii) $\forall (i_u, j_u) \in T : (i_{u+1} - i_u) + (j_{u+1} - j_u) \geq 1$.

For any traversal $T$, we define $d_T(P, Q) := \max_{(i, j) \in T} \|p_i - q_j\|_2$.

**Definition 10** (Discrete Fréchet distance). Given $P = p_1, \ldots, p_m \in (\mathbb{R}^d)^m$ and $Q = q_1, \ldots, q_k \in (\mathbb{R}^d)^k$, we define the discrete Fréchet distance between $P$ and $Q$ as follows:

$$d_{dF}(P, Q) = \min_{T \in \mathcal{T}} \max_{(i_u, j_u) \in T} \|p_{i_u} - q_{j_u}\|_2,$$

where $\mathcal{T}$ denotes the set of all possible traversals for $P, Q$. Thus, $d_{dF}(P, Q) = \min_{T \in \mathcal{T}} d_T(P, Q)$.

## 2 A constant-factor approximation for time series

In this section, we show a data structure for the $(5 + \epsilon)$-ANN problem of time series. We initiate our exposition with a simple corollary regarding the covering properties of ranges centered at signature vertices.

**Corollary 11.** Let $\sigma_{\pi}$ be a $2\delta$-signature of $\tau$ and let $\sigma_{\pi}$ be a $\delta$-signature of $\pi$ with vertices $v_1, \ldots, v_\ell$. Let $r_i := [v_i - 2\delta, v_i + 2\delta]$, for $i \in [\ell]$, be ranges at the vertices of $\sigma_{\pi}$ ordered along $\sigma_{\pi}$. If $d_{dF}(\pi, \tau) \leq \delta$, then the vertices of $\sigma_{\pi}$ are contained in $\bigcup_{i=1}^{\ell} r_i$ and the vertices of $\sigma_{\pi}$ appear in the ranges $r_i$, in the order of $i$.

**Proof.** If $d_{dF}(\pi, \tau) \leq \delta$ then by the triangular inequality and Lemma 7, $d_{dF}(\sigma_{\pi}, \tau) \leq 2\delta$. Let $u_1, \ldots, u_{\ell'}$ be the vertices of $\sigma_{\pi}$ and define for each $i \in [\ell']$, $r'_i := [u_i - 2\delta, u_i + 2\delta]$. By Lemma 8, $\sigma_{\pi}$ has a vertex in each range $r'_i$ and these vertices appear on $\sigma_{\pi}$ in the order of $i$. $\square$
The data structure  The input consists of a set $\Pi$ of $n$ curves in $\mathbb{R}$, the distance threshold $r > 0$, and the approximation error $\epsilon > 0$. We assume that the distance threshold is $r := 1 - \epsilon/10$. (To solve the problem for a different value of $r$, the input set can be uniformly scaled.) Let $G_w := \{i \cdot w \mid i \in \mathbb{Z}\}$ be the regular grid with side-length $w := \epsilon/10$. Let $H$ be a hashtable, which is initially empty. For $H$, we assume perfect hashing, implying that for any key of complexity $O(k)$, we need $O(k)$ time to access the corresponding bucket. For each input curve $\pi \in \Pi$, we compute its 1-signature $\sigma_\pi$, with vertices $V(\sigma_\pi) = v_1, \ldots, v_\ell$, and for each $v_i \in V(\sigma_\pi)$ we define the range $r_i := [v_i - 2, v_i + 2]$. Corollary [11] ensures that the vertices of the 2-signature of a query $\tau$, for which it holds that $d_F(\pi, \tau) \leq 1$, lie in the ranges $r_i$ satisfying the order of $i$. Hence, we enumerate all curves with at most $k$ vertices, chosen from the sets $r_1 \cap G_w, r_2 \cap G_w, \ldots$, and satisfying the order of $i$, and we store them in a set $C'$. Next, we compute the set $C(\pi) := \{\sigma \in C' \mid d_F(\sigma, \pi) \leq 3\}$. We store $C(\pi)$ in $H$ as follows: for each $\sigma \in C(\pi)$, we use as key $\kappa(\sigma)$ the sequence of its vertices. Let $H(\sigma)$ be the bucket with key $\kappa(\sigma)$. Then, for each $\sigma \in C(\pi)$, we store in $H(\sigma)$ a pointer to $\pi$. Thus, after processing all input curves, $H(\sigma)$ contains a list of all relevant pointers to curves in $\Pi$. The total space required is $O(n \cdot \max_{\pi \in \Pi} |C(\pi)|)$.

Our intuition is the following. We would like the set $C(\pi)$ to contain all those curves that correspond to 2-signatures of query curves that have $\pi$ as an approximate near neighbor in the set $\Pi$. So when presented with a query we can simply compute its 2-signature and do a lookup in the table $H$. However, this set is infinite. Therefore, we snap the vertices to a grid to obtain a discrete set of bounded size.

The query algorithm  When presented with a query curve $\tau$, we first compute a 2-signature $\sigma_\tau$, and then we compute a key by snapping the vertices to the same grid $G_w$. Snapping to $G_w$ is implemented as follows: if $V(\sigma_\tau) = v_1, \ldots, v_\ell$ then $\sigma'_\tau := \langle g_w(v_1), \ldots, g_w(v_\ell) \rangle$, where for any $x \in \mathbb{R}$, $g_w(x)$ is the nearest point of $x$ in $G_w$. We perform a lookup in the hashtable $H$ with the key $\kappa(\sigma'_\tau)$ and return the result: if there is a bucket $H(\sigma'_\tau)$ then we return any curve which has a pointer stored there, otherwise we return “no”. Section 2.1 contains detailed pseudocode of the basic algorithms.

Lemma 12. Let $\tau$ be a query curve of complexity $k$. If there exists a curve $\pi \in \Pi$ such that $d_F(\pi, \tau) \leq 1 - \epsilon/10$ then the query algorithm returns a curve $\tau' \in \Pi$ such that $d_F(\tau', \tau) \leq 5 + \epsilon/10$. If for any curve $\pi \in \Pi$, $d_F(\pi, \tau) > 5 + \epsilon/10$ then the query algorithm returns “no”.

Proof. Let $\pi$ be any input curve in $\Pi$, and let $\tau$ be a query curve. First suppose that $d_F(\pi, \tau) \leq 1 - \epsilon/10$. By Lemma [7] we have that $d_F(\tau, \sigma_\tau) \leq 2$, and by the triangular inequality,

$$d_F(\pi, \sigma_\tau) \leq d_F(\pi, \tau) + d_F(\tau, \sigma_\tau) \leq 3 - \epsilon/10.$$  

Then, again, by the triangular inequality,

$$d_F(\pi, \sigma'_\tau) \leq d_F(\pi, \sigma_\tau) + d_F(\sigma_\tau, \sigma'_\tau) \leq 3.$$  

By Corollary [11] we are guaranteed that $\sigma'_\tau$ will be considered during preprocessing, so there will be a pointer to $\pi$ in the bucket $H(\sigma'_\tau)$.

Now consider the case $d_F(\pi, \tau) > 5 + \epsilon/10$. By Lemma [7] we have that $d_F(\tau, \sigma_\tau) \leq 2$. By the triangular inequality,

$$d_F(\pi, \sigma_\tau) \geq d_F(\pi, \tau) - d_F(\tau, \sigma_\tau) > 3 + \epsilon/10$$  

and then,

$$d_F(\pi, \sigma'_\tau) \geq d_F(\pi, \sigma'_\tau) - d_F(\sigma_\tau, \sigma'_\tau) > 3,$$

which means that $\sigma'_\tau$ will not be assigned to $C(\pi)$ during preprocessing. The approximation factor is $\frac{5 + \epsilon/10}{1 - \epsilon/10} < 5 + \epsilon$, for any $\epsilon \in [0, 4]$. 

\[\square\]
Figure 1: Two examples which show that the analysis of the approximation factor of our data structure presented in Section 2 is essentially tight. In both figures, we show the function graphs of the curves $\pi : [0, 1] \to \mathbb{R}$ (an input curve) and $\tau : [0, 1] \to \mathbb{R}$ (a query curve) and $\sigma_\tau : [0, 1] \to \mathbb{R}$ (the signature of $\tau$).

**Theorem 13.** Let $\epsilon \in (0, 4]$. There is a data structure for the $(5 + \epsilon)$-ANN problem, which stores $n$ curves in $\mathbb{R}$ and supports query curves of complexity $k$ in $\mathbb{R}$, which uses space in $n \cdot \mathcal{O}\left(\frac{1}{\epsilon}\right)^k + \mathcal{O}(nm)$ preprocessing time and answers a query in $\mathcal{O}(k)$ time.

**Proof.** The data structure is described above. By Lemma 12 the data structure returns a correct result. It remains to analyze the complexity. Our data structure solves the $(5 + \epsilon)$-ANN problem with distance threshold $1 - \epsilon/10$. The space required for each input curve is proportional to the number of candidate signatures computed in the preprocessing phase. Indeed, we will show now that $|C'| \leq \mathcal{O}\left(\frac{1}{\epsilon}\right)^k$. Notice that if there exists a curve with $k$ vertices which is within distance 1 from $\pi$ then $\ell \leq k$, by Lemma 8. Recall that the curves in $|C'|$ have vertices in the ranges $r_i \cap G_w$ and the vertices respect the order of $i$. If we fix the choices of $t_1, \ldots, t_\ell$, where each $t_i$ denotes the number of vertices in $r_i \cap G_w$ to be used in the creation of those curves, we can produce at most $\prod_{i=1}^{\ell} |r_i \cap G_w|^{t_i}$ distinct curves. Hence,

$$|C'| \leq \sum_{\ell_1 + \ldots + \ell_\ell = k} \prod_{i=1}^{\ell} \left(\frac{40}{\epsilon} + 1\right)^{t_i} \leq \sum_{\ell_1 + \ldots + \ell_\ell = k} \left(\frac{40}{\epsilon} + 1\right)^k \leq \left(k + \ell - 1\right) \cdot \left(\frac{40}{\epsilon} + 1\right)^k = \mathcal{O}\left(\frac{1}{\epsilon}\right)^k.$$

The time to compute a signature for a curve of complexity $m$ is $\mathcal{O}(m)$, because we can use the algorithm of [17]. So, we have a total of $\mathcal{O}(nm) + n \cdot \mathcal{O}(1/\epsilon)^k$ for the preprocessing time, $n \cdot \mathcal{O}(1/\epsilon)^k$ space for the data structure and $\mathcal{O}(nm)$ space for storing the input curves and each query costs $\mathcal{O}(k)$ time, since we employ perfect hashing for $\mathcal{H}$, and snipping a curve costs $\mathcal{O}(k)$ time assuming that a floor function operation needs $\mathcal{O}(1)$ time.

Deciding whether a query curve $\tau$ is near to a given curve $\pi$ by only having a 2-signature of $\tau$ is subject to a $\pm 2$ error. Figure 1 shows an example where this approximation factor is attained.
2.1 Pseudocode of the basic result

2.1.1 Preprocessing

preprocess(set of time series Π)
1: Initialize empty hashtable H
2: for each π ∈ Π do
3: \( C(\pi) \leftarrow \text{generate_candidates}(\pi) \)
4: if \( C(\pi) \neq \emptyset \) then
5: for each στ ∈ \( C(\pi) \) do
6: add a pointer to \( \pi \) in \( H(\sigma \tau) \)

\text{generate_candidates}(time series π)
1: \( \sigma \pi \leftarrow 1\text{-signature of } \pi, \text{ with } V(\sigma \pi) = v_1, \ldots, v_\ell \)
2: if \( \ell > k \) then
3: return \( \emptyset \)
4: for each \( i = 1, \ldots, \ell \) do
5: \( r_i \leftarrow [v_i - 2, v_i + 2] \)
6: \( C' \leftarrow \emptyset \)
7: for each \( j = 1, \ldots, \ell \) do
8: for each \( p \in r_j \cap G_w \) do
9: \( \text{generate_sequences}(\langle p \rangle, j, C') \)
10: \( C(\pi) \leftarrow \emptyset \)
11: for each στ ∈ \( C' \) do
12: if \( d_F(\pi, \sigma \tau) \leq 3 \) then
13: \( C(\pi) \leftarrow C(\pi) \cup \{\sigma \tau\} \)
14: return \( C(\pi) \)

\text{generate_sequences}(time series } \sigma, \text{ integer } i, \text{ returned set } C' \)
// Stores in \( C' \) all possible time series which begin with \( \sigma \), have at most \( k \) vertices that belong to \( r_j' \cap G_w \), for \( j = i, \ldots, \ell \), and appear in them in the order of \( i \).
1: \( v_1, \ldots, v_\ell \leftarrow V(\sigma) \)
2: if \( |V(\sigma)| \leq k \) then
3: \( C' \leftarrow C' \cup \{\sigma\} \)
4: if \( |V(\sigma)| < k \) then
5: for each \( j = i, \ldots, \ell \) do
6: for each \( p \in r_j \cap G_w \) do
7: \( \sigma' \leftarrow \langle v_1, \ldots, v_\ell, p \rangle \)
8: \( \text{generate_sequences}(\sigma', j, C') \)

2.1.2 Query

query(time series τ)
1: \( \sigma_\tau \leftarrow \text{compute a } 2\text{-signature of } \tau \).
2: \( \sigma'_\tau \leftarrow \text{snap } \sigma_\tau \text{ to } G_w \)
3: if \( \exists \pi \in \Pi, \sigma'_\tau \in C(\pi) \) then
4: \( \text{report } \pi \) // check the bucket \( H(\sigma'_\tau) \)
5: else
6: \( \text{report } \text{"no"} \) // arbitrary \( \pi \) s.t. \( \sigma'_\tau \in C(\pi) \)
3 Improving the approximation factor to \((2 + \epsilon)\)

In this section, we extend the ideas developed in Section 2 to improve the approximation factor of our data structure from \((5 + \epsilon)\) to \((2 + \epsilon)\). The key to circumventing the large approximation factor resulting from the use of the triangle inequality seems to be a careful construction of matchings. For this we define the notion of a \(\delta\)-tight matching for two curves.

3.1 Tight matchings

Intuitively, a \(\delta\)-tight matching is a matching which attains a distance of at most \(\delta\) and matches as many pairs of points as possible at distance zero.

