Extended Abstracts for the 29th Fall Workshop on Computational Geometry

2021

LOW DIMENSIONAL SPACES OF PERSISTENCE DIAGRAMS

DONALD R. SHEEHY AND SIDDHARTH S. SHETH
NORTH CAROLINA STATE UNIVERSITY

Abstract. The space of persistence diagrams under bottleneck distance is known to be high-dimensional. Because many metric search algorithms and data structures have bounds that depend on the dimension of the search space, the high-dimensionality of this space makes it difficult to analyze and compare asymptotic running times of metric search algorithms on this space. In this paper, we explore the dimension of a generalization of the space of PDs. We show that for a class of quotient metrics that includes the bounded persistence plane (i.e., a bounded region of the plane modulo the diagonal) the metric is close in a Gromov-Hausdorff sense to a metric of bounded dimension. We also show how to bound the dimension of bottleneck metrics over k-point subsets of doubling metrics. Finally, we put these together to show that the space of bounded, k-point persistence diagrams is close to a low-dimensional space.

1. Introduction

A persistence diagram (PD) is a topological invariant commonly used in topological data analysis (TDA). Ever since their introduction, PDs have been a popular tool to compare the shapes of point clouds, metric spaces, and real-valued functions.

A significant advantage of PDs over many other topological invariants is that they come equipped with a natural metric, the bottleneck distance, and thus topological features are rendered not only qualitative, but also quantitative. This opens the possibility of doing metric analysis on PDs, such as (approximate) nearest neighbor search or range search.

Many metric proximity search algorithms and data structures have asymptotic running time bounds in terms of the doubling dimension of the search space [2, 3]. The metric space of PDs with the bottleneck distance is known to have infinite doubling dimension[4], making it unclear whether one ought to apply standard data structures such as cover trees[1] or net trees[5] to search in this space. Although the space of all persistence diagrams is infinite-dimensional, there may be reasonable subspaces of persistence diagrams that either are or at least behave like they are low-dimensional. This paper describes some general classes of persistence diagrams that are nearly low-dimensional in the sense that they are close in Gromov-Hausdorff distance to a low-dimensional space (see Theorem 5.1). Along the way, we give general techniques for proving quotient metrics are nearly low-dimensional (Section 3) and a general bound on the doubling dimension of bottleneck spaces over doubling metrics (Section 4)

2. Definitions

2.1. **Metric, Cover, and Dimension.** A *metric space*, $\mathsf{X} = (X, \mathrm{d})$ is a set X and a metric d. The distance between $a \in X$ and a set Y is given by $\mathrm{d}(x, Y) := \inf_{b \in Y} \mathrm{d}(a, b)$. The *diameter* of a set X is $\mathrm{diam}(X) = \sup_{a,b \in X} \mathrm{d}(a, b)$. An *$\varepsilon$-ball centered at a*, denoted by $B(a)$, is the set of all points within ε distance of a.

A collection of sets Y is said to cover X if the union of the sets in Y contains X. A cover Y is an *ε-cover* of X if every set in Y has diameter at most 2ε. An ε-cover Y of X of minimum cardinality is a *minimum ε-cover* and $\mathrm{N}_\varepsilon(X) = |Y|$ is the *covering number* of X. The *ε-metric entropy* of X is defined as $\mathrm{H}_\varepsilon(X) = \log_2 \mathrm{N}_\varepsilon(X)$.

The *doubling constant*, λ, of X is defined as,

$$\lambda = \max_{Z \subseteq X} \mathrm{N}_{\mathrm{diam}(Z)/2}(Z).$$

The *doubling dimension* of X is $\dim(\mathsf{X}) := \log_2(\lambda)$. If $\dim(\mathsf{X})$ is finite, then X is a *doubling metric*. Throughout this paper, all mentions of dimension are referring to the doubling dimension.

2.2. **Quotient Metric Spaces.** Let X be a metric space and let Y be a subspace. The *quotient space* $\mathsf{X}/\mathsf{Y} = (X/Y, \mathrm{d}_{X/Y})$ is defined so that $\mathrm{d}_{X/Y}([a],[b]) := \min\{\mathrm{d}(a,b), \mathrm{d}(a,Y) + \mathrm{d}(b,Y)\}$. There also exists a surjective quotient map, $q : X \to X/Y$ such that $q(x) = [x]$.

