Property Testing of Curve Similarity

Peyman Afshani
Department of Computer Science, Aarhus University, Denmark

Maike Buchin
Faculty of Computer Science, Ruhr University Bochum, Germany

Anne Driemel
Institute for Computer Science, University of Bonn, Germany

Marena Richter
Institute for Computer Science, University of Bonn, Germany

Sampson Wong
Department of Computer Science, University of Copenhagen, Denmark

Abstract

We propose sublinear algorithms for probabilistic testing of the discrete and continuous Fréchet distance – a standard similarity measure for curves. We assume the algorithm is given access to the input curves via a query oracle: a query returns the set of vertices of the curve that lie within a radius δ of a specified vertex of the other curve. The goal is to use a small number of queries to determine with constant probability whether the two curves are similar (i.e., their discrete Fréchet distance is at most δ) or they are "ε-far" (for $0 < \varepsilon < 2$) from being similar, i.e., more than an ε-fraction of the two curves must be ignored for them to become similar.

We present two algorithms which are sublinear assuming that the curves are t-approximate shortest paths in the ambient metric space, for some $t \ll n$. The first algorithm uses $O(\frac{t}{\varepsilon} \log \frac{t}{\varepsilon})$ queries and is given the value of t in advance. The second algorithm does not have explicit knowledge of the value of t and therefore needs to gain implicit knowledge of the straightness of the input curves through its queries. We show that the discrete Fréchet distance can still be tested using roughly $O(\frac{t^3+t^2 \log n}{\varepsilon})$ queries ignoring logarithmic factors in t. Our algorithms work in a matrix representation of the input and may be of independent interest to matrix testing.

Our algorithms use a mild uniform sampling condition that constrains the edge lengths of the curves, similar to a polynomially bounded aspect ratio. Applied to testing the continuous Fréchet distance of t-straight curves, our algorithms can be used for $(1+\varepsilon')$-approximate testing using essentially the same bounds as stated above with an additional factor of $\text{poly}(\frac{1}{\varepsilon'})$.

2012 ACM Subject Classification Theory of computation → Design and analysis of algorithms

Keywords and phrases Fréchet distance, Trajectory Analysis, Curve Similarity, Property Testing, Monotonicity Testing

Digital Object Identifier 10.4230/LIPIcs.ESA.2025.84

Related Version *Full Version*: https://arxiv.org/abs/2502.17277

Funding *Anne Driemel*: Affiliated with Lamarr Institute for Machine Learning and Artificial Intelligence.

Acknowledgements Anne Driemel would like to thank Ronitt Rubinfeld and Ilan Newman for helpful discussions in the early stages of this research. The authors thank Sariel Har-Peled for feedback on an earlier version of this manuscript.

1 Introduction

We initiate the study of property testing for measures of curve similarity, motivated by the need for fast solutions for curve classification and clustering. Thus, our research lies at the intersection of property testing and computational geometry. While property testing is a very broad area, we believe this intersection has received far less attention than it deserves.

We also believe that property testing for well-studied measures such as the Fréchet distance is especially well-motivated due to its connections to other problems studied on the curves, such as clustering [3, 8], similarity search [1, 7, 14] and map reconstruction [4, 5].

Typically in property testing, we are given access to a (large) data set and the goal is to very quickly assess whether the data has a certain property. The classical notation of correctness is typically not suitable for property testing since these algorithms are intended to be randomized algorithms that access (very small) parts of data via queries. Instead, a property testing algorithm is considered correct if it can satisfy the following two conditions, with a probability close to 1 (e.g., at least $1-\gamma$ probability for some small value of γ): first, if the input has the desired property, the algorithm must return *accept* and second, if the input is "far" from having the property (under some suitable definition of "far"), the algorithm should *reject* the input. Our algorithms have *one-sided error*, meaning, they will not produce *false negatives* and thus rejection means that the input does not have the desired property.

For further reading on property testing, see [24, 2]. Here, we quickly review the main motivation for property testing in general and then we focus on computational geometry.

Property testing algorithms can be useful in many different scenarios. In particular, if the input is extremely large, it makes sense to obtain a quick approximate answer before deciding to run more expensive algorithms. This is especially useful if the testing algorithm has one-sided error; in that case, if the test returns *reject*, there is no need to do any extra work. For the Fréchet distance, there are applications [1, 7, 14] where negative filters are used to minimize expensive Fréchet distance computations; testing may be useful in such cases. Property testing is also useful if most inputs are expected to not have the desired property, or the input distribution is such that each input either has the property or it is "far" from having that property. In addition, the definition of property testing has a strong connection to topics in learning theory such as *probably approximately correct (PAC)* learners and this was in fact one of the important motivations behind its conception [23].

Another motivation for property testing is when small errors can be tolerated or objects that are close to having the desired property are acceptable outcomes; in this view, property testing is in spirit similar to approximation algorithms where an approximate answer is considered to be an acceptable output. In the context of similarity testing we can also compare it to a partial similarity measure. In fact, our chosen error model will be very close to a partial Fréchet distance [6]. For more details on motivations to study property testing see [17, 24]. See also the aforementioned references for more detailed results on property testing as surveying the known results on property testing is beyond the scope of this paper.

Computational geometry has a long tradition of using randomization and sampling to speed up algorithmic approaches [22, 19, 20, 18]. Property testing has received some attention within computational geometry, but is much less explored compared to other areas. There are fast and efficient testers for many basic geometric properties, such as convexity of a point set [13], disjointness of objects [13], the weight of Euclidean minimum spanning tree [10, 11, 12], clustering [21], the Voronoi property of a triangulation [13], ray shooting [9, 13], as well as LP-type problems [15].

Unlike in graph property testing, where it is assumed that we can sample a vertex uniformly at random, and query for its neighbors, it seems that, for geometric data, there is no commonly agreed upon query model. Different types of queries are used to access the data in the above cited works, such as range queries, cone nearest neighbor queries, queries for the bounding box of the data inside a query hyper-rectangle. In our paper, we use queries to the free space matrix of the two input curves, see Section 3 for the precise definition.

We mention that there is also work on matrix testing [16]. Compared to our setting, the input size is defined as n^2 – the number of entries needed to specify the matrix. In our case, we assume the input size to be n, the maximum of column and row dimension of the matrix.

2 Preliminaries

Let (M, d) be a metric space. We say a *(discrete) curve* P in (M, d) is an ordered point sequence $\langle p_1, \ldots, p_n \rangle$ with $p_i \in M$ for all $i = 1, \ldots, n$. We call the points of the curve *vertices*. We denote by $|P|$ the number of vertices in P and by $\ell(P)$ its *length*, which is defined as $\ell(P) = \sum_{i=1}^{n-1} d(p_i, p_{i+1})$. The *subcurve* of P between p_i and p_j is denoted by $P[i, j]$. A curve P is called *t-straight* if for any two vertices p_i and p_j in P, we have $\ell(P[i, j]) \leq t \cdot d(p_i, p_j)$. Denote by $[n]$ the set of integers from 1 to n and by $[n] \times [n] \subset \mathbb{N} \times \mathbb{N}$ the integer lattice of n times n integers. Given two curves $P = \langle p_1, \ldots, p_n \rangle$ and $Q = \langle q_1, \ldots, q_n \rangle$, we say that an ordered sequence $\mathcal{C}$ of elements in $[n] \times [n]$ is a *coupling* of P and Q, if it starts in $(1, 1)$, ends in (n, n) and for any consecutive tuples $(i, j), (i', j')$ in $\mathcal{C}$ it holds that $(i', j') \in \{(i + 1, j), (i, j + 1), (i + 1, j + 1)\}$. We define the *discrete Fréchet distance* between P and Q as follows

$$D_{\mathcal{F}}(P, Q) \coloneqq \min_{\text{coupling } \mathcal{C}} \max_{(i,j) \in \mathcal{C}} d(p_i, q_j).$$

For brevity, we simply call this the Fréchet distance between P and Q. The *free space matrix* of P and Q with distance value δ is an $n \times n$ matrix M_δ, where the i-th column corresponds to the vertex p_i of P and the j-th row corresponds to the vertex q_j of Q.