Full proofs of the lemmas of this section can be found in Section 3.2.

Definition 14 (\(\delta\)-tight matching). Given two curves \(\pi\) and \(\tau\), consider a monotone path \(\lambda\) through the parametric space of \(\pi\) and \(\tau\) consisting of two types of segments:

(i) a segment contained in the 0-free space (corresponding to identical subcurves of \(\pi\) and \(\tau\)),
(ii) a horizontal line segment contained in the \(\delta\)-free space (corresponding to a point on \(\pi\) and a subcurve on \(\tau\)).

If \(\lambda\) exists, we say \(\lambda\) is a tight matching of width \(\delta\) from \(\pi\) to \(\tau\).

Lemma 15. Let \(X = ab \subset \mathbb{R}\) be a line segment and let \(\tau : [0, 1] \rightarrow \mathbb{R}\) be a curve with \([a, b] \subseteq [\tau(0), \tau(1)]\). If \(d_F(X, \tau) \leq \delta\) then there exists a \(\delta\)-tight matching from \(X\) to \(\tau\).

Proof Sketch. We first construct a connected path in the \(\delta\)-free space of the two curves that only consists of sections of the 0-free space and horizontal line segments, but is not necessarily monotone. We do this by parametrizing the set that constitutes the 0-free space and connecting it by horizontal line segments. We obtain an \(x\)-monotone connected curve from \((0, 0)\) to \((1, 1)\) which lies inside the \(\delta\)-free space. We then show that this path can be iteratively “repaired” by replacing non-monotone sections of the path with horizontal segments, while maintaining the property that the path is contained inside the \(\delta\)-free space. After a finite number of iterations of this procedure we obtain a \(\delta\)-tight matching from \(X\) to \(\tau\). Figure 2 illustrates the process.

In the next lemma we combine tight matchings from a line segment to show an upper bound on the Fréchet distance. Using this lemma, we can show upper bounds on the distance that are stronger than bounds obtained by triangle inequality. Figure 3 illustrates the idea of the proof.

Lemma 16. Let \(X = ab \subset \mathbb{R}\) be a line segment and let \(\tau\) and \(\pi\) be curves with \([a, b] \subseteq [\tau(0), \tau(1)]\) and \([a, b] \subseteq [\pi(0), \pi(1)]\). If \(d_F(X, \tau) = \delta_1\) and \(d_F(X, \pi) = \delta_2\), then \(d_F(\tau, \pi) \leq \max(\delta_1, \delta_2)\).
Figure 3: Example of the path constructed in the proof of Lemma 16. The left figure shows a tight matching from $X$ to $\pi$. The middle figure shows a tight matching from $X$ to $\tau$. Diagonal edges of the 0-free space of these can be transferred to the diagram on the right, which is the free space diagram of $\pi$ and $\tau$. The final path results from connecting these diagonal segments using horizontal and vertical line segments.

**Lemma 17.** Let $\sigma_\pi$ be a $\delta$-signature of a time series $\pi$ and let $\sigma_\tau$ be a $2\delta$-signature of a time series $\tau$. Suppose that the first edge and the last edge of $\tau$ are both of length more than $4\delta$. If $d_F(\sigma_\pi,\sigma_\tau) \leq \delta$ then $d_F(\pi,\tau) \leq 2\delta$.

**Proof Sketch.** Let $\pi(p_1)\ldots,\pi(p_\ell)$ be the vertices of $\sigma_\pi$ and let $\tau(t_1),\ldots,\tau(t_\ell')$ be the vertices of $\sigma_\tau$. The properties of the signatures of Definition 6 together with Lemma 8 imply that a weakly monotonic matching of $\sigma_\pi$ with $\sigma_\tau$ which attains a distance of at most $\delta$ can be structured as follows: each edge is either entirely matched with an edge, or with a vertex.

This observation combined with the range property of signatures and the assumption that the first and the last edge of $\tau$ are of length greater than $4\delta$ allows us to focus on any pair of matched edges $\tau([t_j,t_{j+1}])$, $\pi([p_i,p_{i+1}])$, and show that $d_F(\pi[p_i,p_{i+1}],\tau[t_j,t_{j+1}]) \leq 2\delta$. To do that, we partition $\pi[p_i,p_{i+1}]$ into three subcurves $\pi[p_i,p_a],\pi[p_a,p_b],\pi[p_b,p_{i+1}]$ and $\tau[t_j,t_{j+1}]$ into three subcurves $\tau[t_j,t_a]$, $\tau[t_a,t_b]$, $\tau[t_b,t_{j+1}]$. The main property of this decomposition is that either $\pi[p_i,p_a]$ or $\tau[t_j,t_a]$ is a point and we can easily derive $d_F(\pi[p_i,p_a],\tau[t_j,t_a]) \leq 2\delta$ by greedily matching the other curve. The same holds for $\pi[p_b,p_{i+1}]$ and $\tau[t_b,t_{j+1}]$. Then, to prove that $d_F(\pi[p_a,p_b],\tau[t_a,t_b]) \leq 2\delta$ we rely on bounding $d_F(\pi[p_a,p_b],\pi[p_a][p_b])$, $d_F(\tau[t_a,t_b],\tau[t_a][t_b])$ and then applying Lemma 16. Edges which are matched with vertices are special cases which can be easily handled.

We conclude that each subcurve $\tau_{e_1}$ (or vertex) of $\tau$ corresponding to some edge $e_1$ (or vertex) of $\sigma_\pi$ which is matched with some edge $e_2$ (or vertex) in $\sigma_\tau$ by a matching of $\sigma_\pi$ with $\sigma_\tau$ which attains a distance of at most $\delta$, can be matched with the subcurve $\tau_{e_2}$ of $\tau$ which corresponds to $e_2$, such that $d_F(\tau_{e_1},\tau_{e_2}) \leq 2\delta$. Hence, $d_F(\pi,\tau) \leq 2\delta$.

**Lemma 18.** Let $\sigma_\pi$ be a $\delta$-signature of a time series $\pi$ and let $\sigma_\tau$ be a $2\delta$-signature of a time series $\tau$. Suppose that the first edge and the last edge of $\tau$ are both of length more than $4\delta$. For any $\delta' \leq \delta$, if $d_F(\pi,\tau) \leq \delta'$ then $d_F(\sigma_\pi,\sigma_\tau) \leq \delta'$.

**Proof Sketch.** Let $\pi(p_1)\ldots,\pi(p_\ell)$ be the vertices of $\sigma_\pi$ and let $\tau(t_1),\ldots,\tau(t_\ell')$ be the vertices of $\sigma_\tau$. The assumption on the first and last edge together with the properties of the signatures of Definition 6 imply that any two consecutive $2\delta$-ranges along $\sigma_\tau$ are disjoint. Moreover, we would like to use the property that the length of any edge of $\sigma_\pi$ is strictly longer than $2\delta$. This is not entirely true for the first and last edge, but we can show something similar exploiting the fact that the edge lengths of $\tau$ are long and that the Fréchet distance of $\sigma$ and $\tau$ is bounded by $\delta'$. As a result, we can
show that any edge $\overline{(p_i)p_{i+1}}$ can be matched with a segment $e_i := \tau(h_i)\tau(h_{i+1})$ lying in some edge of $\sigma$, such that for each $i \in [\ell - 1]$, it holds that $d_F(\overline{(p_i)p_{i+1}}, e_i) \leq \delta'$, which implies that $d_F(\sigma_\pi, \sigma_\tau) \leq \delta'$. \hfill \qed

3.2 Full Proofs of Section 3.1

Lemma 15. Let $X = \overline{ab} \subset \mathbb{R}$ be a line segment and let $\tau : [0, 1] \to \mathbb{R}$ be a curve with $[a, b] \subseteq [\tau(0), \tau(1)]$. If $d_F(X, \tau) \leq \delta$ then there exists a $\delta$-tight matching from $X$ to $\tau$.

Proof. Consider the $\delta$-free space of the two curves $X$ and $\tau$, which is a subset of $[0, 1]^2$. We adopt the convention that a point $(x, y) \in [0, 1]$ in this diagram corresponds to two points $X(y)$ and $\tau(x)$ (so $X$ corresponds to the vertical axis and $\tau$ corresponds to the horizontal axis). Let $0 = x_1 \leq \cdots \leq x_p = 1$ denote the parameter values at vertices of $\tau$. The $\delta$-free space is subdivided into cells $[0, 1] \times [x_i, x_{i+1}]$. We call the intersection of the $\delta$-free space with the vertical cell boundary at $x$-coordinate $x_i$ the free space interval at index $i$ and denote it with $[\ell_i, r_i]$. Consider the 0-free space inside this diagram, this is the set of points $(x, y) \in [0, 1]^2$ with $X(y) = \tau(x)$. This set forms a set of paths $\lambda_1, \ldots, \lambda_r$, for some $r \in \mathbb{N}$, which is $x$-monotone, since $X$ is a line segment. Therefore, we can parameterize this set by $x$. We concatenate any two $\lambda_i$ and $\lambda_{i+1}$ by adding a line segment between their endpoints. A connecting segment will be a horizontal line, either at $\ell_i = 0$ or at $r_i = 1$. This can easily be proved by contradiction (assume that $\lambda_i$ ends at 0 and $\lambda_{i+1}$ starts at 1, then the section of $\tau$ between those endpoints would have to be disconnected). In addition, we add line segments to connect $\lambda_1$ to $(0, 0)$ and to connect $\lambda_r$ to $(1, 1)$. We obtain a connected path $\lambda$ from $(0, 0)$ to $(1, 1)$, which lies inside the $\delta$-free space, but is not necessarily monotone in $y$. Figure 2 shows an example.

We now describe how to obtain a $\delta$-tight matching from $\lambda$ by repeatedly replacing sections of $\lambda$ with horizontal line segments, until $\lambda$ is monotone in both parameters, $x$ and $y$.

Assume $\lambda$ is not monotone. Then, there exists a horizontal line that properly intersects $\lambda$ in three different points. Consider a horizontal line at height $y$ with three distinct intersections at $(s_1, y)$, $(s_2, y)$, and $(s_3, y)$, such that

(i) the section of $\lambda$ between $s_1$ and $s_2$ lies completely below $y$

(ii) the section of $\lambda$ between $s_2$ and $s_3$ lies completely below $y$

There exist indices $i$ and $j$, such that $s_1 \leq \ell_i < t_j \leq s_3$ and such that $t_i$ is minimal and $t_j$ is maximal in this set of indices. Let $L$ be the line segment from $(s_1, y)$ to $(s_3, y)$. If $L$ is contained inside the $\delta$-free space, then we replace the corresponding section of $\lambda$ with $L$ and obtain monotonicity of $\lambda$ in the cell(s) $[0, 1] \times [x_i, x_j]$.

Otherwise, let $i_- \in [i, j]$ be the index that maximizes $\ell_{i_-}$ and let $i_+$ be the index in $[i, j]$ which minimizes $r_{i_+}$. (Recall that $[\ell_i, r_i]$ denotes the free space interval at index $i$). It must be that $y \notin [\ell_{i_-}, r_{i_+}]$, otherwise the line segment $L$ would be contained inside the $\delta$-free space.

Assume $y > r_{i_+}$ (the other case is symmetric and handled below). This case is illustrated in Figure 2. Let $y' = r_{i_+}$ and consider the intersections of $\lambda$ with the horizontal line at $y'$. It must be that there exist intersection points with $s_1', s_2', s_3'$ with $s_1' < s_2' < s_3'$ such that

(i) the section of $\lambda$ between $s_1'$ and $s_2'$ lies completely above $y'$

(ii) the section of $\lambda$ between $s_2'$ and $s_3'$ lies completely below $y'$

Let $L'$ be the line segment from $(s_1', y')$ to $(s_3', y')$. Since $d_F(X, \tau) \leq \delta$, it holds that $\ell_j \leq r_{i_+}$ for any $j \leq i_+$, otherwise there cannot be a monotone path in the $\delta$-free space. Therefore, $L'$ is contained in the $\delta$-free space and we can use it to shortcut $\lambda$ and obtain monotonicity of $\lambda$ in the cell(s) $[0, 1] \times [x_i, x_j]$.

Otherwise, we have $y < \ell_{i_-}$. We handle this case symmetrically. Let $y' = \ell_{i_-}$ and consider the intersections of $\lambda$ with the horizontal line at $y'$. It must be that there exist intersection points with $s_1', s_2', s_3'$ with $s_1' < s_2' < s_3' < s_3$, such that
(i) the section of \( \lambda \) between \( s'_1 \) and \( s'_3 \) lies completely above \( y' \)
(ii) the section of \( \lambda \) between \( s'_2 \) and \( s'_3 \) lies completely below \( y' \)

Let \( L' \) be the line segment from \( (s'_1, y') \) to \( (s'_3, y') \). Since \( d_F(X, \tau) \leq \delta \), it holds that \( \ell_{i-} \leq r_j \leq \) for any \( j \geq i \), otherwise there cannot be a monotone path in the \( \delta \)-free space. Therefore, \( L' \) is contained in the \( \delta \)-free space and we can use it to shortcut \( \lambda \) and obtain monotonicity of \( \lambda \) in the cell(s) \( [0,1] \times [x_i, x_j] \).

With each shortcutting step we obtain monotonicity of the path \( \lambda \) in at least one of the cells. Therefore, the process ends after a finite number of steps.

\[ \square \]

**Lemma 16.** Let \( X = \overline{ab} \subset \mathbb{R} \) be a line segment and let \( \tau \) and \( \pi \) be curves with \( [a,b] \subseteq [\tau(0), \tau(1)] \) and \( [a,b] \subseteq [\pi(0), \pi(1)] \). If \( d_F(X, \tau) = \delta_1 \) and \( d_F(X, \pi) = \delta_2 \), then \( d_F(\tau, \pi) \leq \max(\delta_1, \delta_2) \).