The *persistence plane* is the quotient of $(\mathbb{R}^2, \ell_\infty)$ modulo the diagonal. A persistence diagram is a multiset of points in the persistence plane. The dimension is infinite. There is an interesting observation here: a quotient of two low-dimensional metric spaces can be infinite dimensional.

2.3. **Gromov-Hausdorff Distance and Nearly Low-Dimensional Spaces.** For metric spaces $\mathsf{P} = (P, \mathrm{d}_P)$ and $\mathsf{Q} = (Q, \mathrm{d}_Q)$, a *correspondence* between P and Q is a relation $\mathrm{R} \subseteq P \times Q$ such that for its canonical projections on P and Q, we have $\pi_P(\mathrm{R}) = P$ and $\pi_Q(\mathrm{R}) = Q$ respectively. The *distortion* of R is defined as

$$\mathrm{distort}(\mathrm{R}) := \sup_{(p_1,q_1),(p_2,q_2)\in \mathrm{R}} |\mathrm{d}_P(p_1,p_2) - \mathrm{d}_Q(q_1,q_2)|.$$

The *Gromov-Hausdorff distance* d_{GH} is a metric on metric spaces defined as

$$\mathrm{d}_{GH}(\mathsf{P},\mathsf{Q}) := \frac{1}{2}\inf\{\mathrm{distort}(\mathrm{R}) \mid \mathrm{R} \subseteq P \times Q \text{ is a correspondence}\}.$$

A metric space P is *ε-nearly low-dimensional* if there exists a doubling metric space Q such that $\mathrm{d}_{GH}(\mathsf{P},\mathsf{Q}) \leq \varepsilon$.

3. ε-Close Quotient Metric Spaces

In this section show how to approximate a quotient space with a lower dimensional quotient space. We first present a lemma on the dimension of a quotient of a doubling metric modulo finite subset.

Lemma 3.1. *Let X be a metric space with $\dim(\mathsf{X}) = d$. If $Y \subset X$ is finite, then $\dim(\mathsf{X}/\mathsf{Y}) \leq d + \log|Y|$*

Proof. Let $S \subseteq X$ be such that $\mathrm{diam}_{X/Y}(S) = 2\varepsilon$. Let $I : Y \to 2^S$ be a cover of S indexed by the points of Y.

For all $y \in Y$ and $a, b \in I(y)$,

$$\begin{aligned} 2\varepsilon &\geq \mathrm{d}_{X/Y}([a],[b]) \\ &:= \min\{\mathrm{d}(a,b), \mathrm{d}(a,Y) + \mathrm{d}(b,Y)\} \\ &= \min\{\mathrm{d}(a,b), \mathrm{d}(a,y) + \mathrm{d}(b,y)\} \\ &= \mathrm{d}(a,b). \end{aligned}$$

Thus,

$$\mathrm{diam}(I(y)) = \sup_{a,b\in V(y)} \mathrm{d}(a,b) \leq 2\varepsilon.$$

By the definition of doubling dimension, for each $I(y)$ there exists an ε-cover of size at most 2^d sets. Thus, an ε-cover for S can be constructed using the $|Y|$ set $\{I(y) \mid y \in Y\}$ separately. So, we have an ε-cover of S of size at most $|Y|2^d$ and thus,

$$\dim(\mathsf{X}/\mathsf{Y}) \leq d + \log|Y|.$$

□

Lemma 3.2. *For any quotients* $\mathsf{X/Y}$ *and* $\mathsf{X/Y}_\varepsilon$ *over* X*, if* Y_ε *is an* ε*-cover of* Y*, then there exists a correspondence between* $\mathsf{X/Y}$ *and* $\mathsf{X/Y}_\varepsilon$*.*

Proof. Let q and q_ε denote the canonical quotient maps for X/Y and X/Y_ε respectively. Let $\mathrm{R} \subseteq X/Y \times X/Y_\varepsilon$ be a relation such that $\mathrm{R} = \{([a],[b]) \mid \exists x \in X : q(x) = [a], q_\varepsilon(x) = [b]\}$. Because the quotient maps are surjective, it is easy to verify for the canonical projections of R that $\pi_{X/Y}(\mathrm{R}) = X/Y$ and $\pi_{X/Y_\varepsilon}(\mathrm{R}) = X/Y_\varepsilon$. Thus, R is a correspondence. □