The entry $M_\delta[i, j]$ has the value 0 if $d(p_i, q_j) \leq \delta$ and 1 otherwise.[1] A coupling path $\mathcal{C}$ is the path through the $[n] \times [n]$ integer lattice defined by the coupling $\mathcal{C}$. We define the *cost* of such a path as $c(\mathcal{C}) = \sum_{(i,j) \in \mathcal{C}} M_\delta[i, j]$. Note that the Fréchet distance between P and Q is at most δ if and only if there exists a coupling path with cost 0 from $(1, 1)$ to (n, n).

Our analysis is based on a property of the free space matrix. We first define this property and then link the property to a certain class of well-behaved input curves.

▶ **Definition 1** (t-local). *Let M be a free space matrix of curves P and Q. We say that M is* t-local *if, for any tuples (i_1, j_1) and (i_2, j_2) with $M[i_1, j_1] = 0 = M[i_2, j_2]$, it holds that $|i_1 - i_2| \leq t \cdot (2 + |j_1 - j_2|)$ and $|j_1 - j_2| \leq t \cdot (2 + |i_1 - i_2|)$.*

See Figure 1 for a visualization. A consequence of this property is that zero-entries within one row or within one column cannot be too far away from each other.

▶ **Observation 2.** *Suppose M is t-local. If we have $M[i, j] = 0 = M[i, j']$, then we have $|j - j'| \leq 2t$. If we have $M[i, j] = 0 = M[i', j]$, we have $|i - i'| \leq 2t$.*

In the following, we argue why we think the t-locality assumption is reasonable. Note that t will not exceed n on any input. There are classes of well-behaved curves that satisfy the locality assumption for some $t \ll n$. Lemma 3 below shows that the free space matrix of approximate shortest paths is t-local. For simplicity, we impose the assumption that the lengths of the edges are bounded by a fixed multiple of δ, the query radius of the Fréchet distance. In the full version, we show that our approach can be adapted to work as well if the lengths of the edges are bounded by a constant multiple of any fixed value.

[1] Note we use 0 and 1 in switched roles compared to the conventions in the literature on the free space matrix. Our notation makes the cost-function of the path more intuitive.

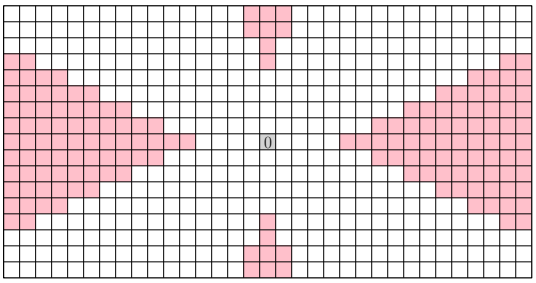

Figure 1 The red cells cannot be zero-entries if the gray cell is a zero-entry and the matrix is 2-local.

▶ **Lemma 3.** *Let P and Q be t-straight curves with edge lengths in $[\delta/\alpha, \alpha\delta]$ for some constant $\alpha \geq 1$. Let M be their free space matrix with distance value δ. Then, M is $\mathcal{O}(t)$-local.*

Proof. Let $(i_1, j_1), (i_2, j_2)$ be such that $M[i_1, j_1] = 0 = M[i_2, j_2]$ and $i_1 \leq i_2$. So we have $d(p_{i_1}, q_{j_1}) \leq \delta$ and $d(p_{i_2}, q_{j_2}) \leq \delta$. We also have $d(q_{j_1}, q_{j_2}) \leq \ell(Q[j_1, j_2]) \leq \alpha\delta\,|j_1 - j_2|$ because of the edge length constraint, where $Q[j_1, j_2]$ is replaced by $Q[j_2, j_1]$ if $j_2 < j_1$. Using the edge length constraint and t-straightness we derive $\frac{\delta}{\alpha}\,|i_2 - i_1| \leq \ell(P[i_1, i_2]) \leq t d(p_{i_1}, p_{i_2})$. These observations together with the triangle inequality yield

$$|i_2 - i_1| \leq \frac{\alpha t}{\delta}(d(p_{i_1}, q_{j_1}) + d(q_{j_1}, q_{j_2}) + d(q_{j_2}, p_{i_2})) \leq \alpha^2 t(2 + |j_2 - j_1|).$$

The other inequality is shown by reversing the roles of P and Q. ◀

3 Problem definition and results

The problem we study in this paper is the following. Assume we want to determine for two curves P and Q, each consisting of n vertices,[2] if their Fréchet distance is at most a given value of δ. We do not have direct access to the input curves. Instead, we have access to an oracle that returns the information in a given row or column of the δ-free space matrix in the form of a sorted list of indices of zero-entries. We call this a *query* and we want to determine $D_{\mathcal{F}}(P, Q) \leq \delta$ using as few (sublinear in n) queries as possible. Note that from the point of view of a data structure setting, our query corresponds to a classical ball range query with a vertex p of one curve and returns the list of vertex indices of the other curve that are contained in the ball of radius δ centered at p.

Our bounds on the number of queries will be probabilistic and will hold under a certain error model that allows for the coupling path to pass through a bounded number of one-entries. These are entries where the corresponding points are far from each other. In the full version, we discuss alternative error models and how they relate to our chosen error model.

▶ **Definition 4** ((ε, δ)-far)**.** *Given two curves P and Q consisting of n vertices each, we say that P and Q are (ε, δ)-far from each other if there exists no coupling path from $(1, 1)$ to (n, n) in the δ-free space matrix of cost εn or less.*

▶ **Definition 5** (Fréchet-tester)**.** *Assume we are given query-access to two curves P and Q, and we are given values $\delta > 0$ and $0 < \varepsilon < 2$. If the two curves have Fréchet distance at most δ, we must return "yes", and if they are (ε, δ)-far from each other w.r.t. the Fréchet distance, the algorithm must return "no", with probability at least $\frac{4}{5}$.*

[2] For ease of notation, our analysis assumes the input curves have the same number of vertices.