**Proof.** Let \( \delta = \max(\delta_1, \delta_2) \). By Lemma \([15]\) there exists a \( \delta \)-tight matching from \( X \) to \( \tau \) and another one from \( X \) to \( \pi \). We construct a monotone path in the \( \delta \)-free space of \( \tau \) and \( \pi \) from these two tight matchings. In particular, we first specify diagonal segments of the constructed path, which lie in the 0-free space, and then connect these segments with horizontal, resp., vertical segments. Let \( S \subset [0,1] \) be the finite set of parameter values of \( X \), which correspond to the horizontal segments of the tight matching from \( X \) to \( \tau \). Let \( Q \subset [0,1] \) be the finite set of parameters of the horizontal segments of the tight matching from \( X \) to \( \pi \). Let \( y_1 < \cdots < y_r \) be the sorted list of the values \( S \sqcup Q \) (without multiplicities). For any interval \( y_i, y_{i+1} \) in this list, there exists a diagonal segment in both tight matchings that covers the entire interval in the \( y \)-direction. That is, the tight matching matches \( X[y_i, y_{i+1}] \) to a subcurve on \( \tau \) and a subcurve on \( \pi \) that are identical. Let \( \tau[t, t'] \) and \( \pi[p, p'] \) be these subcurves. Let \( \lambda_t = (t, p)(t', p') \) be the corresponding diagonal segment of the \( \delta \)-free space of \( \tau \) and \( \pi \). Since the two subcurves are identical, \( \lambda_t \) is part of the 0-free space. We obtain a set of diagonal segments in the 0-free space, which we intend to connect to piecewise-linear path where every edge is of one of the three types: (i) a diagonal edge contained in the 0-free space, (ii) a horizontal edge, (iii) a vertical edge. For connecting two diagonal segments \( \lambda_i \) and \( \lambda_{i+1} \), there are three cases:

(i) \( y_{i+1} \in S \) and \( y_{i+1} \notin Q \): in this case \( \lambda_i \) and \( \lambda_{i+1} \) can be connected by a horizontal line segment.
(ii) \( y_{i+1} \notin S \) and \( y_{i+1} \in Q \): in this case \( \lambda_i \) and \( \lambda_{i+1} \) can be connected by a vertical line segment.
(iii) \( y_{i+1} \in S \) and \( y_{i+1} \in Q \): in this case \( \lambda_i \) and \( \lambda_{i+1} \) can be connected by a horizontal line segment followed by a vertical line segment.

From this, we obtain a monotone path in the \( \delta \)-free space of \( \pi \) and \( \tau \) from \( (0,0) \) to \( (1,1) \).

\[ \square \]

**Lemma 17.** Let \( \sigma_\pi \) be a \( \delta \)-signature of a time series \( \pi \) and let \( \sigma_\tau \) be a \( 2\delta \)-signature of a time series \( \tau \). Suppose that the first edge and the last edge of \( \tau \) are both of length more than \( 4\delta \). If \( d_F(\sigma_\pi, \sigma_\tau) \leq \delta \) then \( d_F(\pi, \tau) \leq 2\delta \).

**Proof.** Let \( \pi(p_1) \ldots, \pi(p_\ell) \) be the vertices of \( \sigma_\pi \) and let \( \tau(t_1), \ldots, \tau(t_{\ell'}) \) be the vertices of \( \sigma_\tau \). Let \( r_i := [\pi(p_i) - \delta, \pi(p_i) + \delta] \), for \( i \in [\ell] \) and let \( r'_i := [\tau(t_i) - \delta, \tau(t_i) + \delta] \), for \( i \in [\ell'] \). By Lemma \([8]\) since \( \sigma_\pi \) is a \( \delta \)-signature of \( \sigma_\pi \), \( \sigma_\pi \) has a vertex in each range \( r_i \) and such that these vertices appear in the order of \( i \). Similarly, since \( \sigma_\tau \) is a \( 2\delta \)-signature and hence a \( \delta \)-signature of \( \sigma_\tau \), \( \sigma_\tau \) has a vertex in each range \( r'_i \) and such that these vertices appear in the order of \( i \). Hence, a weakly monotonic matching of \( \sigma_\pi \) with \( \sigma_\tau \) which attains a distance of at most \( \delta \) can be structured as follows: each edge is either entirely matched with a vertex, or entirely matched with an edge.

First, consider an edge \( \tau(t_j) \tau(t_{j+1}) \), for some \( 1 \leq j \leq \ell' \). We will show now that \( d_F(\pi[p_i, p_{i+1}], \tau[t_j, t_{j+1}]) \leq 2\delta \). By the minimum edge length property and the assumption that the first and the last edge of \( \tau \) is of length
more than $4\delta$, we conclude that the two edges must have the same direction. Assume w.l.o.g. that
\( \pi(p_i) < \pi(p_{i+1}) \) and \( \tau(t_j) < \tau(t_{j+1}) \).

If \( \pi(p_i) < \tau(t_j) \) then we set \( p_a = \max\{t \in [p_i, p_{i+1}) \mid \pi(t) = \tau(t_j)\} \), \( t_a = t_j \). If \( \pi(p_i) \geq \tau(t_j) \) then we set \( t_a = \min\{t \in [t_j, t_{j+1}) \mid \tau(t) = \tau(p_i)\} \), \( p_a = p_i \). Similarly, if \( \pi(p_{i+1}) > \tau(t_{j+1}) \) then we set \( t_b = \max\{t \in [t_j, t_{j+1}) \mid \tau(t) = \tau(p_{i+1})\} \), \( t_b = t_{j+1} \). If \( \pi(p_{i+1}) \leq \tau(t_{j+1}) \) then we set \( t_b = \max\{t \in [t_j, t_{j+1}) \mid \tau(t) = \pi(p_{i+1})\} \), \( p_b = p_{i+1} \). Let \( a = \pi(p_a) = \tau(t_a) \) and \( b = \pi(p_b) = \tau(t_b) \).

If \( \pi(p_i) < \tau(t_j) \) then by the range property, the direction preserving property of \( \pi, \tau \), and the fact that \( |\pi(p_i) - \tau(t_j)| \leq \delta \), we conclude that \( \pi[p_a, p_b] \subseteq [\tau(t_j) - 2\delta, \tau(t_j) + 2\delta] \). Hence \( d_F(\pi[p_a, p_b], \tau(t_j, t_b)) \leq 2\delta \). If \( \pi(p_i) \geq \tau(t_j) \) then by the range property of \( \pi, \tau \), the assumption that first and the last edges are of length larger than \( 4\delta \), and the fact that \( |\pi(p_i) - \tau(t_j)| \leq \delta \), \( \tau[t_j, t_a] \subseteq [\pi(p_i) - \delta, \pi(p_i)] \). Hence, \( d_F(\pi[p_i, p_a], \tau(t_j, t_b)) \leq 2\delta \). Applying the same arguments symmetrically, we conclude that \( d_F(\pi[p_b, p_{i+1}], \tau(t_b, t_{j+1})) \leq 2\delta \).

For \( \pi[p_a, p_b], \tau[t_a, t_b] \) we invoke Lemma \ref{lem:tau-range} with \( X = ab \). Notice that \( X \) is a \( \delta \)-signature of \( \pi[p_a, p_b] \) because \( \pi[p_a, p_b] \subseteq [a - \delta, b + \delta] \), by the triangular inequality and the range property of \( \pi \), and because the rest of the signature properties are still satisfied when we restrict \( \pi \) to \([p_a, p_b]\). So by Lemma \ref{lem:tau-range} we have \( d_F(\pi[p_a, p_b], X) \leq \delta \). Similarly, \( d_F(\pi[t_a, t_b], X) \leq \delta \), because we have that \( \tau[t_a, t_b] \subseteq [a - \delta, b + \delta] \) and by the range property of \( \tau \) and the assumption that the first and the last edge are of length larger than \( 4\delta \), and because observing that \( X \) is a \( 2\delta \)-signature of \( \tau[t_a, t_b] \) allows us to apply Lemma \ref{lem:tau-range}. Hence, by Lemma \ref{lem:tau-range} \( d_F(\pi[p_a, p_b], \tau[t_a, t_b]) \leq 2\delta \).

By the minimum edge length property, the only edges that can be matched entirely with a vertex are the first and the last edge of \( \sigma_\pi \). Suppose that \( \pi(p_1) \pi(p_2) \) is entirely matched with \( \tau(t_1) \) and assume w.l.o.g. \( \pi(p_1) \leq \pi(p_2) \). Then, \( |\pi(p_1) - \tau(t_1)| \leq \delta \) and \( |\pi(p_2) - \tau(t_1)| \leq \delta \). By the range property of \( \pi, \pi[p_1, p_2] \subseteq [\pi(p_1) - \delta, \pi(p_2)] \). By the triangular inequality, \( [\tau(t_1) - 2\delta, \tau(t_1) + 2\delta] \) covers \( \pi[p_1, p_2] \). The case of the last edge of \( \pi \) is symmetric.

We have shown that each subcurve \( \pi_{e_1} \) (or vertex) of \( \pi \) corresponding to some edge \( e_1 \) (or vertex) of \( \sigma_\pi \) which is matched with some edge \( e_2 \) (or vertex) in \( \sigma_\tau \) by a matching of \( \sigma_\pi \) with \( \sigma_\tau \) which attains a distance of at most \( \delta \), can be matched with the subcurve \( \tau_{e_2} \) of \( \tau \) which corresponds to \( e_2 \), such that \( d_F(\pi_{e_1}, \tau_{e_2}) \leq 2\delta \). Hence, \( d_F(\pi, \tau) \leq 2\delta \).

\hfill \Box

Lemma 18. Let \( \sigma_\pi \) be a \( \delta \)-signature of a time series \( \pi \) and let \( \sigma_\tau \) be a \( 2\delta \)-signature of a time series \( \tau \). Suppose that the first edge and the last edge of \( \sigma_\pi \) are both of length more than \( 4\delta \). For any \( \delta' \leq \delta \), if \( d_F(\pi, \tau) \leq \delta' \) then \( d_F(\sigma_\pi, \sigma_\tau) \leq \delta' \).

\textbf{Proof.} Let \( \pi(p_1), \ldots, \pi(p_k) \) be the vertices of \( \sigma_\pi \) and let \( \tau(t_1), \ldots, \tau(t_k) \) be the vertices of \( \sigma_\tau \). Let \( \pi(p_1) \pi(p_{i+1}) \) be an edge of \( \sigma_\pi \). Assume wlog that \( \pi(p_i) < \pi(p_{i+1}) \). Now let \( h_i, h_{i+1} \in [0,1] \) be parameters such that \( \tau[h_i, h_{i+1}] \) is matched with \( \pi[p_i, p_{i+1}] \) by the optimal matching between \( \pi, \tau \). Since \( d_F(\pi, \tau) \leq \delta' \), it holds that \( |\pi(p_i) - \tau(h_i)| \leq \delta' \) and \( |\pi(p_{i+1}) - \tau(h_{i+1})| \leq \delta' \).

First, consider the case \( i \in [2, \ell - 2] \cap \mathbb{Z} \). By the minimum edge length property we know that \( |\pi(p_i) - \pi(p_{i+1})| > 2\delta \) which implies that \( \tau(h_i) < \tau(h_{i+1}) \). We refer to edges going to the opposite direction of that of \( \pi[p_i] \pi(p_{i+1}) \) as backward edges. The subcurve \( \pi[p_i, p_{i+1}] \) can contain backward edges of length at most \( 2\delta \), because the existence of longer backward edges would refute the direction preserving property. This means that there is no backward edge in \( \tau[h_i, h_{i+1}] \) of length greater than \( 4\delta \) because otherwise we would have \( d_F(\pi, \tau) > \delta \). Hence, \( \tau(h_i) \) and \( \tau(h_{i+1}) \) belong to the same edge of \( \sigma_\tau \), and \( d_F(\pi(p_i) \pi(p_{i+1}), \tau(h_i) \tau(h_{i+1})) > \delta' \), because \( |\pi(p_i) - \tau(h_i)| \leq \delta' \) and \( |\pi(p_{i+1}) - \tau(h_{i+1})| \leq \delta' \).

Now, we focus on the case \( i \in \{1, \ell - 1\} \). In that case the minimum edge length property is weaker than in the case \( i \in [2, \ell - 2] \cap \mathbb{Z} \). The first edge of \( \sigma_\tau \) is of length more than \( 4\delta \), but
we cannot claim that \(|\pi(p_i) - \pi(p_{i+1})| > 2\delta\). Instead, we assume that \(|\pi(p_i) - \pi(p_{i+1})| \in (\delta, 2\delta]\) and we will exploit the fact that the first edge (resp. the last) of \(\tau\) is long. We focus on the case \(i = 1\), since the case \(i = \ell - 1\) is symmetric. By the assumption that \(d_F(\pi, \tau) \leq \delta'\), we have that \(|\pi(p_1) - \tau(t_1)| = |\pi(0) - \tau(0)| \leq \delta'\). The range property implies that the first edge of \(\tau\) is entirely contained in \(\tau(t_1)\tau(t_2)\), which implies that \(|\tau(t_1) - \tau(t_2)| > \delta\). Since \(|\tau(t_1) - \tau(p_1)| \leq \delta\), \(|\tau(t_2) - \tau(p_2)| \leq \delta|, \(|\pi(p_1) - \pi(p_2)| \leq 2\delta\) and \(|\tau(t_1) - \tau(t_2)| > 4\delta\), it must hold that \(\tau(h_2) \in \tau(t_1)\tau(t_2)\).

Hence, \(\tau(h_1)\) and \(\tau(h_2)\) belong to the same edge of \(\sigma_\tau\), and \(d_F(\pi(p_1)\pi(p_2), \tau(h_1)\tau(h_2)) \leq \delta'\), because \(|\pi(p_1) - \tau(h_1)| \leq \delta'\) and \(|\pi(p_2) - \tau(h_2)| \leq \delta'\).

We have shown that each edge \(\pi(p_1)\pi(p_{i+1})\) can be matched with a segment \(e_i := \overline{\tau(h_i)\tau(h_{i+1})}\) lying in some edge of \(\sigma_\tau\) such that for each \(i \in [\ell - 1]\), \(d_F(\pi(p_i)\pi(p_{i+1}), e_i) \leq \delta'\), which implies that \(d_F(\sigma_\pi, \sigma_\tau) \leq \delta'\).

\[\square\]

3.3 The data structure

The signatures fail to successfully capture the structure of a curve in the beginning and in the end. For that reason, we introduce the notions of the prefix and the suffix of a curve, which are maximal subcurves with short edges.