Theorem 3.3. *For every quotient* $\mathsf{X/Y}$ *of a compact metric space* X *with doubling dimension* d*, there exists a space of doubling dimension* $d + \mathrm{H}_\varepsilon(Y)$ *that is* ε*-close to* $\mathsf{X/Y}$ *in terms of the Gromov-Hausdorff distance.*

Proof. Let $Y_\varepsilon \subseteq Y$ be such that $\bigcup_{y_i \in Y_\varepsilon} B(y_i)$ is a minimal ε-cover of Y. Then, $\mathrm{H}_\varepsilon(Y) = \log |Y_\varepsilon|$. So, by Lemma 3.1, we know that $\mathsf{X/Y}_\varepsilon$ has doubling dimension at most $d + \mathrm{H}_\varepsilon(Y)$. It suffices to show that $\mathrm{d}_{GH}(\mathsf{X/Y}, \mathsf{X/Y}_\varepsilon) \leq \varepsilon$.

By lemma 3.2, we have a correspondence, R, between $\mathsf{X/Y}$ and $\mathsf{X/Y}_\varepsilon$. For an arbitrary pair $([a],[a]'),([b],[b]') \in \mathrm{R}$, let

$$\begin{aligned}\delta &= |\mathrm{d}_{X/Y}([a],[b]) - \mathrm{d}_{X/Y_\varepsilon}([a]',[b]')| \\ &= |\min\{\mathrm{d}(a,b), \mathrm{d}(a,Y) + \mathrm{d}(b,Y)\} - \min\{\mathrm{d}(a,b), \mathrm{d}(a,Y_\varepsilon) + \mathrm{d}(b,Y_\varepsilon)\}|.\end{aligned}$$

Because $Y_\varepsilon \subset Y$, we have $\mathrm{d}(a,Y) \leq \mathrm{d}(a,Y_\varepsilon) \leq \mathrm{d}(a,Y) + \varepsilon$. Thus, computing distort(R) reduces to the following two cases:

Case. $\mathrm{d}(a,b) \leq \mathrm{d}(a,Y) + \mathrm{d}(b,Y)$

In this case, we have

$$\mathrm{d}_{X/Y}([a],[b]) = \mathrm{d}(a,b), \text{ and } \mathrm{d}_{X/Y_\varepsilon}([a]',[b]') = \mathrm{d}(a,b).$$

Thus, $\delta = 0$.

Case. $\mathrm{d}(a,Y) + \mathrm{d}(b,Y) < \mathrm{d}(a,b)$

On the other hand, if $\mathrm{d}(a,Y) + \mathrm{d}(b,Y) < \mathrm{d}(a,b)$, then,

$$\mathrm{d}_{X/Y}([a],[b]) = \mathrm{d}(a,Y) + \mathrm{d}(b,Y) \text{ and}$$

$$\mathrm{d}_{X/Y_\varepsilon}([a]',[b]') \leq \mathrm{d}(a,Y_\varepsilon) + \mathrm{d}(b,Y_\varepsilon) \leq \mathrm{d}(a,Y) + \mathrm{d}(b,Y) + 2\varepsilon.$$

Thus, $\delta \leq 2\varepsilon$.

Thus, the distortion of R is at most 2ε and hence,

$$\mathrm{d}_{GH}(\mathsf{X/Y}, \mathsf{X/Y}_\varepsilon) \leq \frac{1}{2}\mathrm{distort}(\mathrm{R}) \leq \varepsilon.$$

Because Y is compact, Y_ε is finite and $\mathsf{X/Y}_\varepsilon$ is the required ε-close space with doubling dimension $d + \mathrm{H}_\varepsilon(Y)$.

□

4. d-Dimensional k-Point Diagrams

If the doubling dimension of X is d, then a *d-dimensional k-point diagram* is a set of k distinct elements of X. Let $A = \{a_i\}_{i \in I_k}$ and $B = \{b_i\}_{i \in I_k}$ be d-dimensional k-point diagrams where $I_k = \{1, \ldots, k\}$. The bottleneck of a bijection $\eta : A \to B$ is $\max_{a_i \in A} \mathrm{d}(a_i, \eta(a_i))$. An optimal matching minimizes the bottleneck and the *bottleneck distance* between A and B is $\mathrm{d}_B(a,b) = \min_\eta \max_{a_i \in a} \mathrm{d}(a_i, \eta(a_i))$. Thus, $\mathrm{C}_k^{\mathsf{X}} = (\mathrm{C}_k^X, \mathrm{d}_B)$ is a metric space defined over the set of all d-dimensional k point diagrams and the bottleneck distance of their matchings, d_B.