Results. We present two algorithms for testing whether a pair of input curves have discrete Fréchet distance at most a given real value δ. Assume that the algorithm has query access only to the δ-free space matrix of the input curves and that this matrix is t-local (see Definition 1). The first algorithm requires knowledge of t, and it uses $O(\frac{t}{\varepsilon}\log\frac{t}{\varepsilon})$ queries (Theorem 18). Note that this bound is intrinsically independent of n while t can be at most n, by the definition of locality. The second algorithm does not require knowledge of t in advance (and thus it can be applied to any matrix) and requires $\mathcal{O}((t^3+t^2\log n)\frac{\log\log t}{\varepsilon})$ many queries (Theorem 25). This algorithm gains its knowledge of t with a variant of monotonicity testing (Theorem 23). Using Lemma 3 these results directly imply the same bounds for the discrete Fréchet distance of t-straight curves with bounded edge lengths. In Section 6 we show how to apply our algorithms to testing the continuous Fréchet distance. As a result, we obtain essentially the same bounds with respect to t-straight curves while removing the assumption on the edge lengths entirely (Theorem 29).

Techniques. Our algorithms use the concept of permeability, which tests whether a specified subcurve has small partial Fréchet distance to any subcurve of the other curve. Concretely, the algorithm samples a block of consecutive columns (or rows) in the δ-free space matrix and checks if there is a coupling path of cost 0 passing somewhere through this block (see Algorithm 1). If the algorithm happens to find a block that does not satisfy this condition, then we know that the global Fréchet distance is larger than the distance parameter δ and the algorithm can safely return 'no'. The difficulty lies in proving that after a certain number of random queries, the algorithm can return 'yes' with sufficient certainty. To this end, we show that if the two curves are (ε,δ)-far then the algorithm, which tests subcurves of varying sizes up to roughly $O(\frac{t}{\varepsilon})$, is likely to sample a permeability query that would return 'no'.

In the full version, we discuss a simple tester for the Hausdorff distance and a simple t-approximate Fréchet tester. This algorithm merely tests for columns and rows that contain one-entries only and are therefore somewhat blocked for coupling paths in the free space matrix. The example in Figure 2 shows that such simple queries are not sufficient to test the Fréchet distance exactly.

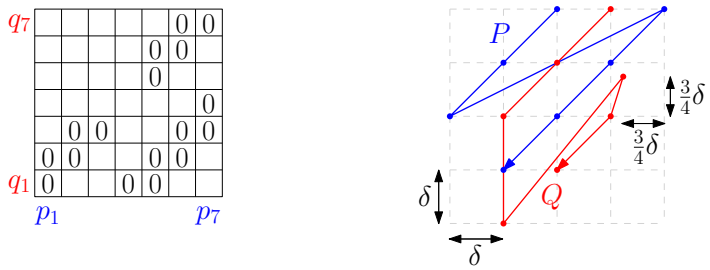

Figure 2 On the left we have M_δ for the curves on the right. All zero-entries are written.

4 Testing the discrete Fréchet distance

In this section, we describe and analyze a Fréchet tester under the assumption that the free space matrix is t-local for a given value of t.

4.1 The algorithm

The idea of Algorithm 1 is to sample a number of columns and rows and check whether there is locally a coupling path of cost zero possible. The following definition classifies when such a (local) path exists.

▶ **Definition 6** (Permeability). *We say a block $[i, i']$ of consecutive columns (resp., rows) from index i to index i' is* permeable *if there exists a coupling path of cost zero that starts in column (resp., row) i and ends in column (resp., row) i'.*

If a column (resp. row) is individually not permeable, i.e. it contains only one-entries, we call it a *barrier-column* (resp. *barrier-row*). Note that any non-permeable block of rows or columns is a witness to the fact that no global coupling path of cost 0 exists and that the algorithm can answer "no".

The algorithm first tests if the first or last entry (i.e. $M[1,1]$ or $M[n,n]$) is a one-entry. If not, it queries $\mathcal{O}\left(\frac{t}{\varepsilon}\right)$ randomly sampled columns and rows and checks if any of them is a barrier-column or barrier-row. In Lines 6-8 we sample $\mathcal{O}(\frac{t}{2^{i+1}})$ random intervals of length 2^{i+2} for each i from 0 to $\mathcal{O}\left(\log \frac{t}{\varepsilon}\right)$. We then check all corresponding blocks of columns and rows for permeability and if all blocks are permeable, we return "yes".

Algorithm 1 Fréchet-Tester1$(M, t, \varepsilon, k = \frac{24t}{\varepsilon}, c = 4)$.

1. **If** $M[1,1] = 1$ or $M[n,n] = 1$ **then return** "no".
2. **repeat** $\lceil k \rceil$ **times**:
3. $\quad j \leftarrow$ sample an index uniformly at random from $[n]$.
4. $\quad$ **if** row j or column j of M is a barrier-column or barrier-row **then return** "no".
5. $K \leftarrow \lceil \frac{\varepsilon n}{32t} \rceil - 1$, $\ell \leftarrow \lceil \frac{128t}{\varepsilon} \rceil$, let $\mathcal{J}$ be a set of intervals and set $\mathcal{J} \leftarrow \emptyset$.
6. **for** $i = 0, \ldots, \lfloor \log_2 \ell \rfloor$ **do**:
7. $\quad I \leftarrow$ sample $\lceil \frac{4cn}{2^{i+1}K} \rceil$ different indices uniformly at random from $\{0, 1, \ldots, \frac{n}{2^{i+1}} - 2\}$.
8. $\quad$ **for each** $j \in I$ **do**: add $[j2^{i+1}, (j+2)2^{i+1}]$ to $\mathcal{J}$.
9. **foreach** $[i, j] \in \mathcal{J}$ **do**
10. $\quad$ **if** block $[i, j]$ of consecutive columns is not permeable **then return** "no".
11. $\quad$ **if** block $[i, j]$ of consecutive rows is not permeable **then return** "no".
12. **return** "yes".

To check if a block of k columns (or rows) is permeable, the algorithm first performs k queries to obtain the positions of zero-entries in these columns (or rows), then we build the induced subgraph of the grid for these zero-entries, connect all neighboring zero-entries according to the possible steps of a coupling and then connect all zero-entries of the last column to a sink and all zero-entries of the first column from a source. It remains to check if there is a path from source to sink, which can be done in linear time since the graph is acyclic. The running time is linear in the total number of zero-entries queried. We call this a *permeability query*.

4.2 Basic properties of permeability

For two curves P and Q with n vertices each, and a t-local free space matrix, Algorithm 1 returns "no" only if it has determined that no coupling path of cost zero from $(1,1)$ to (n,n) can exist (by finding blocks of rows or columns that are not permeable). We need to show that it returns "yes" correctly with sufficiently high probability.

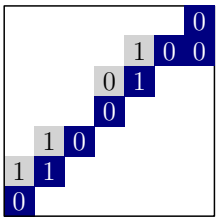

Figure 3 The blue path is a coupling path of cost 2. By adding the gray entries, we get a diagonal-restricted path of cost 5.

For technical reasons, our analysis uses a specific kind of coupling paths that are more restricted than the ones given in Section 2. We introduce them here and argue that it suffices to consider these coupling paths only.

▶ **Definition 7** (diagonal-restricted path). *A* diagonal-restricted path $\mathcal{C}$ *through the free space matrix* M *is a path corresponding to a coupling* $\mathcal{C}$ *in which for any consecutive tuples of the form* $(i,j),(i+1,j+1) \in \mathcal{C}$ *it holds that* $M[i,j] = 0 = M[i+1,j+1]$.