**Definition 19** (\(\delta\)-prefix). Let \(\tau\) be a curve \([0, 1] \mapsto \mathbb{R}\) and let \(0 = t_1, \ldots, t_m = 1\) be the parameters corresponding to vertices of \(\tau\). The \(\delta\)-prefix of \(\tau\) is the maximal sequence \(\tau(t_1), \tau(t_2), \ldots, \tau(t_\ell)\) for which it holds: for any \(i \in [\ell - 1]\), \(|\tau(t_i) - \tau(t_{i+1})| \leq \delta\). If \(|\tau(t_1) - \tau(t_2)| \geq \delta\) then the \(\delta\)-prefix is \(\tau(t_1)\).

**Definition 20** (\(\delta\)-suffix). Let \(\tau\) be a curve \([0, 1] \mapsto \mathbb{R}\) and let \(0 = t_1, \ldots, t_m = 1\) be the parameters corresponding to vertices of \(\tau\). The \(\delta\)-suffix of \(\tau\) is the maximal sequence \(\tau(t_\ell), \tau(t_{\ell+1}), \ldots, \tau(t_m)\) for which it holds: for any \(i \in [\ell, m - 1] \cap \mathbb{Z}\), \(|\tau(t_i) - \tau(t_{i+1})| \leq \delta\). If \(|\tau(t_m) - \tau(t_{m-1})| \geq \delta\) then the \(\delta\)-suffix is \(\tau(t_m)\).

The data structure. The input consists of a set \(\Pi\) of \(n\) curves in \(\mathbb{R}\), the distance threshold \(r > 0\), and the approximation error \(\epsilon > 0\). As before, we assume that the distance threshold is \(r := 1 - \epsilon/10\) (for other values of \(r\) we scale the input curves during preprocessing). To discretize the query space, we use the regular grid \(\mathcal{G}_w := \{i \cdot w \mid i \in \mathbb{Z}\}\), where \(w := \epsilon/30\). For each input curve \(\pi \in \Pi\), we first compute sets \(C_1, C_2\), where \(C_1\) contains all possible curves with at most \(k\) vertices from \(\mathcal{G}_w\), its first vertex within distance \(1 + w\) from the first vertex of \(\pi\) and edge lengths at most \(4\), and \(C_2\) contains all possible curves with at most \(k\) vertices from \(\mathcal{G}_w\), its last vertex within distance \(1 + w\) from the last vertex of \(\pi\) and edge lengths at most \(4\). Then for each pair \(\sigma_1 \in C_1, \sigma_2 \in C_2\), which satisfies \(|\sigma_1| + |\sigma_2| \leq k\), we first partition \(\pi\) into three parts \(\pi[0, p_a], \pi[p_a, p_b], \pi[p_b, 1]\), where \(\pi(p_a)\) is the last point in \(\pi\) such that \(d_F(\pi[0, p_a], \sigma_1) \leq 1 - 2w\) and \(\pi(p_b)\) is the first point in \(\pi\) such that \(d_F(\pi[p_b, 1], \sigma_2) \leq 1 - 2w\). Next, we compute the 1-signature \(\sigma_\pi\) of \(\pi[p_a, p_b]\), with vertices \(V(\sigma_\pi) = v_1, \ldots, v_\ell\) and we define ranges \(r'_i := v_i \pm (2 + w)\). We use these ranges \(r'_i\) to construct \(C'\), the set of all possible 2-signatures of at most \(k - |\sigma_1| - |\sigma_2| + 2\) vertices that belong to \(r'_i \cap \mathcal{G}_w\), for \(i = 1, \ldots, \ell\), appear in them in the order of \(i\), and have as their first vertex the last vertex of \(\sigma_1\), and as their last vertex the first vertex of \(\sigma_2\). At last, we compute \(C(\pi) = \{(\sigma_1, \sigma_\tau, \sigma_2) \in C_1 \times C' \times C_2 \mid d_F(\sigma_1, \pi[0, p_a]) \leq 1, d_F(\sigma_2, \pi[p_b, 1]) \leq 1, d_F(\sigma_\pi, \sigma_\tau) \leq 1\}\). The intuition is that \(\sigma_1\) corresponds to a 4-prefix, \(\sigma_2\) corresponds to a 4-suffix and \(\sigma_\pi\) is the signature of the subcurve lying between \(\sigma_1\) and \(\sigma_2\), for some approximate near neighbor \(\tau\), modulo snapping to \(\mathcal{G}_w\).

The complete pseudocode for this procedure is diverted to Section 3.4 (see **generate candidates**(\(\pi\))). We store \(C(\pi)\) in a hashable \(\mathcal{H}\) as follows: for each \((\sigma_1, \sigma_\tau, \sigma_2) \in C(\pi)\), we use as key \(\kappa(\sigma_1, \sigma_\tau, \sigma_2)\)
the sequence of vertices $V(\sigma_1), V(\sigma_\tau), V(\sigma_2)$. Let $H(\sigma_1, \sigma_\tau, \sigma_2)$ be the bucket with key $\kappa(\sigma_1, \sigma_\tau, \sigma_2)$. Then, for each $(\sigma_1, \sigma_\tau, \sigma_2) \in C(\pi)$, we store in $H(\sigma_1, \sigma_\tau, \sigma_2)$ a pointer to $\pi$. For $H$, we assume perfect hashing, as in the data structure of Section 2.2.

The query algorithm For a query curve $\tau$ with $V(\tau) = \tau(t_1), \ldots, \tau(t_k)$, the algorithm query($\tau$) snaps $\tau$ to $G_w$, to obtain $\tau'$. Then, it computes a triplet $(\sigma_1, \sigma_\tau', \sigma_2)$, where $\sigma_1$ is a 4-prefix of $\tau'$, $\sigma_2$ is 4-suffix of $\tau'$ and $\sigma_\tau'$ is the curve obtained by snapping the 2-signature of the subcurve $\tau[t_{|\sigma_1|}, t_{k-|\sigma_2|+1}]$ to $G_w$. A lookup in $H$, with key $\kappa(\sigma_1, \sigma_\tau', \sigma_2)$, suffices to report a valid answer: if there is a bucket $H(\sigma_1, \sigma_\tau', \sigma_2)$ then one arbitrary curve stored in it will be reported, otherwise the algorithm returns “no”. The complete pseudocode for the query algorithm is diverted to Section 3.4 (see query($\tau$)).

3.4 Pseudocode of the improved result

3.4.1 Preprocessing

```plaintext
generate_candidates(time series $\pi$)
1: $\pi(p_1), \ldots, \pi(p_m) \leftarrow V(\pi)$
2: $r_1 \leftarrow [\pi(p_1) - 1 - w, \pi(p_1) + 1 + w]$
3: $r_m \leftarrow [\pi(p_m) - 1 - w, \pi(p_m) + 1 + w]$
4: $C_1 \leftarrow \emptyset$
5: for each $p \in r_1 \cap G_w$ do
6: generate_bounded_curves($p, C_1$)
7: $C_2 \leftarrow \emptyset$
8: for each $p \in r_m \cap G_w$ do
9: generate_bounded_curves($p, C_2$)
10: $C(\pi) \leftarrow \emptyset$
11: for each $\sigma_1 \in C_1$, $\sigma_2 \in C_2$ such that $|\sigma_1| + |\sigma_2| \leq k$ do
12: if $\{p \in [0, 1] | d_F(\sigma_1, \pi[p, 1]) \leq 1\} \neq \emptyset$ then
13: $p_a \leftarrow \max\{p \in [0, 1] | d_F(\sigma_1, \pi[p, 1]) \leq 1 - 2w\}$.
14: else
15: $p_a \leftarrow \perp$
16: if $\{p \in [0, 1] | d_F(\sigma_2, \pi[p, 1]) \leq 1\} \neq \emptyset$ then
17: $p_b \leftarrow \min\{p \in [0, 1] | d_F(\sigma_2, \pi[p, 1]) \leq 1 - 2w\}$
18: else
19: $p_b \leftarrow \perp$
20: $\sigma_\pi \leftarrow 1$-signature of $\pi[p_a,p_b]$, with $V(\sigma_\pi) = v_1, \ldots, v_\ell$
21: if $\ell \leq k$ and $p_a \neq \perp$ and $p_b \neq \perp$ then
22: for each $i = 1, \ldots, \ell$ do
23: $r_i' \leftarrow [v_i - 2 - w, v_i + 2 + w]$
24: $u_1 \leftarrow$ the last vertex of $\sigma_1$
25: $u_2 \leftarrow$ the first vertex of $\sigma_2$
26: $C' \leftarrow \{(u_1, u_2)\}$
27: for each $j = 1, \ldots, \ell$ do
28: for each $p \in r_j' \cap G_w$ do
29: generate_sequences2($u_1, p, j, u_2, C'$)
30: for each $\sigma_\tau \in C'$ do
31: if $d_F(\sigma_1, \pi[0, p_a]) \leq 1$ and $d_F(\sigma_2, \pi[p_b, 1]) \leq 1$ and $d_F(\sigma_\pi, \sigma_\tau) \leq 1$ then
32: $C(\pi) \leftarrow C(\pi) \cup \{(\sigma_1, \sigma_\tau, \sigma_2)\}$
33: return $C(\pi)$
```
we know that \( F \) would lead to a contradiction. Since \( |\mathcal{N}(s)| < k \) then
\[
\text{for each } p \in [v_i - 4, v_i + 4] \cap G_w \text{ do }
\]
\[
\sigma' \leftarrow \langle v_1, \ldots, v_i, p \rangle
\]
\[\text{generate_bound\_curves}(\sigma, C')\]

3.4.2 Query

query(time series \( \tau \) with \( \mathcal{N}(\tau) = \tau(t_1), \ldots, \tau(t_k) \))
\[
\tau' \leftarrow \text{snap } \tau \text{ to } G_w
\]
\[
\sigma_1 \leftarrow 4\text{-prefix of } \tau'
\]
\[
\sigma_2 \leftarrow 4\text{-suffix of } \tau'
\]
\[
\sigma' \leftarrow \text{compute a 2-signature of } \tau[t_{|\sigma_1|}, t_{k-|\sigma_2|+1}] .
\]
\[
\sigma' \leftarrow \text{snap } \sigma' \text{ to } G_w
\]
\[
\text{if } \exists \pi \in \Pi, (\sigma_1, \sigma_2, \sigma_3) \in C(\pi) \text{ then } \]
\[
\text{report } \pi \quad \text{// arbitrary } \pi \text{ s.t. } (\sigma_1, \sigma_2, \sigma_3) \in C(\pi)
\]
\[
\text{else}
\]
\[
\text{report } \text{“no”}.
\]

3.5 Analysis

We now prove correctness of the query algorithm. We start by proving a technical lemma.

Lemma 21. Let \( \pi \) and \( \tau \) be two curves in \( \mathbb{R} \) such that \( d_F(\pi, \tau) \leq \delta \). Assume that \( \tau \) has at least one edge of length more than \( 4\delta \). Let \( \tau(q_a) \) be the last vertex of the \( \delta' \)-prefix of \( \tau \) and let \( \tau(q_b) \) be the first vertex of the \( \delta' \)-suffix of \( \tau \), where \( \delta' \geq 4\delta \). Let \( \pi(p_a) \) be the last point in \( \pi \) such that \( d_F(\pi[0, p_a], \tau[0, q_a]) \leq \delta \) and let \( \pi(p_b) \) be the first point in \( \pi \) such that \( d_F(\pi[p_b, 1], \tau[0, q_a]) \leq \delta \). Then \( d_F(\pi[p_a, p_b], \tau(q_a, q_b]) \leq \delta \).

Proof. Any optimal matching for \( \pi \), \( \tau \) matches \( \pi(p_a) \) with some point \( \tau(q'_a) \) such that \( q_a \leq q'_a \), because otherwise \( \tau(p_a) \) would not be the last point in \( \pi \) such that \( d_F(\pi[0, p_a], \tau[0, q_a]) \leq \delta \) which would lead to a contradiction. Since \( d_F(\pi[0, p_a], \tau[0, q_a]) \leq \delta \), we know that \( |\pi(p_a) - \tau(q_a)| \leq \delta \).

We also know that \( \tau(q_a) \) is the first endpoint in an edge of length at least \( 4\delta \). Hence, \( \tau[q_a, q'_a] \) is a...
subsegment of an edge of \( \tau \), and can be matched with \( \pi(p_a) \) since both endpoints are within distance \( \delta \) from it. The rest of the points in \( \pi[p_a, p_b] \), \( \tau[q_a, q_b] \) are matched according to the optimal matching of \( \pi, \tau \), until we reach \( \pi(p_b) \). If we had reached \( \tau(q_b) \) before \( \pi(p_b) \), this would mean that \( \pi(p_b) \) is not the first point in \( \pi \) such that \( d_F(\pi[p_b, 1], \tau[q_b, 1]) \leq \delta \). Hence \( \pi(p_b) \) is matched with some point \( \tau(q'_b) \) with \( q'_b \leq q_b \), and \( \tau[q'_b, q_b] \) is a subsegment in an edge of \( \tau \) which means that it can be matched with \( \pi(p_b) \) since both of its endpoints are within distance \( \delta \) from \( \pi(p_b) \).

**Lemma 22.** For any query curve \( \tau \) of complexity \( k \), \textbf{query}(\( \tau \)) reports as follows: if there exists \( \pi \in \Pi \) such that \( d_F(\pi, \tau) \leq 1 - \epsilon/10 \) then it returns \( \pi' \in \Pi \) such that \( d_F(\pi', \tau) \leq 2 + \epsilon/30 \), if for any \( \pi \in \Pi \), \( d_F(\pi', \tau) > 2 + \epsilon/30 \), then it returns “no”.