Theorem 4.1. *Let $\mathsf{X} = (X, \mathrm{d})$ be a d-dimensional doubling metric. Let $\mathrm{C}_\mathsf{k}^\mathsf{X}$ denote the metric space of k-point diagrams in X with the bottleneck metric. Then,* $\dim(\mathrm{C}_\mathsf{k}^\mathsf{X}) \leq kd$.

Proof. Let T be an ε-ball in C_k^X. So by definition, for matchings η,

$$\sup_{A,B \in T} \inf_{\eta : A \to B} \max_i \mathrm{d}(a_i, \eta(a_i)) \leq \varepsilon$$

Therefore, $T = \{\{a_1, \ldots, a_k\} \mid a_i \in B_i\}$ where each B_i is an ε-ball in X. There exists an ε-cover, U_i, for every B_i such that $|U_i| \leq 2^d$.

Moreover, for every $\{a_1, \ldots, a_k\} \in T$ there exists $\{V_1, \ldots, V_k\}$ where each set $V_i \in U_i$ is such that $a_i \in V_i$. Thus $C = \{\{V_1 \ldots V_k\} \mid V_i \in U_i\}$ is an ε-cover of T of size at most 2^{kd}. So $\dim(\mathrm{C}_\mathsf{k}^\mathsf{X}) \leq kd$. □

5. Space of Bounded Persistence Diagrams

From the preceding two sections we get an approximation of single-class quotient spaces and a bound on the doubling dimension of finite point bottleneck spaces respectively. These results come together in the space of bounded persistence diagrams to form a nearly low dimensional subspace of persistence diagrams.

Let $\mathsf{D} = (\mathbb{R}^2, \ell_\infty)$. Denoting G as the diagonal from $(0,0)$ to (N,N), let $\mathsf{D}_\mathsf{N}/\mathsf{G} = (D_N/G, \mathrm{d}_{D_N/G})$ be the N-bounded persistence plane. By definition, $\mathrm{d}_{D_N/G}([a],[b]) = \min\{\ell_\infty(a,b), \ell_\infty(a,G) + \ell_\infty(b,G)\}$. Let $\mathrm{C}_\mathsf{k}^{\mathsf{D}_\mathsf{N}/\mathsf{G}} = (\mathrm{C}_k^{D_N/G}, \mathrm{d}_B)$ be the bottleneck space of all k-point persistence diagrams bounded by N and let $\mathrm{C}_\mathsf{k}^{\mathsf{D}_\mathsf{N}/\mathsf{G}_\varepsilon} = (\mathrm{C}_k^{D_N/G_\varepsilon}, \mathrm{d}_B)$ denote the k-point bottleneck space in the approximate N-bounded k-point persistence plane where G_ε is the minimum ε-cover of G.

Theorem 5.1. *The space of k-point N-bounded persistence diagram is ε-close to a space of doubling dimension at most $(2 + \log\lceil N/2\varepsilon \rceil)k$ in terms of the Gromov-Hausdorff distance.*

Proof. Because $\dim(\mathsf{D}) = 2$ and $|G_\varepsilon| = \lceil N/2\varepsilon \rceil$, from theorem 4.1, we know that $\dim(\mathrm{C}_\mathsf{k}^{\mathsf{D}_\mathsf{N}/\mathsf{G}_\varepsilon}) \leq (2 + \log\lceil N/2\varepsilon \rceil)k$. To show that this approximate persistence plane is ε-close the bounded persistence plane we use a technique similar to theorem 3.3.