In other words, a diagonal-restricted path $\mathcal{C}$, is allowed to take diagonal steps in the free space matrix only if both entries are zero. Hence, any one-entry visited by $\mathcal{C}$ must be reached by either a vertical or horizontal step in M and it must be left by a vertical or horizontal step in M. This definition, which might seem overly technical at first sight, will simplify a lot of our arguments. The following observations show that it suffices to consider diagonal-restricted paths instead of all coupling paths in the analysis of a Fréchet tester.

▶ **Observation 8.** *For two curves* P *and* Q *with* n *vertices each, the following holds:*

- **i)** *There exists a diagonal-restricted path of cost zero from* $M_\delta[1,1]$ *to* $M_\delta[n,n]$ *if and only if there exists a coupling path of cost zero from* $M_\delta[1,1]$ *to* $M_\delta[n,n]$.
- **ii)** *If there is no diagonal-restricted path of cost at most* εn *through* M_δ *from* $M_\delta[1,1]$ *to* $M_\delta[n,n]$, *the curves are* $(\frac{\varepsilon}{3},\delta)$*-far. See Figure 3.*
- **iii)** *If* P *and* Q *are* (ε,δ)*-far, there is no diagonal-restricted path of cost at most* εn *through* M_δ *from* $M_\delta[1,1]$ *to* $M_\delta[n,n]$.

Below, we present three structural lemmas that help with the analysis of Algorithm 1. For the proofs of these lemmas, we refer to the full version. The first lemma considers a subpath of a diagonal-restricted path $\mathcal{C}$ that starts and ends in a zero-entry and visits only one-entries in between. We show that the bounding box of the two zero-entries must otherwise be filled with one-entries. Note that this lemma does not hold for general coupling paths of Section 2. Using it will simplify our analysis and is the main motivation for introducing Definition 7.

▶ **Lemma 9** (Box of ones). *Let* $\mathcal{C}$ *be a diagonal-restricted path of minimum cost. Let* $(i,j),(i',j') \in \mathcal{C}$ *be zero-entries such that* $\mathcal{C}$ *visits only one-entries between them. Then, for all tuples* $(k,l) \in \{(k,l) \mid i \le k \le i', j \le l \le j'\} \setminus \{(i,j),(i',j')\}$ *we have that* $M[k,l] = 1$.

We give a definition describing when two columns or rows satisfy the constraints for t-locality.

▶ **Definition 10.** *The columns* i_1 *and* i_2 pass t-locality *if any two zero-entries* (i_1,j_1) *and* (i_2,j_2) *in columns* i_1 *and* i_2 *satisfy* $|j_1 - j_2| \le t \cdot (2 + |i_1 - i_2|)$. *Similarly the rows* j_1 *and* j_2 pass t-locality *if any two zero-entries* (i_1,j_1) *and* (i_2,j_2) *in rows* j_1 *and* j_2 *satisfy* $|i_1 - i_2| \le t \cdot (2 + |j_1 - j_2|)$.

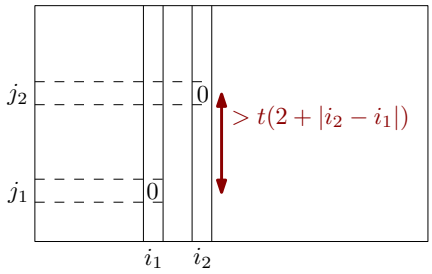

Figure 4 Two columns that do not pass t-locality.

A column (resp. row) can also pass t-locality with itself if $i_1 = i_2$ (resp. $j_1 = j_2$). Intuitively, two columns pass t-locality, when all their zero-entries are not too far away vertically. Note that all columns (resp. rows) pass t-locality with each other if M is t-local.

The second lemma shows that if an optimal diagonal-restricted path contains a long sequence of one-entries, then there must be many barrier-columns and barrier-rows.

▶ **Lemma 11** (Barrier Lemma). *Let $\mathcal{C}$ be a diagonal-restricted path of minimum cost. Suppose that there are two zero-entries $(i,j),(i',j') \in \mathcal{C}$ such that $\mathcal{C}$ visits no zero-entry and at least $4t$ one-entries in between them and suppose that all columns in $[i,i']$ and all rows in $[j,j']$ pass t-locality with each other. Then there is a total of at least $\lceil \frac{i'-i+j'-j}{2t} \rceil$ barrier-rows between j and j' and barrier-columns between i and i'.*

The third lemma shows that if an optimal diagonal-restricted path has a long stretch with relatively many one-entries on the path, there cannot be any long coupling path of cost zero in the same sequence of rows or columns of the matrix M, implying impermeability.

▶ **Lemma 12** (Impermeability Lemma). *Let $\mathcal{C}$ be a diagonal-restricted path through M of minimum cost. Suppose $(i,j),(i',j') \in \mathcal{C}$ with $j' - j > 2t$ (resp. $i' - i > 2t$) correspond to zero-entries in M and the subpath of $\mathcal{C}$ from (i,j) to (i',j') visits at least $4t-1$ one-entries and suppose that any column in $[i,i']$ and any row in $[j,j']$ passes t-locality with itself. Then, the block of columns $[i,i']$ (resp. rows $[j,j']$) is not permeable.*

4.3 Analysis of the algorithm

Algorithm 1 samples a range of intervals in Lines 6-8 which are then tested with a permeability query. For this subroutine we can prove the following lemma. For a proof see the full version.

▶ **Lemma 13.** *Let $\mathcal{I} = \{[i,j] \mid 1 \leq i < j \leq n\}$ be a set of all non-empty intervals from $[n]$ and let $X \subset \mathcal{I}$ be a set of K unknown intervals such that the length of each interval is at most ℓ and each element of $[n]$ is contained in at most c of these intervals for some constant $c \geq 1$. Lines 6-8 of Algorithm 1 select a subset $\mathcal{J} \subset \mathcal{I}$ of intervals with $\sum_{J \in \mathcal{J}} |J| = O(\frac{n \log \ell}{K})$ such that with probability at least $9/10$, at least one of the intervals in X is contained in at least one interval in $\mathcal{J}$.*

The next lemma shows that Algorithm 1 is very likely to return "no" in Line 4 if there are many barrier-columns and barrier-rows. The proof can be found in the full version.

▶ **Lemma 14.** *If the total number of barrier-columns and barrier-rows is more than $\frac{3n}{k}$, then Algorithm 1 returns "no" in Line 4 with probability at least $\frac{9}{10}$.*

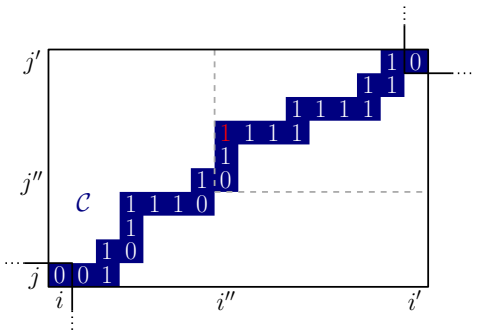

Figure 5 Example for grouping in the proof of Lemma 15 with $t = 2$. The group starts in (i, j). The red one-entry is the $(4t + 1)$-th one-entry in the group. So we stop the group at the next zero-entry (i', j'). Since it contains more than $8t$ one-entries, this is a hollow group. The entries $(i'', j''), (i', j')$ satisfy the conditions in the Barrier (Lemma 11).