**Proof.** Let \( \pi \) be any input curve in \( \Pi \). First, suppose that \( d_F(\pi, \tau) \leq 1 - 3w \), where \( w = \frac{\epsilon}{30} \), which is the side length the grid. By the triangular inequality,

\[
d_F(\pi, \tau') \leq d_F(\pi, \tau) + d_F(\tau, \tau') \leq 1 - 2w,
\]

where \( \tau' \) denotes the curve resulting by snapping the vertices of \( \tau \) to \( G_w \). Let \( \sigma_1 \) be the the 4-prefix of \( \tau' \) and let \( \sigma_2 \) be the 4-suffix of \( \tau' \). We define \( a := t_{|\sigma_1|} \) and \( b := t_{k-|\sigma_2|+1} \). Let \( \pi(p_a) \) be the last point in \( \pi \) such that \( d_F(\pi[0, p_a], \sigma_1) \leq 1 - 2w \) and let \( \pi(p_b) \) be the first point in \( \pi \) such that \( d_F(\pi[p_b, 1], \sigma_2) \leq 1 - 2w \). Let \( \tau'_{ab} \) be the curve resulting by snapping the vertices of \( \tau[q_a, q_b] \) to \( G_w \). Then, by Lemma [21] \( d_F(\pi[p_a, p_b], \tau[q_a, q_b]) \leq 1 - 2w \) and by the triangular inequality

\[
d_F(\pi[p_a, p_b], \tau[q_a, q_b]) \leq d_F(\pi[p_a, p_b], \tau'_{ab}) + d_F(\tau'_{ab}, \tau[q_a, q_b]) \leq 1 - w.
\]

Now, by Lemma [18] \( d_F(\sigma_1, \sigma_2) \leq 1 - w \), where \( \sigma_1 \) is a 1-signature of \( \pi[p_a, p_b] \) and \( \sigma_2 \) is a 2-signature of \( \tau[q_a, q_b] \). During preprocessing, we definitely consider \( \sigma_1 \) since we enumerate all possible curves which have their first vertex within distance \( 1 + w \) from the first vertex of \( \pi \), the length of each one of their edges is at most 4, and their vertices are in \( G_w \). Similarly, we definitely consider \( \sigma_2 \) since we enumerate all possible curves which have their last vertex within distance \( 1 + w \) from the last vertex of \( \pi \), he length of each one of their edges is at most 4 and their vertices are in \( G_w \). Let \( v_1, \ldots, v_\ell \) be the vertices of \( \sigma_1 \). By Corollary [11] we know that the union of the intervals \( [v_i - (2 + w), v_i + (2 + w)] \), \( i = 1, \ldots, \ell \), cover the vertices of \( \sigma_2 \). Hence, the union of the intervals \( r_i' = [v_i - (2 + w), v_i + (2 + w)] \) cover the vertices of \( \sigma_2' \), the curve obtained by snapping the vertices of \( \sigma_2 \) to \( G_w \), which means that we also consider \( \sigma_2' \) in \( \mathcal{C}' \). By the triangular inequality,

\[
d_F(\sigma_1, \sigma_2') \leq d_F(\sigma_1, \tau') + d_F(\tau', \sigma_2) \leq 1,
\]

and the answer will be correctly computed in the preprocessing phase.

Now assume that \( d_F(\sigma_1, \sigma_2') \leq 1 \), \( d_F(\pi[0, p_a], \sigma_1) \leq 1 \) and \( d_F(\pi[p_b, 1], \sigma_2) \leq 1 \). By Lemma [17] \( d_F(\pi[p_a, p_b], \tau[q_a, q_b]) \leq 2 \). Then, \( d_F(\pi, \sigma_1 \oplus \tau[q_a, q_b] \oplus \sigma_2) \leq 2 \) and by the triangular inequality,

\[
d_F(\pi, \tau) \leq d_F(\pi, \sigma_1 \oplus \tau[q_a, q_b] \oplus \sigma_2) + d_F(\tau, \sigma_1 \oplus \tau[q_a, q_b] \oplus \sigma_2) \leq 2 + w.
\]

We conclude that the approximation factor is \( \frac{2+w}{1-3w} < 2 + \epsilon \), for \( w = \epsilon/30 \) and \( \epsilon \in [0,7] \).
Proof. The data structure is described above. By Lemma 11 the data structure solves the $(2+\epsilon)$-ANN with distance threshold $1-3\epsilon$. We can solve for any distance threshold by scaling the ambient space. It remains to prove our complexity bounds. The algorithm generate_candidates first computes sets $C_1$ and $C_2$. We bound the cardinality of those sets as follows:

$$|C_1| \leq \sum_{i=1}^{k} \left( \frac{88}{\epsilon} + 2 \right)^i \leq \sum_{i=1}^{k} \left( \frac{22}{\epsilon} \right)^i = \frac{(22/\epsilon)^{k+1} - 1}{22/\epsilon - 1} \leq \frac{22}{15} \cdot \left( \frac{22}{\epsilon} \right)^k = O \left( \frac{1}{\epsilon} \right)^k .$$

Similarly, $|C_2| = O \left( \frac{1}{\epsilon} \right)^k$.

We proceed by analyzing the steps in the main loop of generate_candidates. To compute $p_a$, we compute the free-space diagram of $\sigma_1$ and $\pi$, as in [8], in $O(mk)$ time. We can then find the edge of $\pi$ with the largest index that contains a point which is reachable in the free-space diagram by a monotone path, and the focusing on the two edges realizing that point, we can compute $p_a$. Similarly for $p_b$. The signature of a curve of complexity at most $m$ can be computed in time $O(m)$ by the result of [17]. We can now upper bound the number of triplets $(\sigma_1, \sigma_2, \sigma_2)$ which will be produced by generate_candidates. Assuming that $\sigma_1$ and $\sigma_2$ are fixed, we have $|C'| = O \left( \frac{1}{\epsilon} \right)^{k-|\sigma_1|-|\sigma_2|}$ similarly to the proof of Theorem [13] and because the first and the last vertex are fixed. In total, over all iterations, the number of all triplets $(\sigma_1, \sigma_2, \sigma_2)$ produced by the algorithm is upper bounded by

$$\sum_{t_1+t_2\leq k} O \left( \frac{1}{\epsilon} \right)^{t_1} \cdot O \left( \frac{1}{\epsilon} \right)^{t_2} \cdot O \left( \frac{1}{\epsilon} \right)^{k-t_1-t_2} = \sum_{t_1+t_2\leq k} O \left( \frac{1}{\epsilon} \right)^k = O(k^2) \cdot O \left( \frac{1}{\epsilon} \right)^k .$$

Hence, the preprocessing time is $O(n \cdot k^2) \cdot O \left( \frac{1}{\epsilon} \right)^k + O(nmk^3)$ and the space needed is $O(n \cdot k^2) \cdot O \left( \frac{1}{\epsilon} \right)^k + O(nk^3)$ assuming that we only store indices for the curves and not the actual curves (this would just require an additional $O(mn)$ of space). The query time is $O(k)$ because we need $O(k)$ time to compute curves $\tau'$, $\sigma_1$, $\sigma_2$, $\sigma_\tau$, $\sigma'_\tau$ and then we rely on the guarantees of perfect hashing to probe $H$ in $O(k)$ time.

4 Distance oracles and asymmetric communication

In this section, we study lower bounds on the cell-probe-complexity of distance oracles for the Fréchet distance and the discrete Fréchet distance. We focus on the decision version of the problem. In particular, we say a distance oracle with input curve $\pi$, threshold $r > 0$, and approximation factor $c > 1$, is a data structure which reports as follows: for any query $\tau$, if $d_F(\pi, \tau) \leq r$ then it outputs “yes”, else if $d_F(\pi, \tau) \geq cr$ then it outputs “no” and otherwise both answers are acceptable. This can be viewed as a special case of the $c$-ANN problem. To show our lower bounds, we employ a technique first introduced by Miltersen [31], which implies that lower bounds for communication problems can be translated into lower bounds for cell-probe data structures. The following communication problem is known as the lopsided (or asymmetric) disjointness problem.

Definition 23 ((k, U)-Disjointness). Alice receives a set $S$, of size $k$, from a universe $[U] = \{1 \ldots U\}$, and Bob receives $T \subset [U]$ of size $m \leq U$. They need to decide whether $T \cap S = \emptyset$.

A randomized $[a, b]$-protocol for a communication problem is a protocol in which Alice sends $a$ bits, Bob sends $b$ bits, and the error probability is bounded away from $1/2$. The following result by Pătraşcu gives a lower bound on the randomized asymmetric communication complexity of the $(k, U)$-Disjointness problem.
Theorem 24 (Theorem 1.4 [32]). Assume Alice receives a set \( S \), \( |S| = k \) and Bob receives a set \( T \), \( |T| = m \), both sets coming from a universe of size \( U \), such that \( k \leq m \leq U \). In any randomized, two-sided error communication protocol deciding disjointness of \( S \) and \( T \), either Alice sends at least \( \delta k \log \left( \frac{k}{\delta} \right) \) bits or Bob sends at least \( \Omega \left( k \left( \frac{U}{k} \right)^{1-C \delta} \right) \) bits, for any \( \delta > 0 \), and \( C = 1799 \).

We now define the distance threshold estimation problem (DTEP), where two parties must determine whether two curves are near or far. This is basically the communication version of our data structure problem (for \( n = 1 \)).

**Definition 25 ((\( k,U \))-Fréchet DTEP).** Given parameters \( c \geq 1 \), \( r > 0 \), Alice receives a curve \( \tau \) of complexity \( k \) in \( \mathbb{R}^d \), Bob receives a curve \( \pi \) of complexity \( m \leq U \) in \( \mathbb{R}^d \). If \( d_F(\pi,\tau) \leq r \) then they must output “yes”. If \( d_F(\pi,\tau) > cr \) then they must output “no”. Otherwise, both answers are acceptable.

Similarly, we define the \((k,U)\)-Discrete Fréchet DTEP.

**Definition 26 ((\( k,U \))-Discrete Fréchet DTEP).** Given parameters \( c \geq 1 \), \( r > 0 \), Alice receives a curve \( \tau \) of complexity \( k \) in \( \mathbb{R}^d \), Bob receives a curve \( \pi \) of complexity \( m \leq U \) in \( \mathbb{R}^d \). If \( d_{DF}(\pi,\tau) \leq r \) then they must output “yes”. If \( d_{DF}(\pi,\tau) > cr \) then they must output “no”. Otherwise, both answers are acceptable.

4.1 A cell-probe lower bound for the Fréchet distance

We first reduce the lopsided set disjointness problem to the problem of approximating the Fréchet distance of two curves in \( \mathbb{R} \). A similar reduction appears in [30], which however works for curves in \( \mathbb{R}^2 \) and it is used to lower bound the sketching complexity.

First consider an instance of the set disjointness problem: Alice has a set \( A = \{\alpha_1, \ldots, \alpha_k\} \subset [U] \) and Bob has a set \( B = \{\beta_1, \ldots, \beta_m\} \subset [U] \), where \( U \) is the size of the universe. We now describe our main gadgets which will be used to define one curve of complexity \( O(k) \) for \( A \) and one curve of complexity \( O(U - m) \) for \( B \).

For each \( i \in [U] \):

- If \( i \in A \) then \( x_i := 4i + 4 \), \( x_{i+1} := 4i \),
- If \( i \notin A \) then \( x_i := 4i \), \( x_{i+1} := 4i \),
- If \( i \in B \) then \( y_i := 4i \), \( y_{i+1} := 4i \),
- If \( i \notin B \) then \( y_i := 4i + 3 \), \( y_{i+1} := 4i + 1 \),

We now define \( \tilde{x} := (0, x_1, \ldots, x_{2U}, 4U + 5) \) and \( \tilde{y} := (0, y_1, \ldots, y_{2U}, 4U + 5) \). Notice that the number of vertices of \( \tilde{x} \) is \( 2k + 2 \), and the number of vertices of \( \tilde{y} \) is \( 2(U - m) + 2 \). The arclength of any of \( \tilde{x} \), \( \tilde{y} \) is at most \( 12U + 2 \).

**Theorem 27.** If \( A \cap B = \emptyset \) then \( d_F(\tilde{x}, \tilde{y}) \leq 1 \). If \( A \cap B \neq \emptyset \) then \( d_F(\tilde{x}, \tilde{y}) \geq 2 \).

**Proof.** If there is no \( i \in A \cap B \) then there is a monotonic matching which implies \( d_F(\tilde{x}, \tilde{y}) \leq 1 \). For any \( i \in [U] \), let \( \tilde{x}_i := (4i, x_i, x_{i+1}, 4i + 4) \) and \( \tilde{y}_i := (4i, y_i, y_{i+1}, 4i + 4) \). To show that, it is sufficient to show that for any \( i \in [U] \), \( d_F(\tilde{x}_i, \tilde{y}_i) \leq 1 \). If \( i \notin A \) and \( i \in B \) then the two subcurves are just straight line segments and their distance is 0. If \( i \notin A \) and \( i \notin B \) then \( \tilde{x}_i \) is a line segment and \( \tilde{y}_i \) consists of three line segments forming a zig-zag. The matching works as follows: it first matches the interval \([4i, 4i + 2]\) of \( \tilde{x}_i \) with the interval \([4i, 4i + 2]\) of \( \tilde{y}_i \) by moving in both curves at the same speed, then it stops moving in \( \tilde{x}_i \), while it moves from \( 4i + 2 \) to \( y_i \) and then to \( y_{i+1} \) and then to
4i + 2 in $\tilde{y}_i$. The matching continues by moving in the two remaining subsegments at the same speed. This is a matching that attains $d_F(\tilde{x}_i, \tilde{y}_i) \leq 1$, because $4i + 2$ is within distance 1 from any of $y_i, y_{i+1}$. Finally if $i \in A$ and $i \notin B$ then the matching works as follows: it first matches $[0, x_i]$ with $[0, y_i]$, then it matches $(x_i, x_{i+1})$ with $(y_i, y_{i+1}]$, and it finally matches $(x_{i+1}, 4i + 4)$ with $(y_{i+1}, 4i_4]$. Since it basically matches pairs of line segments having endpoints at distance at most 1 from each other, the Fréchet distance is again at most 1.

Suppose now that there is an $i$ such that $i \in A$ and $i \in B$. Let $v$ be the first appearance of the point $4i + 4$ in $\tilde{x}$, and let $u$ be the second appearance of the point $4i$ in $\tilde{x}$. Assume that $d_F(\tilde{x}, \tilde{y}) = \delta < 2$. Then, $v$ is matched with some point $z$ in $\tilde{y}$ which lies within distance $\delta$. However, there is no point in $\tilde{y}$ which lies within distance $\delta$ from $u$, and appears in $\tilde{y}$ after $z$. This implies that $\delta \geq 2$, because the matching required by the definition of the Fréchet distance has to be monotonic.