By lemma 3.2, we know that the correspondence R between $\mathsf{D}_\mathsf{N}/\mathsf{G}$ and $\mathsf{D}_\mathsf{N}/\mathsf{G}_\varepsilon$ has distortion at most 2ε. Let R^k denote the correspondence between $\mathrm{C}_\mathsf{k}^{\mathsf{D}_\mathsf{N}/\mathsf{G}}$ and $\mathrm{C}_\mathsf{k}^{\mathsf{D}_\mathsf{N}/\mathsf{G}_\varepsilon}$ defined as follows:

$$\mathrm{R}^k = \{(\{a_1, \ldots, a_k\}, \{b_1, \ldots, b_k\}) \mid \exists m : I_k \to I_k, \forall i : (a_i, b_{m(i)}) \in \mathrm{R}\}.$$

To show that $\mathrm{d}_{GH}(\mathrm{C}_\mathsf{k}^{\mathsf{D}_\mathsf{N}/\mathsf{G}_\varepsilon}, \mathrm{C}_\mathsf{k}^{\mathsf{D}_\mathsf{N}/\mathsf{G}}) \leq \varepsilon$, it is sufficient to bound the distortion of R^k.

Let (A, B) and (A', B') be arbitrary pairs in the R^k, where $A = \{a_i\}_{i \in I_k}$, $A' = \{a'_i\}_{i \in I_k}$, $B = \{b_i\}_{i \in I_k}$, and $B' = \{b'_i\}_{i \in I_k}$. Without loss of generality, we may assume they are indexed so that for all j, we have $(a_j, b_j) \in \mathrm{R}$ and $(a'_j, b'_j) \in \mathrm{R}$. Let $\eta : I_k \to I_k$ be the permutation of indices that gives the bottleneck matching between A and A', i.e.,

$$\mathrm{d}_B(A, A') = \max_{i \in I_k} \mathrm{d}_{D^N/G}(a_i, a'_{\eta(i)}).$$

It follows from the distortion bound on R that

$$\begin{aligned} \mathrm{d}_B(B, B') &\leq \max_{j \in I_k} \mathrm{d}_{D^N/G_\varepsilon}(b_j, b'_{\eta(j)}) \\ &\leq \max_{j \in I_k} (\mathrm{d}_{D^N/G}(a_j, a'_{\eta(j)}) + 2\varepsilon) \\ &= \mathrm{d}_B(A, A') + 2\varepsilon. \end{aligned}$$

Symmetrically, we have $\mathrm{d}_B(A, A') \leq \mathrm{d}_B(B, B') + 2\varepsilon$ and thus, $\mathrm{distort}(R^k) \leq 2\varepsilon$ as desired. □

Thus, the space of bounded k-point persistence diagrams is nearly low-dimensional.

References

[1] A. Beygelzimer and J. Langford. Cover trees for nearest neighbor. pages 97–104, 2006.

[2] K. Clarkson. Nearest-Neighbor Searching and Metric Space Dimensions. In G. Shakhnarovich, T. Darrell, and P. Indyk, editors, *Nearest-Neighbor Methods in Learning and Vision*. The MIT Press, 2006.

[3] A. Efrat, A. Itai, and M. J. Katz. Geometry Helps in Bottleneck Matching and Related Problems. *Algorithmica*, 31(1):1–28, Sept. 2001.

[4] B. T. Fasy, X. He, Z. Liu, S. Micka, D. L. Millman, and B. Zhu. Approximate Nearest Neighbors in the Space of Persistence Diagrams. *arXiv:1812.11257 [cs]*, Mar. 2021. arXiv: 1812.11257.

[5] S. Har-Peled and M. Mendel. Fast Construction of Nets in Low-Dimensional Metrics and Their Applications. *SIAM Journal on Computing*, 35(5):1148–1184, Jan. 2006.

Large sample spectral analysis of graph-based multi-manifold clustering

Nicolas Garcia Trillos, Pengfei He, Chenghui Li*

Abstract

In this work[1] we study statistical properties of graph-based algorithms for multi-manifold clustering (MMC). In MMC the goal is to retrieve the multi-manifold structure underlying a given Euclidean data set when this one is assumed to be obtained by sampling a distribution on a union of manifolds $\mathcal{M} = \mathcal{M}_1 \cup \cdots \cup \mathcal{M}_N$ that may intersect with each other and that may have different dimensions. We investigate sufficient conditions that similarity graphs on data sets must satisfy in order for their corresponding graph Laplacians to capture the right geometric information to solve the MMC problem. Precisely, we provide high probability error bounds for the spectral approximation of a tensorized Laplacian on $\mathcal{M}$ with a suitable graph Laplacian built from the observations; the recovered tensorized Laplacian contains all geometric information of all the individual underlying manifolds. We provide an example of a family of similarity graphs, which we call annular proximity graphs with angle constraints, satisfying these sufficient conditions. We contrast our family of graphs with other constructions in the literature based on the alignment of tangent planes. Extensive numerical experiments expand the insights that our theory provides on the MMC problem.