▶ **Lemma 15** (Main Lemma). *Let M be t-local and $M[1, 1] = 0 = M[n, n]$. Suppose the total number of barrier-columns and barrier-rows is at most $\frac{\varepsilon n}{8t}$. Let $\mathcal{C}$ be a diagonal-restricted path with lowest cost through M and suppose that $c(\mathcal{C}) > \varepsilon n$. Then, with probability at least $\frac{9}{10}$ at least one sampled interval in the set $\mathcal{J}$ during the execution of Algorithm 1 corresponds to a non-permeable block of columns or rows.*

Proof. We first divide the path $\mathcal{C}$ into a minimum number of groups that each contain more than $4t$ one-entries and start and end with a zero-entry. A visualization of this can be seen in Figure 5. Here, the start of one group is the same entry as the end of the prior group. We do this greedily by following $\mathcal{C}$ until we have visited $4t + 1$ one-entries. Then, we end the group at the next zero-entry after that and repeat the process. Note that the last group may contain fewer than $4t + 1$ one-entries.

We will first show that there are not too many groups that contain more than $8t$ one-entries. Then, we will show that within the remaining groups, there are sufficiently many groups that are not too big. With these groups we then form the set X for the interval sampling in Lemma 13 (i.e., each interval in X will induce a permeability query that fails).

A group that contains more than $8t$ one-entries is called *hollow*, otherwise, it is *dense*.

▷ Claim 16. At most $\frac{\varepsilon n}{2}$ of the one-entries of $\mathcal{C}$ are in hollow groups.

For a proof of the claim, we refer to the full version. So we know that at least $\frac{\varepsilon n}{2}$ of the one-entries of $\mathcal{C}$ are in dense groups. Since a dense group contains at most $8t$ one-entries, we have at least $\lceil \frac{\varepsilon n}{16t} \rceil$ dense groups. Let G be a dense group that starts in (i, j) and ends in (i', j'). We say that $i' - i + j' - j + 2$ is the *size* of G. This corresponds to the sum of the number of columns and rows that it visits. In total, every row and column is counted at most twice by the definition of the groups. So the sum of all group sizes is at most $4n$. We observe that at least half of the groups (which are at least $\lceil \frac{\varepsilon n}{32t} \rceil$ many) have size at most $2 \cdot 4n / \lceil \frac{\varepsilon n}{16t} \rceil \leq \frac{128t}{\varepsilon}$. We call these groups *small*. At last, we create a set X of intervals so that we can apply Lemma 13 to it. Each of the intervals $[i, i']$ in X will have the property that either the block $[i, i']$ of consecutive columns or the block $[i, i']$ of consecutive rows is not permeable.

Let G be a small dense group and suppose it starts in (i, j) and ends in (i', j'). We observe that unless G is the very last group, it visits at least $4t + 1$ one-entries and therefore either visits more than $2t$ rows or more than $2t$ columns. In the first case, we can apply the

Impermeability Lemma to see that the block $[i, i']$ of columns is not permeable. So we add $[i, i']$ to X. In the second case, we can apply the Impermeability Lemma to see that the block $[j, j']$ of rows is not permeable. So we add $[j, j']$ to X. In the end the set X has at least $\lceil \frac{\varepsilon n}{32t} \rceil - 1$ elements and each of them has length at most $\frac{128t}{\varepsilon}$. If we take a closer look at the definitions of the groups, we see that all columns are only contained in one group except for the first and last column of each group. So we can say that every column is contained in at most two groups, i.e. at most two intervals of X. The same holds for rows. So, we have that each element of $[n]$ is contained in at most four of the intervals in X. So we can apply Lemma 13 to X with $\ell = \lceil \frac{128t}{\varepsilon} \rceil$, $K = \lceil \frac{\varepsilon n}{32t} \rceil - 1$, $c = 4$ to see that the algorithm samples one of the intervals from X with probability at least $\frac{9}{10}$. So the permeability query fails for the columns or for the rows with the respective interval, which yields the desired statement. ◀

▶ **Lemma 17.** *Algorithm 1 performs $\mathcal{O}(\frac{t}{\varepsilon} \log \frac{t}{\varepsilon})$ queries.*

Proof. Line 4 performs $\mathcal{O}(\frac{t}{\varepsilon})$ queries. By Lemma 13, Lines 6-8 sample a set $\mathcal{J}$ of intervals that contain a total of $\mathcal{O}(\frac{n \log \ell}{K})$ rows and columns. Each of these columns and rows cause a query in a permeability query. As $\ell = \lceil \frac{128t}{\varepsilon} \rceil$ and $K = \lceil \frac{\varepsilon n}{32t} \rceil - 1$, the claim follows. ◀

▶ **Theorem 18.** *Let $\delta > 0$ and $0 < \varepsilon < 2$ be given. Let P and Q be curves with n vertices such that their free space matrix with value δ is t-local and t is known. Then, Algorithm 1 is a Fréchet-tester that uses $\mathcal{O}(\frac{t}{\varepsilon} \log \frac{t}{\varepsilon})$ queries.*

Proof. If the Fréchet distance of P and Q is at most δ, there is a coupling path of cost 0 through M and Algorithm 1 always returns "yes". So we only need to determine the probability of a false positive. Suppose P and Q are (ε, δ)-far and therefore, a minimum cost diagonal-restricted path visits more than εn one-entries. If there are at least $\frac{\varepsilon n}{8t}$ barrier-columns and barrier-rows, we return "no" with probability at least $\frac{9}{10}$ in Line 4 by Lemma 14. If there are fewer than $\frac{\varepsilon n}{8t}$ barrier-columns and barrier-rows and we get past Line 4, we return "no" with probability at least $\frac{9}{10}$ in Lines 10 and 11 by the Lemma 15. So in any case, the probability for a false positive is at most $\frac{1}{10}$. The number of queries follows from Lemma 17. ◀

5 Locality-oblivious testing of the discrete Fréchet distance

Our main goal in this section is to obtain a Fréchet-tester that does not require an additional input (the value of t). Our approach for this is to first run a "tester" to find an estimate for t and then run Algorithm 1 with that parameter. Unfortunately, this combination is more complicated than just running two testers, for a number of different reasons: First, Algorithm 1 assumes that the matrix is t-local whereas the best we can do is to show that with some probability close to 1, it is "close" to being t-local. This change breaks the proof of correctness of Algorithm 1. The second complication is that we cannot plug in *any* tester for locality, we need a specific one for our analysis that also guarantees some additional properties. Guaranteeing these properties requires more queries than just testing for locality.

5.1 Testing the locality of the Free Space Matrix

To this end, we present a specific error model that is suitable for our purposes. In the following we define how t-locality of a matrix can be violated. First, it could be that two columns or rows do not pass t-locality according to Definition 10. In this case, we say that the respective columns or rows *fail t-locality*. We introduce a second definition that considers a single column and looks for violations of t-locality within that column and within the rows that have a zero-entry in this column.