We use a technique of obtaining cell-probe lower bounds first introduced by Miltersen [31]. For a static data structure problem with input $p \in P$, which computes $f(p,q)$ for any query $q \in Q$, we consider the communication problem, where Alice gets $q \in Q$, Bob gets $p \in P$, and they must determine $f(q,p)$. If there is a solution to the data structure problem with parameters $s, w$ and $t$, then there is a protocol for the communication problem, with $2t$ rounds of communication, where Alice sends $\lceil \log s \rceil$ bits in each of her messages and Bob sends $w$ bits in each of his messages. The protocol is a simple simulation of the assumed data structure where Alice sends indices to memory cells and Bob responds with the cell content. Theorem 24 combined with Theorem 27 implies lower bounds for cell-probe Fréchet distance oracles.

**Theorem 28.** Consider any Fréchet distance oracle with approximation factor $2 - \gamma$, for any $\gamma \in (0, 1]$, in the cell-probe model, which supports curves in $\mathbb{R}$ as follows: it stores any polygonal curve of arclength at most $L$, for $L \geq 6$, it supports queries of arclength at most $L$ and complexity $k$, where $k \leq L/6$, and it achieves performance parameters $t, w, s$. There exist

$$w_0 = \Omega \left( \frac{k}{t} \left( \frac{L}{k} \right)^{1-\epsilon} \right), \quad s_0 = 2^{\Omega \left( \frac{k \log(L/k)}{t} \right)}$$

such that if $w < w_0$ then $s > s_0$, for any constant $\epsilon > 0$.

**Proof.** By Theorem 27 if there exists a randomized $[a,b]$-protocol for the communication problem, in which, Alice gets any curve $x$ of complexity $2k + 2$ and arclength at most $12U + 2$, Bob gets any curve $y$ of complexity $2(U - m) + 2$ of arclength at most $12U + 2$ and they can decide whether $d_F(x, y) \leq 1$ or $d_F(x, y) \geq 2$, then they can solve the $(k, U)$-Disjointness problem.

By Theorem 24 for any $\delta > 0$, there exists $b_0 = \Omega \left( k \left( \frac{U}{k} \right)^{1-1799\delta} \right)$, such that a randomized $[a,b]$-protocol for $(k, U)$-Disjointness, for any $k \leq m \leq U$, requires either $a \geq \delta k \log \left( \frac{U}{k} \right)$ or $b \geq b_0$.

Hence, for any $\delta > 0$, and any $k \leq m \leq U$, if there exists a randomized $[a,b]$-protocol for the $(2k + 2, 2(U - m) + 2)$-Fréchet DTEP for any curves of arclength at most $12U + 2$, then either $a \geq \delta k \log \left( \frac{U}{k} \right)$ or $b \geq b_0$.

The simulation argument implies that if there exists a cell-probe data structure with parameters $t, w, s$ for curves in $\mathbb{R}$, with query complexity $2k + 2$, and arclength at most $12U + 2$, then there exists a randomized $[2t \log s, 2tw]$-protocol for the Fréchet DTEP. Hence, it should be either that $2t \log s \geq \delta k \log \left( \frac{U}{k} \right)$ or $2tw \geq b_0$. There exists a $w_0 = \Omega \left( k \left( \frac{U}{k} \right)^{1-1799\delta} \right)$ such that if $w < w_0 \leq b_0$, then $s \geq \frac{2\delta k \log(U/k)}{2t}$. The theorem is now implied by setting $\delta = \epsilon/1799$, $L = 12U + 2$ and rescaling $k \leftarrow 2k + 2$.  \qed
We want to construct point sequences, one for each input set of Alice and Bob, such that there exists a common element in Alice’s and Bob’s input sets, if and only if the discrete Fréchet distance of the two sequences is less or equal than a given threshold. Our reduction takes some of its main ideas from [11]. Our gadgets use the following points (see Fig. 4):

\[ a_0 = (−1.61, 0.5), \alpha_1 = (−1.61, −0.5) \alpha'_0 = (1.61, 0.5), \alpha'_1 = (1.61, −0.5), \beta_0 = (−0.61, 0.5), \beta_1 = (−0.61, −0.5), \beta'_0 = (0.61, 0.5), \beta'_1 = (0.61, −0.5), \]
\[ s = (0, 0), w = (−0.75, 0), w' = (0.75, 0), y_1 = (−0.25, 0), y_2 = (0.25, 0), x_1 = (−0.25, −1), x_2 = (0.25, −1). \]

Let \( D = \lceil \log U \rceil \), where \( U \) is the size of the universe in the \((k, U)\)-Disjointness instance. We further assume that \( D \) is even for convenience. We treat elements of the universe as binary vectors: Alice’s set corresponds to a set \( \{a_1, \ldots, a_k\} \), where each \( a_i \in \{0, 1\}^D \), and Bob’s set corresponds to a set \( \{b_1, \ldots, b_k\} \), where each \( b_i \in \{0, 1\}^D \). For each vector \( a_i \in \{0, 1\}^D \) we have a gadget \( A_i \) which is a sequence of points constructed as follows: for each odd coordinate \( j \) we either put \( a_0 \) or \( \alpha_1 \) depending on whether \( (a_i)_j \) is 0 or 1 and for each even coordinate \( j \) we either put \( \alpha'_0 \) or \( \alpha'_1 \) depending on whether \( (a_i)_j \) is 0 or 1. For example, for the vector \((0, 1, 0, 0)\) (assuming that it belongs to Alice) we create \( a_0, a'_1, a_0, a'_0 \). Similarly for each vector \( b_i \) we have a gadget \( B_i \) which is a sequence...
of points constructed as follows: for each odd coordinate $j$ we either put $\beta_0$ or $\beta_1$ depending on whether $(b_i)_j$ is 0 or 1 and for each even coordinate $j$ we either put $\beta_0'$ or $\beta_1'$ depending on whether $(b_i)_j$ is 0 or 1. Given two sequences $P = p_1, \ldots, p_m$ and $Q = q_1, \ldots, q_m$, we say that a traversal $T = (i_1, j_1), \ldots, (i_m, j_m)$ is parallel if for all $k = 1, \ldots, m$ we have $i_k = j_k = k$.

**Lemma 30.** Let $a_i, b_j \in \{0, 1\}^D$. If $a_i = b_j$ then $d_{df}(A_i, B_j) \leq 1$. If $a_i \neq b_j$ then $d_{df}(A_i, B_j) \geq \sqrt{2}$.

Moreover, for any non-parallel traversal $T$, we have $d_T(A_i, B_j) \geq 2$.

**Proof.** If $a_i = b_j$ then the parallel traversal gives $d_{df}(A_i, B_j) \leq 1$. If $a_i \neq b_j$ then $d_{df}(A_i, B_j) \geq \sqrt{2}$.

To see that notice that $\|\beta_0 - \alpha_1\|_2 = \|\beta_1 - \alpha_0\|_2 = \|\beta_0' - \alpha_1'\|_2 = \|\beta_1' - \alpha_0'\|_2 = \sqrt{2}$. Furthermore, for each $z, w \in \{0, 1\}$ $\|z - w\|_2 > 2$ and $\|z' - w\|_2 > 2$.

We define $W = \bigcup_{i=1}^{Dm/2} (w \circ w')$. Given $a_1, \ldots, a_k$ and $b_1, \ldots, b_m$, we construct two point sequences as follows:

$$P = W \circ x_1 \circ \bigcup_{i=1}^{m/2} (s \circ B_i) \circ s \circ x_2 \circ W,$$

$$Q = \bigcup_{i=1}^{k} (y_1 \circ A_i \circ y_2).$$

**Lemma 31.** Let $a_1, \ldots, a_k \in \{0, 1\}^D$ and $b_1, \ldots, b_m \in \{0, 1\}^D$. If there exist $i, j$ such that $a_i = b_j$ then $d_{df}(P, Q) \leq 1$.

**Proof.** We assume that there exist $i^* \in [k], j^* \in [m]$ such that $a_{i^*} = b_{j^*}$. We describe one traversal $T$ which achieves $d_T(P, Q) \leq 1$ and hence $d_{df}(P, Q) \leq 1$.

1. The first $2D(i^* - 1)$ points of $W$ are matched with the first $(i^* - 1)(D + 2)$ points of $q$. In particular, for each $i = 1, \ldots, i^* - 1$: (i) $w$ is matched with $y_1$, (ii) $T$ proceeds in parallel for $\bigcup_{j=1}^{D/2} (w \circ w')$ and $A_i$, (iii) $w'$ is matched with $y_2$.

2. $T$ remains in $y_1$ and it matches it with the rest of $W$. Then, $x_1$ is matched with $y_1$.

3. $y_1$ is matched with all points in $\bigcup_{j=1}^{D-1} (s \circ B_i)$.

4. $T$ proceeds in parallel for $A_{i^*}$ and $B_{j^*}$.

5. $T$ remains in $y_2$ and proceeds only in $p$ until it reaches $W$.

6. The first $2D(m - i^*)$ points of $W$ are matched with the rest of $Q$ as in step 1.

7. $T$ remains in $y_2$ (the last point of $q$) and it proceeds in $P$ until the end.

Points $w, w'$ are within distance 1 from any of $y_1, y_2, \alpha_0, \alpha_1, \alpha_0', \alpha_1'$. Points $y_1$ are within distance 1 from $x_1, s$ and any of $\beta_0, \beta_1, \beta_0', \beta_1'$. By Lemma 30 $d_{df}(A_{i^*}, B_{j^*}) \leq 1$. Then $y_2$ is within distance 1 from $x_2, s$ and any of $\beta_0, \beta_1, \beta_0', \beta_1'$.

$$P = \quad w \quad x_1 \quad \cdots \quad p_{y_1} \quad \cdots \quad p_{y_2} \quad \cdots \quad x_2 \quad w$$

$$Q = \quad \cdots \quad y_1' \quad \cdots \quad a' \quad \cdots \quad y_2' \quad \cdots$$

Figure 5: $x_1$ is matched with $y_1'$, $p_{y_1'}$ is the last point in $p$ which is matched with $y_1'$ and $p_{y_2'}$ is the first point in $p$ which is matched with $y_2'$.
Lemma 32. Let $a_1, \ldots, a_k \in \{0, 1\}^D$ and $b_1, \ldots, b_m \in \{0, 1\}^D$. If there are no $i, j$ such that $a_i = b_j$, then $d_{DF}(P, Q) \geq 1.11$.

Proof. Consider some traversal $T$. We assume that $T$ matches $x_1$ with some $y_1$ and no other point of $Q$. Likewise, $x_2$ is matched with some $y_2$ and no other point of $Q$. If these assumptions do not hold and $x_1$ or $x_2$ are matched with some other point then $d_T(P, Q) \geq 1.11$. Furthermore we assume that each $s$ is matched with either a $y_1$ or a $y_2$, because otherwise $d_T(P, Q) \geq 1.68$. Now let $y_1^i$ be the $i$th appearance of point $y_1$ in $Q$ and assume that $x_1$ is matched with it. Let $p_{y_1^i}$ be the last point in $P$ which is matched with $y_1^i$ and let $p_{y_2^i}$ be the first point in $P$ which is matched with $y_2^i$ (see Fig. 5). We consider all cases for $p_{y_1^i}$:

- If $p_{y_1^i}$ is $x_1$ then the first appearance of $s$ is matched with one of $\alpha_0, \alpha_1, \alpha'_0, \alpha'_1$ and hence $d_{DF}(P, Q) \geq 1.68$.
- If $p_{y_1^i}$ is the $j$th appearance of $s$ then:

  - If $j = k + 1$ then the first point of $A_i$ is matched with either $s$ or $x_2$. Hence, the distance is at least 1.68.
  - If $j < k + 1$, then by our initial assumption that $s$ is always matched with either a $y_1$ or a $y_2$. $p_{y_1^j}$ cannot appear after the $(j + 1)$th appearance of $s$. Hence, a subsequence of $B_j$ is compared to $A_i$. By Lemma 30 this implies that $d_{DF}(P, Q) \geq \sqrt{2}$.
- If $p_{y_1^i}$ is a point of some gadget $B_j$ then the same reasoning implies that a subsequence of $B_j$ is compared to $A_i$. By Lemma 30 this implies that $d_{DF}(P, Q) \geq \sqrt{2}$.
- If $p_{y_1^i} \in \{w, w'\}$ then this means that $x_2$ is matched with $y_1^i$ because of monotonicity of the matching, but then the distance is at least 1.11.

We conclude that if there are no $i, j$ such that $a_i = b_j$, then $d_{DF}(P, Q) \geq 1.11$. 

Theorem 33. Suppose that there exists a randomized $[a, b]$-protocol for the discrete Fréchet DTEP with approximation factor $c < 1.11$ where Alice receives a sequence of $k(2 + \lceil \log(U) \rceil)$ points in $X \subset \mathbb{R}^2$ and Bob receives a sequence of $3\lceil \log(U) \rceil + m + 3$ points in $X$, where $X$ is a bounded domain and $|X| \leq 15$. Then there exists a randomized $[a, b]$-protocol for the $(k, U)$-Disjointness problem in a universe $U$, where Alice receives a set $S \subset U$ of size $k$ and Bob receives a set $T \subseteq U$ of size $m$.

Proof. First Alice and Bob convert their inputs to their binary representation. Alice uses her binary vectors $a_1, \ldots, a_k$ and constructs a sequence of points $Q = \bigcup_{i=1}^{k} (y_i \circ A_i \circ y_2)$, as described above. Similarly, Bob uses his binary vectors $b_1, \ldots, b_m$ and constructs $P = W \circ x_1 \circ \bigcup_{i=1}^{m} (s \circ B_i \circ s \circ x_2 \circ W)$. Then, Alice and Bob run the assumed $[a, b]$-protocol which allows them to determine whether $d_{DF}(P, Q) \leq 1$ or $d_{DF}(P, Q) \geq 1.11$. If $d_{DF}(P, Q) \leq 1$ then the answer to the $(k, U)$-Disjointness instance is “yes” and if $d_{DF}(P, Q) \geq 1.11$ then the answer is “no”. Lemmas 31 and 32 imply that in either case the answer is correct.