▶ **Definition 19.** *A single column i (resp. row j)* fails second order t-locality *if either it contains two zero-entries $(i,j),(i,j')$ (resp. $(i,j),(i',j)$) with $|j'-j|>2t$ (resp. $|i'-i|>2t$) or if there exists a zero-entry (i,j) and two zero-entries (i_1,j) and (i_2,j) (resp. (i,j_1) and (i,j_2)) such that $|i_1-i_2|>2t$ (resp. $|j_1-j_2|>2t$). Otherwise, it* passes second order t-locality.

Intuitively, a single column fails second order t-locality if it has a zero-entry in a row that also has two zero-entries more than $2t$ distance apart (similar intuition holds for the rows). See Figure 6. Observe that if a matrix is not t-local then there either exists a pair of columns or rows that fail t-locality or a column or row that fails second order t-locality.

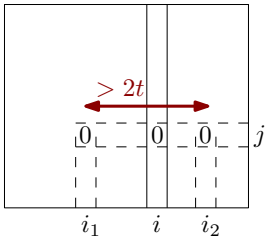

Figure 6 A column i fails second order t-locality.

We now want to define what it means for a matrix to be ζ-close to t-local. This definition is specifically designed to fit the Fréchet-tester in Section 5. Other, maybe more natural definitions of (t,ζ)-close may need different property-testers. However, testing t-locality is not our main focus in this paper.

▶ **Definition 20** (strongly (t,ζ)-local)**.** *We say that a free space matrix M is* strongly (t,ζ)-local *for some $t\geq 1$ and $\zeta\in[0,1]$ if we can partition the set of all columns and rows into two disjoint sets W and I with $|I|=\zeta n$ such that the following properties hold: i) columns $1,n$ and rows $1,n$ are in W, ii) any two columns and any two rows in W pass t-locality and iii) any row or column in W passes second order t-locality. We call the set W the* witness set *and the columns and rows in I* ignored.

If there are $(1-\zeta)n$ rows and $(1-\zeta)n$ columns such that no two of these rows or columns fail t-locality and none of these columns or rows fail second order t-locality, then they serve as a witness set to see that the matrix is $(t,2\zeta)$-local. Because of this, we will check for failures of t-locality and second order t-locality among the columns and rows.

▶ **Observation 21.** *When we query two columns i and i', we can check if they fail t-locality by comparing the lowest zero-entry in column i with the highest zero-entry in column i' and vice versa. Also, after querying a column i, testing for second order t-locality can be done with at most $2t+1$ additional row queries at all zero-entries of the column i.*

Our testing algorithm is quite simple and, in fact, it is an extension of the classic "monotonicity" testing in an array [2]. The algorithm is given in Algorithm 2.

▶ **Lemma 22.** *Given $\sigma>0$ and $t\geq 1$, Algorithm 2 returns "yes" if M is t-local; if M is not strongly $(2t,\sigma)$-local, it returns "no" with probability at least $\frac{9}{10}$. It uses $\mathcal{O}(\frac{t+\log n}{\sigma})$ queries.*

For a detailed proof of Lemma 22 we refer to the full version. The proof relies on standard techniques for monotonicity testing [2]. The Locality-tester is given a value t as an argument. Next, we show that it is also possible to obtain a good estimate of the locality, as well. The

Algorithm 2 Locality-tester(M,σ,t).

1. $T \leftarrow$ binary search tree of height $\lceil \log_2 n \rceil$ on $[n]$ with root $r = \lceil \frac{n}{2} \rceil$.
2. **repeat** $\lceil \frac{3}{\sigma} \rceil + 2$ times:
3. First two iterations: $i \leftarrow 1$ and $i \leftarrow n$. Later: Choose $i \in [2, n-1]$ unif. at random.
4. **If** column i or row i fails second order t-locality **then** return "no".
5. **foreach** j on the r-i-path in T **do**:
6. **If** columns i and j fail t-locality **then** return "no".
7. **If** rows i and j fail t-locality **then** return "no".
8. return "yes".

straightforward approach is to start with a small estimate for t, e.g., $t = 1$, use Algorithm 2 and every time it fails, we can double the t and keep repeating. Our algorithm is very similar to this approach. The difference is merely that we use different ζ values throughout the iterations and we call the locality tester multiple times for the same t.

▶ **Theorem 23.** *Suppose M is t^*-local and we are given $\zeta > 0$. We can design an algorithm that returns a value $t \leq 2t^*$ such that the matrix is strongly $(t, \zeta/t^2)$-local with probability at least $\frac{8}{9}$. The number of queries used is $\mathcal{O}(\frac{\log\log t}{\zeta}(t^3 + t^2 \log n))$.*

Proof. The algorithm operates in rounds. In the i-th round (starting from $i = 1$) we run Locality-tester(M, $\frac{\zeta}{2^{2(i+1)}}, 2^i$) for $2(1 + \log i)$ times. If the locality-tester returns "no" in any of these runs, we go to round $i + 1$, otherwise, we return $t = 2^{i+1}$.

First, we examine the number of queries. Let $i^* = \lceil \log t^* \rceil$. Then, the algorithm terminates before or in round i^* since no two columns or rows fail 2^i-locality for any $2^i \geq t^*$ and all rows and columns pass second order t^*-locality. So we have at most i^* rounds. In round i, we perform $\mathcal{O}((2 + 2\log i) \cdot \frac{2^{2(i+1)}}{\zeta} \cdot (2^i + \log n))$ queries by Lemma 22. The total number of all queries from round 1 to the last round (which is at most i^*) is $\mathcal{O}(\frac{\log\log t}{\zeta}(t^3 + t^2 \log n))$.

By Lemma 22, "no" answers are always correct. However, with probability at most $\frac{1}{10}$, the algorithm can return "yes" even though M is not $(2 \cdot 2^i, \frac{\zeta}{2^{2(i+1)}})$-local. As all the executions of the algorithm are independent, the probability that we get "yes" in all $2(1 + \log i)$ calls of the locality-tester in round i is at most $10^{-(2(1+\log i))} < \frac{1}{100i^2}$. Using a union bound, the probability that we return a value t while M is not $(t, \frac{\zeta}{t^2})$-local is at most $\sum_{i=1}^{\infty} \frac{1}{100i^2} < \frac{1}{9}$. ◀

5.2 The combined algorithm

We now present our algorithm for testing the Fréchet distance without knowing the locality parameter t in advance. Algorithm 3, uses the results from Section 5.1 to first estimate the locality of the matrix M and then uses the Fréchet-tester from Section 4 to decide if the Fréchet distance of the two curves is smaller or equal to δ.

Algorithm 3 Fréchet-tester2(M, ε).

1. $t \leftarrow$ Output of the Algorithm in Theorem 23 for M and $\zeta = \frac{\varepsilon}{1600}$.
2. **return** Fréchet-tester1($M, t, \varepsilon/3, \frac{4800t^2}{\varepsilon}, 2$).

The following lemma is analogous to the main lemma (Lemma 15) from Section 4. For a proof of this lemma we refer to the full version.