Theorem 34. Consider any discrete Fréchet distance oracle in the cell-probe model which supports point sequences from bounded domains in $\mathbb{R}^2$, as follows: for any $k \leq m \leq U$, it stores any point
sequence of length $m$, it supports queries of length $k$, and it achieves performance parameters $t$, $w$, $s$, and approximation factor $c < 1.11$. There exist

$$w_0 = \Omega \left( \frac{k}{t \log m} \cdot \left( \frac{U}{k} \right)^{1-\epsilon} \right),$$

$$s_0 = 2^{\Omega \left( \frac{k \log(U/k)}{t \log m} \right)} ,$$

such that if $w < w_0$, then $s \geq s_0$, for any constant $\epsilon > 0$.

**Proof.** By Theorem 33, for sufficiently large $k' = \mathcal{O}(k \log U)$ and $m' = \mathcal{O}(m \log U)$, there exists a bounded domain $X \subset \mathbb{R}^2$, for which if there exists a randomized $[a, b]$-protocol for the discrete Fréchet DTEP with approximation factor $c < 1.11$, Alice’s input length equal to $k'$, Bob’s input length equal to $m'$, then there exists a randomized $[a, b]$-protocol for $(k, U)$-Disjointness, where Alice receives a set $S \subseteq [U]$ and Bob receives a set $T \subseteq [U]$ of size $m$, with $k \leq m \leq U$.

Now consider the following randomized $[a, b]$-protocol. First, Alice and Bob use public random coins to map all elements of $U$ to random bit strings of dimension $D = 2 \log(mk)$. By a union bound over at most $mk$ different elements of $U$, distinct elements in $T \cup S$ will be mapped to distinct bit strings with probability at least $1 - (mk)^{-1}$. Then, Alice and Bob use the protocol of Theorem 33 to solve $(k, U)$-Disjointness in a universe of size $m^{\mathcal{O}(1)}$. Hence, for sufficiently large $k'' = \Theta(k \log m)$ and $m'' = \Theta(m \log m)$, there exists a bounded domain $X \subset \mathbb{R}^2$, for which if there exists a randomized $[a, b]$-protocol for the discrete Fréchet DTEP with approximation factor $c < 1.11$, Alice’s input length equal to $k''$, Bob’s input length equal to $m''$, then there exists a randomized $[a, b]$-protocol for $(k, U)$-Disjointness in an arbitrary universe $[U]$, where Alice receives a set $S \subseteq [U]$ and Bob receives a set $T \subseteq [U]$ of size $m$, with $k \leq m \leq U$.

By Theorem 24, for any $\delta > 0$, a randomized $[a, b]$-protocol for $(k, U)$-Disjointness, for any $m \leq U$, where $U$ is the size of the universe, requires either $a \geq \delta k \log \left( \frac{U}{k} \right)$ or $b \geq b_0$, where $b_0 = \Omega \left( k \left( \frac{U}{k} \right)^{1-1799\delta} \right)$. Hence, for any $\delta > 0$, and any $k$, $m$, such that $k \leq m$, if there exists a randomized $[a, b]$-protocol for the discrete Fréchet DTEP with the above-mentioned input parameters, then either $a \geq \delta k \log \left( \frac{U}{k} \right)$ or $b \geq b_0$.

The simulation argument implies that if there exists a cell-probe discrete Fréchet distance oracle with parameters $t$, $w$, $s$ for point sequences of size $k''$ and $m''$, for points in $D$, then there exists a randomized $[2t \log s, 2tw]$-protocol for the discrete Fréchet DTEP. Hence, it should be that either $2t \log s \geq \delta k \log \left( \frac{U}{k} \right)$ or $2tw \geq b_0$. In other words, if $w < b_0$, then $s \geq 2^{\frac{\delta k \log(U/k)}{2t}}$. Rescaling for $k'' = \Theta(k \log m)$ and $m'' = \Theta(m \log m)$ implies that there exists

$$w_0 = \Omega \left( \frac{k''}{t \log m} \cdot \left( \frac{U \log m}{k''} \right)^{1-1799\delta} \right),$$

$$s_0 = 2^{\Omega \left( \frac{\delta k'' \log(U/k)}{t \log m} \right)} = 2^{\Omega \left( \frac{\delta k'' \log(U/k)}{t \log m} \right)} ,$$

such that if $w < w_0$, then $s \geq s_0$. The theorem is now implied by just renaming variables $k''$, $m''$ and setting $\delta = \epsilon/1799$.

\[ \square \]

### 4.4 High dimension

The reduction in the previous section uses point sequences in the plane. We now describe a second reduction to show a dependency on the ambient dimension $d$ of the point sequences in case $d$ is sufficiently high. For all $i \in [U], e_i \in \mathbb{R}^d$ denotes the vector of the standard basis, i.e. the vector
with all elements equal to 0 except the $i$-th coordinate which is 1. We use the following points in $\mathbb{R}^{U+2}$:

$$w = (1, 1, 0, \ldots, 0), x_1 = (1, -1, 0, \ldots, 0), s = (0, 0, 0, \ldots, 0), \tilde{b}_i = (0, 0, e_i),$$

$$x_2 = (-1, 1, 0, \ldots, 0), y_1 = (1, 0, 0, \ldots, 0), \tilde{a}_i = (1, 1, e_i), y_2 = (0, 1, 0, \ldots, 0)$$

Given $S = \{s_1, \ldots, s_k\}$, $T = \{t_1, \ldots, t_m\}$ as in Definition 23 we construct the following point sequences:

$$P = w \circ x_1 \circ \bigcirc_{i=1}^{m} (s \circ \tilde{b}_i) \circ s \circ x_2 \circ w,$$

$$Q = \bigcirc_{i=1}^{k} (y_1 \circ \tilde{a}_s \circ y_2).$$

Notice that $P$ is a point sequence of length $2m + 5$ and $Q$ is a point sequence of length $3k$. All points lie in $\mathbb{R}^{U+2}$. Point $w$ serves as a skipping gadget since it is near to any point of $Q$, and points $s, x_1, x_2, y_1, y_2$ are needed for synchronization: $x_1$ is close to $y_1$ but no other point in $Q$, $x_2$ is close to $y_2$ but no other point in $Q$, and $s$ is close to both $y_1$ and $y_2$ but no other point in $Q$. Our analysis is very similar to the one of Section 4.4. A new key component is the use of random projections, and in particular the random projection by Achlioptas [1] to reduce the dimension.

**Lemma 35.** If $S \cap T \neq \emptyset$ then $d_{dF}(P, Q) \leq \sqrt{2}$.

**Proof.** Let $i^*, j^*$ such that $s_i^* = t_j^* \in T \cap S$. We describe a traversal which achieves distance $\sqrt{2}$:

1. $w$ is matched with all points of $q$ before $y_1 \circ \tilde{a}_s \circ y_2$
2. $x_1$ is matched with $y_1$
3. $y_1$ is matched with all points of $p$ before $\tilde{b}_t$
4. $\tilde{a}_s$ is matched with $\tilde{b}_t$
5. $y_2$ is matched with the rest of $P$
6. $w$ is matched with the rest of $Q$

Only the following distances appear in the above matching:

$$\|w - y_1\|_2, \|w - y_2\|_2, \|x_1 - y_1\|_2, \|s - y_1\|_2, \|\tilde{a}_s - \tilde{b}_t\|_2, \|s - y_2\|_2, \|x_2 - y_2\|_2,$$

$$\{\|w - \tilde{a}_s\|_2 \mid i \neq i^*\}, \{\|y_1 - \tilde{b}_t\|_2 \mid j < j^*\}, \{\|y_2 - \tilde{b}_t\|_2 \mid j > j^*\}$$

and all of them are at most $\sqrt{2}$. \hfill \Box

Figure 6: $x_1$ is matched with $y^i_1$, $p_{y^i_1}$ is the last point in $p$ which is matched with $y^i_1$ and $p_{y^i_2}$ is the first point in $p$ which is matched with $y^i_2$.

**Lemma 36.** If $S \cap T = \emptyset$ then $d_{dF}(P, Q) \geq \sqrt{3}$.  

25
Proof. Consider the optimal traversal for $P$ and $Q$. We assume that $x_1$ is matched with some $y_1$ and no other point of $Q$. Likewise, $x_2$ is matched with some $y_2$ and no other point of $Q$. If these assumptions do not hold and $x_1$ or $x_2$ are matched with some other point then $d_{df}(P, Q) \geq \sqrt{5}$. Furthermore we assume that each $s$ is matched with either a $y_1$ or a $y_2$, because otherwise $d_{df}(P, Q) \geq \sqrt{3}$.

Now let $y_1^i$ be the $i$th appearance of point $y_1$ in $Q$ and assume that $x_1$ is matched with it. Now let $p_{y_1}^i$ be the last point in $P$ which is matched with $y_1^i$ and let $p_{y_2}^j$ be the first point in $P$ which is matched with $y_2^j$ (see Fig. 6). We consider all cases for $p_{y_1}^i$:

- If $p_{y_1}^i$ is $x_1$ then at least one of the following must happen:
  - the first appearance of $s$ is matched with $a_{s_i}$ and hence $d_{df}(P, Q) \geq \sqrt{3}$,
  - $x_1$ is matched with $\tilde{a}_{s_i}$ and hence $d_{df}(P, Q) \geq \sqrt{3}$.

- If $p_{y_1}^i$ is the $j$th appearance of $s$ then:
  - If $j = k + 1$ then $\tilde{a}_{s_i}$ is matched with either $s$ or $x_2$ (or both). Hence, the distance is at least $\sqrt{3}$.
  - If $j < k + 1$, then by our initial assumption that $s$ is always matched with either a $y_1$ or a $y_2$, $p_{y_2}^j$ cannot appear after the $(j + 1)$th appearance of $s$. Hence, $\tilde{b}_j$ is matched with $\tilde{a}_i$ (because $s$ is assumed not to be matched with $\tilde{a}_{s_i}$). This implies that $d_{df}(P, Q) \geq 2$.

- If $p_{y_1}^i$ is $x_2$ then one of the aforementioned assumptions is not satisfied and hence $d_{df}(P, Q) \geq \sqrt{3}$.

- If $p_{y_1}^i$ is some point $\tilde{b}_{t_j}$ then $\tilde{a}_{s_i}$ is either matched with $\tilde{b}_{t_j}$ or with $s$. Hence, $d_{df}(P, Q) \geq \sqrt{3}$.

- If $p_{y_1}^i$ is $w$ then this means that $x_2$ is matched with $y_1^i$ because of monotonicity of the matching, but then the distance is at least $\sqrt{3}$.

We conclude that if $T \cap S = \emptyset$, then $d_{df}(P, Q) \geq \sqrt{3}$. \hfill \qed

Both point sequences $P$ and $Q$ consist of points in $\{-1, 0, 1\}^U$. In order to reduce the dimension, we will use the following (slightly rephrased) result by Achlioptas.

**Theorem 37** (Theorem 1.1 [1]). Let $P$ be an arbitrary set of $n$ points in $\mathbb{R}^d$. Given $\epsilon, \beta > 0$, let

$$d_0 = \frac{4 + 2\beta}{\epsilon^2/2 - \epsilon^3/3} \log n.$$  

For integer $d \geq d_0$, let $R$ be a $d' \times d$ random matrix with each $R(i, j)$ being an independent random variable following the uniform distribution in $\{-1, 1\}$. With probability at least $1 = n^{-\beta}$, for all $u, v \in P$:

$$\left\| \frac{1}{\sqrt{d'}} Ru - \frac{1}{\sqrt{d'}} Rv \right\|_2 \in (1 \pm \epsilon)\|u - v\|_2.$$ 

**Theorem 38.** Suppose that there exists a randomized $[a, b]$-protocol for the discrete Fréchet DTEP with approximation factor $c < \sqrt{3}/2$ where Alice receives a sequence of $3k$ points in $([-3, 3] \cap \mathbb{Z})^{\Theta(\log m)}$ and Bob receives a sequence of $2m + 5$ points in $([-3, 3] \cap \mathbb{Z})^{\Theta(\log m)}$. Then there exists a randomized $[a, b]$-protocol for the $(k, U)$-Disjointness problem in a universe $[U]$, where Alice receives a set $S \subseteq [U]$ of size $k$ and Bob receives a set $T \subseteq [U]$ of size $m$.  

26
Proof. Alice constructs a sequence \( Q \) of \( 3k \) points as described above and similarly Bob constructs a sequence \( P \) of \( 2m + 5 \) points. Let \( S \) be the set of all points in \( P,Q \). Alice and Bob use a source of public random coins to construct the same Johnson Lindenstrauss randomized mapping. In particular, we use Theorem 37. Let \( R \) be a \( d' \times d \) matrix with each element \( R(i,j) \) chosen uniformly at random from \( \{-1,1\} \) and let \( f : \mathbb{R}^d \rightarrow \mathbb{R}^{d'} \) be the function which maps any vector \( v \in \mathbb{R}^{d} \) to \( Rv \). Alice and Bob sample \( f(\cdot) \) and project their points to dimension \( d' = \Theta(\log(m+k)) = \Theta(\log m) \). With high probability, for any two points \( x, y \in S \) we have

\[
\| f(x) - f(y) \|_2^2 \in \left[ 0.99 \cdot d' \cdot \| x - y \|_2^2, 1.01 \cdot d' \cdot \| x - y \|_2^2 \right].
\]

Each element of vector \( f(x) \) is produced by an inner product of a vector of \( d \) random signs and a vector of at least \( d - 3 \) zeros and at most \( 3 \) elements from \( \{-1,1\} \). Hence, \( \| f(x) \|_\infty \leq 3 \) and moreover \( f(x) \in \mathbb{Z}^{d'} \). Let \( f(P) \) and \( f(Q) \) be the two point sequences after randomly projecting the points. By Lemmas 35 and 36 we get that if \( T \cap S \neq \emptyset \) then \( d_{dF}(f(P), f(Q)) \leq \sqrt{2.02 \cdot d'} \) and if \( T \cap S = \emptyset \) then \( d_{dF}(f(P), f(Q)) \geq \sqrt{2.97d'} \).

Hence Alice and Bob can now use the assumed protocol for computing the discrete Fréchet distance and decide whether \( T \cap S \neq \emptyset \) or \( T \cap S = \emptyset \). □

**Theorem 39.** There exists \( d_0 = \Theta(\log m) \), such that the following holds. Consider any discrete Fréchet distance oracle in the cell-probe model which supports point sequences in \( \mathbb{R}^d \), \( d \geq d_0 \), as follows: for any \( k \leq m \leq U \), it stores any sequence of length \( m \), it supports queries of length \( k \), and it achieves performance parameters \( t, w, s \), and approximation factor \( c < \sqrt{3/2} \). There exist

\[
w_0 = \Omega \left( \frac{k}{t} \left( \frac{U}{k} \right)^{1-\epsilon} \right), \quad s_0 = 2^{\Omega \left( \frac{k \log(U/k)}{t} \right)}
\]

such that if \( w < w_0 \) then \( s \geq s_0 \), for any constant \( \epsilon > 0 \).

Proof. By Theorem 38, there exists a set \( X \subset \mathbb{R}^{d_0} \), for which if there exists a randomized \([a,b]-\)protocol for the discrete Fréchet DTEP with approximation factor \( c < \sqrt{3/2} \), Alice’s input length equal to \( k' \), Bob’s input length equal to \( m' \), then there exists a randomized \([a,b]-\)protocol for \((k,U)\)-Disjointness in an arbitrary universe \([U]\), where Alice receives a set \( S \subseteq [U] \) and Bob receives a set \( T \subseteq [U] \) of size \( m \), with \( k \leq m \leq U \). By Theorem 24, for any \( \delta > 0 \), a randomized \([a,b]-\)protocol for \((k,U)\)-Disjointness, for any \( m \leq U \), where \( U \) is the size of the universe, requires either \( a \geq \delta k \log \left( \frac{U}{k} \right) \) or \( b \geq b_0 \), where \( b_0 = \Omega \left( \frac{k}{t} \left( \frac{U}{k} \right)^{1-1799} \right) \). Hence, for any \( \delta > 0 \), and any \( k, m \), such that \( k \leq m \), if there exists a randomized \([a,b]-\)protocol for the discrete Fréchet DTEP with the abovementioned input parameters, then either \( a \geq \delta k \log \left( \frac{U}{k} \right) \) or \( b \geq b_0 \).

The simulation argument implies that if there exists a cell-probe data structure with parameters \( t, w, s \) for point sequences of size \( k \) and \( m \), for points in \( \mathbb{R}^d \), then there exists a randomized \([2t \log s, 2tw]-\)protocol for the discrete Fréchet DTEP. Hence it should be either that \( 2t \log s \geq \delta k \log \left( \frac{U}{k} \right) \) or \( 2tw \geq b_0 \). There exists a \( w_0 = \Omega \left( \frac{k}{t} \left( \frac{U}{k} \right)^{1-1799} \right) \) such that if \( w < w_0 \), then \( s \geq 2^\frac{\delta k \log(U/k)}{2t} \). The theorem is now implied by just rescaling \( \delta = \epsilon/1799 \) and substituting for \( k' = \Theta(k), m' = \Theta(m) \). □

5 Conclusion

We have shown a \((2 + \epsilon)\)-ANN data structure for time series under the Fréchet distance. What makes this result interesting is that known generic approaches designed for more general classes of
metric spaces cannot be applied. There exist several data structures which operate on general metric spaces with bounded doubling dimension (see e.g. [29, 27, 10]). However, the doubling dimension of the metric space defined over the space of time series with the continuous Fréchet distance is unbounded [17]. We also showed lower bounds in the cell-probe model, which indicate that a better approximation cannot be achieved, unless we allow space usage depending on the arclength of the time series. An interesting question that remains open concerns the case of polygonal curves in higher dimensions. We do not know if there exists a data structure with similar guarantees when the input curves are not restricted to the real line.

References

[1] Dimitris Achlioptas. Database-friendly random projections: Johnson-lindenstrauss with binary coins. *J. Comput. Syst. Sci.*, 66(4):671–687, 2003. doi:10.1016/S0022-0000(03)00025-4.

[2] Peyman Afshani and Anne Driemel. On the complexity of range searching among curves. In *Proceedings of the 28th Annual ACM-SIAM Symposium on Discrete Algorithms, SODA 2018, New Orleans, LA, USA, January 7-10, 2018*, pages 898–917, 2018. doi:10.1137/19781611975031.58.

[3] Helmut Alt and Michael Godau. Computing the Fréchet distance between two polygonal curves. *Int. J. Comput. Geometry Appl.*, 5:75–91, 1995. URL: https://doi.org/10.1142/S0218195995000064.

[4] Helmut Alt and Michael Godau. Computing the Fréchet distance between two polygonal curves. *International Journal of Computational Geometry & Applications*, 05:75–91, 1995. doi:10.1142/S0218195995000064.

[5] Boris Aronov, Omrit Filtser, Michael Horton, Matthew J. Katz, and Khadijeh Sheikhan. Efficient nearest-neighbor query and clustering of planar curves. In *Algorithms and Data Structures - 16th International Symposium, WADS 2019, Edmonton, AB, Canada, August 5-7, 2019, Proceedings*, pages 28–42, 2019. URL: https://doi.org/10.1007/978-3-030-24766-9_3.

[6] Maria Astefanoaei, Paul Cesaretti, Panagiota Katsikouli, Mayank Goswami, and Rik Sarkar. Multi-resolution sketches and locality sensitive hashing for fast trajectory processing. In *International Conference on Advances in Geographic Information Systems (SIGSPATIAL 2018)*, volume 10, 2018.

[7] Julian Baldus and Karl Bringmann. A fast implementation of near neighbors queries for Fréchet distance (GIS Cup). In *Proceedings of the 25th ACM SIGSPATIAL International Conference on Advances in Geographic Information Systems, SIGSPATIAL’17*, pages 99:1–99:4, 2017.

[8] Julian Baldus and Karl Bringmann. A fast implementation of near neighbors queries for Fréchet distance (gis cup). In *Proceedings of the 25th ACM SIGSPATIAL International Conference on Advances in Geographic Information Systems, SIGSPATIAL ’17*, New York, NY, USA, 2017. Association for Computing Machinery. URL: https://doi.org/10.1145/3139958.3140062.

[9] Alberto Bertoni, Giancarlo Mauri, and Nicoletta Sabadini. Simulations among classes of random access machines and equivalence among numbers succinctly represented. *Ann. Discrete Math.*, 25:65–90, 1985.
[10] Alina Beygelzimer, Sham M. Kakade, and John Langford. Cover trees for nearest neighbor. In *Machine Learning, Proceedings of the Twenty-Third International Conference (ICML 2006), Pittsburgh, Pennsylvania, USA, June 25-29, 2006*, pages 97–104, 2006. URL: https://doi.org/10.1145/1143844.1143857
doi:10.1145/1143844.1143857

[11] Karl Bringmann and Wolfgang Mulzer. Approximability of the discrete Fréchet distance. *JoCG*, 7(2):46–76, 2016. doi:10.20382/jocg.v7i2a4

[12] Kevin Buchin, Yago Diez, Tom van Diggelen, and Wouter Meulemans. Efficient trajectory queries under the Fréchet distance (GIS Cup). In *Proc. 25th Int. Conference on Advances in Geographic Information Systems (SIGSPATIAL)*, pages 101:1–101:4, 2017.

[13] Kevin Buchin, Yago Diez, Tom van Diggelen, and Wouter Meulemans. Efficient trajectory queries under the fréchet distance (gis cup). In *Proceedings of the 25th ACM SIGSPATIAL International Conference on Advances in Geographic Information Systems*, SIGSPATIAL ’17, New York, NY, USA, 2017. Association for Computing Machinery. URL: https://doi.org/10.1145/3139958.3140064
doi:10.1145/3139958.3140064

[14] Timothy M. Chan. Well-separated pair decomposition in linear time? *Inf. Process. Lett.*, 107(5):138–141, August 2008. URL: https://doi.org/10.1016/j.ipl.2008.02.008
doi:10.1016/j.ipl.2008.02.008

[15] Mark De Berg, Atlas F Cook, and Joachim Gudmundsson. Fast Fréchet queries. *Computational Geometry*, 46(6):747–755, 2013.

[16] Anne Driemel and Sariel Har-Peled. Jaywalking your dog: Computing the Fréchet distance with shortcuts. *SIAM Journal on Computing*, 42(5):1830–1866, 2013.

[17] Anne Driemel, Amer Krivosija, and Christian Sohler. Clustering time series under the fréchet distance. In *Proceedings of the Twenty-Seventh Annual ACM-SIAM Symposium on Discrete Algorithms*, SODA 2016, Arlington, VA, USA, January 10-12, 2016, pages 766–785, 2016. URL: https://doi.org/10.1137/1.9781611974331.ch55
doi:10.1137/1.9781611974331.ch55

[18] Anne Driemel, Jeff M. Phillips, and Ioannis Psarros. The VC dimension of metric balls under Fréchet and Hausdorff distances. In *Proc. 35st International Symposium on Computational Geometry*, pages 28:2–28:16, 2019.

[19] Anne Driemel and Francesco Silvestri. Locally-sensitive hashing of curves. In *Proc. 33st International Symposium on Computational Geometry*, pages 37:1–37:16, 2017.

[20] Fabian Dütsch and Jan Vahrenhold. A filter-and-refinement-algorithm for range queries based on the Fréchet distance (GIS Cup). In *Proc. 25th Int. Conference on Advances in Geographic Information Systems (SIGSPATIAL)*, pages 100:1–100:4, 2017.

[21] Fabian Dütsch and Jan Vahrenhold. A filter-and-refinement-algorithm for range queries based on the fréchet distance (gis cup). In *Proceedings of the 25th ACM SIGSPATIAL International Conference on Advances in Geographic Information Systems*, SIGSPATIAL ’17, New York, NY, USA, 2017. Association for Computing Machinery. URL: https://doi.org/10.1145/3139958.3140063
doi:10.1145/3139958.3140063

[22] Ioannis Z. Emiris and Ioannis Psarros. Products of Euclidean metrics and applications to proximity questions among curves. In *Proc. 34th Int. Symposium on Computational Geometry (SoCG)*, volume 99 of LIPICS, pages 37:1–37:13, 2018.
[23] Arnold Filtser, Omrit Filtser, and Matthew J. Katz. Approximate nearest neighbor for curves - simple, efficient, and deterministic. CoRR, abs/1902.07562, 2019. arXiv:1902.07562.

[24] Joachim Gudmundsson and Michiel Smid. Fast algorithms for approximate Fréchet matching queries in geometric trees. Computational Geometry, 48(6):479 – 494, 2015. doi:http://dx.doi.org/10.1016/j.comgeo.2015.02.003.

[25] S. Har-Peled, P. Indyk, and R. Motwani. Approximate nearest neighbor: Towards removing the curse of dimensionality. Theory of Computing, 8(1):321–350, 2012. URL: http://dx.doi.org/10.4086/toc.2012.v008a014 doi:10.4086/toc.2012.v008a014.

[26] Sariel Har-Peled. Geometric Approximation Algorithms. American Mathematical Society, Boston, MA, USA, 2011.

[27] Sariel Har-Peled and Manor Mendel. Fast construction of nets in low-dimensional metrics and their applications. SIAM J. Comput., 35(5):1148–1184, 2006. URL: https://doi.org/10.1137/S0097539704446281 doi:10.1137/S0097539704446281.

[28] Piotr Indyk. Approximate nearest neighbor algorithms for Fréchet distance via product metrics. In Symposium on Computational Geometry, pages 102–106, 2002.

[29] Robert Krauthgamer and James R. Lee. Navigating nets: simple algorithms for proximity search. In Proceedings of the Fifteenth Annual ACM-SIAM Symposium on Discrete Algorithms, SODA 2004, New Orleans, Louisiana, USA, January 11-14, 2004, pages 798–807, 2004. URL: http://dl.acm.org/citation.cfm?id=982792.982913.

[30] Stefan Meintrup, Alexander Munteanu, and Dennis Rohde. Random projections and sampling algorithms for clustering of high-dimensional polygonal curves. In Advances in Neural Information Processing Systems 32: Annual Conference on Neural Information Processing Systems 2019, NeurIPS 2019, 8-14 December 2019, Vancouver, BC, Canada, pages 12807–12817, 2019.

[31] Peter Bro Miltersen. Lower bounds for union-split-find related problems on random access machines. In Proceedings of the Twenty-sixth Annual ACM Symposium on Theory of Computing, STOC ’94, pages 625–634, New York, NY, USA, 1994. ACM. doi:10.1145/195058.195415.

[32] Mihai Patrascu. Unifying the landscape of cell-probe lower bounds. SIAM J. Comput., 40(3):827–847, 2011. doi:10.1137/09075336X.

[33] Martin Werner and Dev Oliver. ACM SIGSPATIAL GIS Cup 2017: Range queries under Fréchet distance. SIGSPATIAL Special, 10(1):24–27, June 2018. doi:10.1145/3231541.3231549.
A Computational models

Our data structures operate in the real-RAM model. That is, we assume that the machine can store and access real numbers in constant time and the operations (+, −, ×, ÷, ≤) can be performed in constant time on these real numbers. In addition, we assume that the floor function of a real number can be computed in constant time. This model is commonly used in the literature, see for example [26, 14]. Nonetheless, the use of this computational model is controversial, since it allows all PSPACE and #P problems to be computed in polynomial time [9]. We stress the fact that, in our algorithms, the floor function is only used in snapping points to a canonical grid. In particular, in our data structures, the omission of the floor function (that is, simulating it by the other operations) merely leads to an additional factor in the query time which is bounded by $O(\log(\frac{C}{r}) + \log(\frac{1}{\epsilon}))$, where $C$ is the largest coordinate of any of the input points and $r$ is the parameter that defines the query radius of the ANN data structure. Moreover, the space and the number of cell probes to the data structure is unaffected by this change. Our lower bounds hold in the cell probe model. In this model of computation we are interested in the number of memory accesses (cell probes) to the data structure which are performed by a query. Given a universe of data and a universe of queries, a cell-probe data structure with performance parameters $s$, $t$, $w$, is a structure which consists of $s$ memory cells, each able to store $w$ bits, and any query can be answered by accessing $t$ memory cells. Note that unlike the real-RAM model, the cell-probe model inherently uses bit-complexity as a measure of space, however the space bounds are usually expressed in terms of the number of words.