▶ **Lemma 24.** *Let $\varepsilon > 0$ and let M be strongly $(t, \frac{\varepsilon}{1600t^2})$-local and $M[1,1] = 0 = M[n,n]$. Suppose the total number of barrier-columns and barrier-rows is at most $\frac{\varepsilon n}{1600t^2}$. Let $\mathcal{C}$ be a diagonal-restricted path with lowest cost through M and suppose that $c(\mathcal{C}) > \varepsilon n$. Then, with probability at least $\frac{9}{10}$ at least one sampled interval in the set $\mathcal{J}$ in the call of Algorithm 1 during Algorithm 3 corresponds to a non-permeable block of columns or rows with probability at least $\frac{9}{10}$.*

▶ **Theorem 25.** *Let $\delta > 0$ and $0 < \varepsilon < 2$ be given. Let P and Q be curves with n vertices such that their free space matrix is t^*-local. Then, Algorithm 3 is a Fréchet-tester that uses $\mathcal{O}(((t^*)^3 + (t^*)^2 \log n)\frac{\log\log t^*}{\varepsilon})$ queries.*

Proof. The Locality-tester uses $\mathcal{O}(\frac{\log\log t^*}{\varepsilon}((t^*)^3 + (t^*)^2 \log n))$ queries. Since $t \leq 2t^*$, we need $\frac{(t^*)^2}{\varepsilon}$ queries to test for barrier-columns and barrier-rows and $\mathcal{O}\left(\frac{t^*}{\varepsilon}\log\left(\frac{t^*}{\varepsilon}\right)\right)$ queries for the permeability queries. The total number of queries follows since $\frac{t^*}{\varepsilon} \in \mathcal{O}(n^2)$.

If the Fréchet distance of P and Q is at most δ, Algorithm 3 will always return "yes". So we have to bound the probability of false positives. Assume that P and Q are (ε, δ)-far and therefore, a minimum cost diagonal-restricted path $\mathcal{C}$ through M has cost higher than εn. Suppose that M is strongly $(t, \frac{\varepsilon}{1600t^2})$-local for the output t of the algorithm from Theorem 23. By the same theorem, this happens with probability at least $\frac{8}{9}$. If the matrix contains more than $\frac{\varepsilon n}{1600t^2}$ barrier-columns and barrier-rows, the subroutine that calls Algorithm 1 returns "no" with probability at least $\frac{9}{10}$ by Lemma 14. If the matrix contains at most $\frac{\varepsilon n}{1600t^2}$ barrier-columns and barrier-rows and the algorithm proceeds until the interval sampling, we are in the setting of Lemma 24 and we return "no" with probability at least $\frac{9}{10}$. Therefore, the probability that the algorithm returns "no" is at least $\frac{8}{9} \cdot \frac{9}{10} = \frac{4}{5}$. So the probability of a false positive is at most $\frac{1}{5}$. ◀

6 Testing the continuous Fréchet distance

This section describes how the Fréchet-testers for the discrete Fréchet distance can be used to serve as $(1+\varepsilon')$-approximate Fréchet-testers for the continuous Fréchet distance. First, we define (ε, δ)-far for the continuous Fréchet-distance. Then, we explain how to use our Fréchet-testers on instances of the continuous Fréchet distance with this error model.

We first define the continuous Fréchet distance. We set F_n to be the set of all continuous non-decreasing functions $f\colon [0,1] \to [1,n]$ with $f(0) = 1$ and $f(1) = n$. Let P and Q be polygonal curves with n and m vertices. The *continuous Fréchet distance* is defined to be

$$\delta_{\mathcal{F}}(P,Q) = \min_{f\in F_n, g\in F_m} \max_{\alpha\in[0,1]} d(P(f(\alpha)), Q(g(\alpha))).$$

Now we want to introduce a possible error model for the continuous Fréchet distance. Given a value $\delta > 0$ and $f \in F_n$, $g \in F_m$, we define

$$\mathcal{P}_{f,g}(P,Q) \coloneqq \int_{d(P(f(t)),Q(g(t)))>\delta} (||(P\circ f)'(t)|| + ||(Q\circ g)'(t)||)dt$$

as the partial difference, namely the total length of the portions of the two curves P and Q that are not matched with distance at most δ by f and g. Building on this, we define the *partial Fréchet difference* to be

$$\mathcal{P}_{\delta}(P,Q) \coloneqq \min_{f\in F_n, g\in F_m} \mathcal{P}_{f,g}(P,Q).$$

This is essentially the same as the partial Fréchet distance as defined in [6].[3]

▶ **Definition 26** ((ε,δ)-far)**.** *Given two polygonal curves P and Q, we say that P and Q are (ε,δ)-far from one another w.r.t. the continuous Fréchet distance if $\mathcal{P}_\delta(P,Q) \geq \varepsilon(\ell(P)+\ell(Q))$.*

▶ **Definition 27** (μ-approximate Fréchet-tester)**.** *Assume we are given query-access to two curves P and Q, and we are given values $\delta > 0$ and $0 < \varepsilon < 1$. If the curves have continuous Fréchet distance at most δ, we must return "yes", if they are $(\varepsilon,\mu\delta)$-far from each other w.r.t. the continuous Fréchet distance, the algorithm must return "no", with probability at least $\frac{4}{5}$.*

In order to use the first Fréchet-tester introduced in this paper, we transform the polygonal curves into discrete curves by subsampling vertices along the edges of the curves. Denote by P_a the discrete curve that arises from P if we subsample vertices along the edges of P such that P visits a curve segment of length a in between any two vertices for some $a \in (0,\ell(P)]$. The last edge might be longer than a but shorter than $2a$. Hence, P_a has $\lfloor \frac{\ell(P)}{a} \rfloor$ edges. This yields that $D_{\mathcal{F}}(P_a,Q_a) - 2a < \delta_{\mathcal{F}}(P,Q) \leq D_{\mathcal{F}}(P_a,Q_a) + 2a$. If P is t-straight, so is P_a for any $a \in (0,\ell(P)]$. The next lemma shows that the notions of (ε,δ)-far are also related.

▶ **Lemma 28.** *Let $0 < a \leq \min\{\ell(P),\ell(Q)\}$. If P and Q are $(12\varepsilon,\delta+2a)$-far from each other, then P_a and Q_a are (ε,δ)-far from each other.*

For a proof see the full version. We adjust our query oracle such that it can access the free space matrix of P_a and Q_a. So we can apply the Fréchet-testers to P_a and Q_a for an adequate choice of a. We show that for given $\varepsilon' > 0$ and $0 < \varepsilon < 2$, we can choose a such that this yields a $(1+\varepsilon')$-approximate Fréchet-tester for the continuous Fréchet distance of P and Q. Let $\varepsilon'' > 0$. For $a = \varepsilon''\delta$, we test $P_{\varepsilon''\delta}$ and $Q_{\varepsilon''\delta}$ for the distance $\delta' := (1+2\varepsilon'')\delta$. We observe that $M_{\delta'}$ is $\mathcal{O}(\frac{t}{\varepsilon''})$-local if P and Q are t-straight. If $\delta_{\mathcal{F}}(P,Q) \leq \delta$, we have $D_{\mathcal{F}}(P_{\varepsilon''\delta},Q_{\varepsilon''\delta}) \leq \delta'$ and if P and Q are $(\varepsilon,\mu\delta)$-far, we have that $P_{\varepsilon''\delta}$ and $Q_{\varepsilon''\delta}$ are $(\frac{\varepsilon}{12},\delta')$-far for $\mu = 1+4\varepsilon'' = 1+\varepsilon'$ for $\varepsilon'' = \varepsilon'/4$. So, if we know t, we can apply the first Fréchet-tester for $\frac{\varepsilon}{12}$ and δ'. If t is unknown, we apply the second Fréchet-tester for the same values.

▶ **Theorem 29.** *Let $\delta,\varepsilon,\varepsilon' > 0$ be given and assume $\varepsilon < 1$. Let P and Q be t-straight curves with $\ell(P) = \ell(Q)$. If t is known, there exists a $(1+\varepsilon')$-approximate Fréchet-tester w.r.t. the continuous Fréchet distance that performs at most $\mathcal{O}\left(\frac{t}{\varepsilon\cdot\varepsilon'}\log\frac{t}{\varepsilon\cdot\varepsilon'}\right)$ queries. If t is unknown, there exists a $(1+\varepsilon')$-approximate Fréchet-tester w.r.t. the continuous Fréchet distance that performs at most $\mathcal{O}\left(\left((\frac{t}{\varepsilon'})^3 + (\frac{t}{\varepsilon'})^2\log\left\lceil\frac{\ell(P)}{\varepsilon'\delta}\right\rceil\right)\frac{\log\log\frac{t}{\varepsilon'}}{\varepsilon}\right)$ queries.*

A natural consequence of the above theorem is that for t-straight curves, the number of queries is bounded in terms of n and the aspect ratio of the input.

References

1 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*, pages 99:1–99:4. ACM, 2017. doi:10.1145/3139958.3140062.

2 Arnab Bhattacharyya and Yuichi Yoshida. *Property Testing: Problems and Techniques.* Springer Nature, 2022.

[3] Note that, in line with their definition, we assume the reparameterizations to be (piecewise) differentiable.

3 Milutin Brankovic, Kevin Buchin, Koen Klaren, André Nusser, Aleksandr Popov, and Sampson Wong. (k, ℓ)-medians clustering of trajectories using continuous dynamic time warping. In *SIGSPATIAL '20: 28th International Conference on Advances in Geographic Information Systems*, pages 99–110. ACM, 2020. doi:10.1145/3397536.3422245.

4 Kevin Buchin, Maike Buchin, David Duran, Brittany Terese Fasy, Roel Jacobs, Vera Sacristán, Rodrigo I. Silveira, Frank Staals, and Carola Wenk. Clustering trajectories for map construction. In *Proceedings of the 25th ACM SIGSPATIAL International Conference on Advances in Geographic Information Systems*, pages 14:1–14:10. ACM, 2017. doi:10.1145/3139958.3139964.

5 Kevin Buchin, Maike Buchin, Joachim Gudmundsson, Jorren Hendriks, Erfan Hosseini Sereshgi, Vera Sacristán, Rodrigo I. Silveira, Jorrick Sleijster, Frank Staals, and Carola Wenk. Improved map construction using subtrajectory clustering. In *LocalRec'20: Proceedings of the 4th ACM SIGSPATIAL Workshop on Location-Based Recommendations, Geosocial Networks, and Geoadvertising*, pages 5:1–5:4. ACM, 2020. doi:10.1145/3423334.3431451.

6 Kevin Buchin, Maike Buchin, and Yusu Wang. Exact algorithms for partial curve matching via the Fréchet distance. In *Proceedings of the twentieth annual ACM-SIAM symposium on Discrete algorithms*, pages 645–654. SIAM, 2009. doi:10.1137/1.9781611973068.71.

7 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*, pages 101:1–101:4. ACM, 2017. doi:10.1145/3139958.3140064.

8 Kevin Buchin, Anne Driemel, Natasja van de L'Isle, and André Nusser. klcluster: Center-based clustering of trajectories. In *Proceedings of the 27th ACM SIGSPATIAL International Conference on Advances in Geographic Information Systems*, pages 496–499. ACM, 2019. doi:10.1145/3347146.3359111.

9 Bernard Chazelle, Ding Liu, and Avner Magen. Sublinear geometric algorithms. In *Proceedings of the thirty-fifth annual ACM symposium on Theory of computing*, pages 531–540, 2003. doi:10.1145/780542.780620.

10 Bernard Chazelle, Ronitt Rubinfeld, and Luca Trevisan. Approximating the minimum spanning tree weight in sublinear time. *SIAM Journal on Computing*, 34(6):1370–1379, 2005. doi:10.1137/S0097539702403244.

11 Artur Czumaj, Funda Ergün, Lance Fortnow, Avner Magen, Ilan Newman, Ronitt Rubinfeld, and Christian Sohler. Approximating the weight of the Euclidean minimum spanning tree in sublinear time. *SIAM Journal on Computing*, 35(1):91–109, 2005. doi:10.1137/S0097539703435297.

12 Artur Czumaj and Christian Sohler. Estimating the weight of metric minimum spanning trees in sublinear time. *SIAM Journal on Computing*, 39(3):904–922, 2009. doi:10.1137/060672121.

13 Artur Czumaj, Christian Sohler, and Martin Ziegler. Property testing in computational geometry. In *European Symposium on Algorithms*, pages 155–166. Springer, 2000. doi:10.1007/3-540-45253-2_15.

14 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*, pages 100:1–100:4. ACM, 2017. doi:10.1145/3139958.3140063.

15 Rogers Epstein and Sandeep Silwal. Property testing of lp-type problems. In *47th International Colloquium on Automata, Languages, and Programming*, volume 168 of *LIPIcs*, pages 98:1–98:18, 2020. doi:10.4230/LIPICS.ICALP.2020.98.

16 Eldar Fischer and Ilan Newman. Testing of matrix properties. In *Proceedings of the thirty-third annual ACM symposium on Theory of computing*, pages 286–295, 2001. doi:10.1145/380752.380812.

17 Oded Goldreich. *Introduction to property testing*. Cambridge University Press, 2017. doi:10.1017/9781108135252.

18 J.E. Goodman, J. O'Rourke, and C.D. Tóth. *Handbook of discrete and computational geometry, third edition*. CRC Press LLC, January 2017. doi:10.1201/9781315119601.

19 Sariel Har-Peled. *Geometric Approximation Algorithms*. American Mathematical Society, USA, 2011.

20 Jirí Matousek. *Lectures on discrete geometry*, volume 212 of *Graduate texts in mathematics*. Springer, 2002.

21 Morteza Monemizadeh. Facility Location in the Sublinear Geometric Model. In *Approximation, Randomization, and Combinatorial Optimization. Algorithms and Techniques (APPROX-/RANDOM 2023)*, volume 275 of *LIPIcs*, pages 6:1–6:24, 2023. doi:10.4230/LIPIcs.APPROX/RANDOM.2023.6.

22 Ketan. Mulmuley. *Computational geometry : an introduction through randomized algorithms*. Prentice-Hall, Englewood Cliffs, N.J, 1994.

23 Dana Ron. Property testing. *Combinatorial Optimization – Dordrecht*, 9(2):597–643, 2001.

24 Dana Ron. Algorithmic and analysis techniques in property testing. *Foundations and Trends® in Theoretical Computer Science*, 5(2):73–205, 2010. doi:10.1561/0400000029.