Keywords: multi-manifold clustering, graph Laplacian, spectral convergence, manifold learning, discrete to continuum limit.

1. Introduction

In this work we study the problem of *multi-manifold clustering* (MMC) from the perspective of spectral geometry. Multi-manifold clustering is the task of identifying the structure of multiple manifolds that underlie an observed data set $X = \{x_1, \dots, x_n\}$, its main challenge being that in general the underlying manifolds may be non-linear, may intersect with each other, and may have different dimensions (see Figures 1-3 for some illustrations). While spectral methods for learning have been analyzed by several authors throughout the past two decades in settings as varied as unsupervised, semi-supervised, and supervised learning, less is known about their theoretical guarantees for the specific multi-manifold clustering problem. We analyze MMC algorithms that are based on the construction of suitable similarity graph representations for the data and in turn on the spectra of their associated graph Laplacians. We provide statistical error guarantees for the identification of the underlying manifolds as well as for the recovery of their individual geometry.

As for most spectral approaches to clustering, we are interested in studying spectral properties of graph Laplacian operators of the form

$$\Delta_n u(x_i) := \sum \omega_{ij}(u(x_i) - u(x_j)), \quad x_i \in X. \tag{1.1}$$

Here, the ω_{ij} are appropriately defined symmetric weights that in general depend on the proximity of points x_i, x_j, and importantly, on a mechanism that detects when points

*. Corrsponding Author:
Email Address: cli539@wisc.edu(Chenghui Li)

1. Submitted to JMLR, preprint see https://arxiv.org/abs/2107.13610

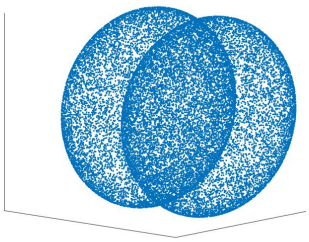


Figure 1

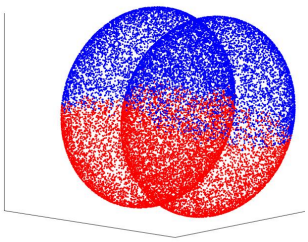


Figure 2

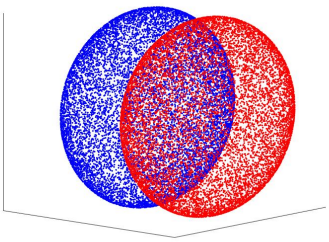


Figure 3

Figure 1 illustrates two intersecting ellipsoids (two dimensional). A *good* multi-manifold clustering algorithm must identify the two underlying ellipsoids. Figure 2 and Figure 3 show the spectral clustering with standard ε-proximity graph and annular proximity graph with angle constraint, respectively; see following discussion.

belong to different manifolds even if lying close to each other. Once the graph Laplacian is constructed we follow the spectral clustering algorithm: the first p eigenvectors of Δ_n (denoted $\psi_1, \ldots, \psi_N$) are used to build an embedding of the data set X into $\mathbb{R}^p$:

$$x_i \in X \longmapsto \begin{pmatrix} \psi_1(x_i) \\ \vdots \\ \psi_N(x_i) \end{pmatrix} \in \mathbb{R}^p.$$

In turn, with the aid of a simple clustering algorithm such as k-means the embedded data set is clustered. A successful algorithm will produce clusters that are in agreement with the different manifolds underlying the data set. It is essential to select the weight ω_{ij} to make spectral clustering algorithm work well as standard proximity graphs do not work well in MMC problem. See Figure 1-3 as an illustration.

2. Setup

To start making our results more precise, let us suppose that the data set X is obtained by sampling a distribution μ supported on a set $\mathcal{M}$ of the form $\mathcal{M} = \mathcal{M}_1 \cup \cdots \cup \mathcal{M}_N$ where the $\mathcal{M}_l$ are smooth compact connected manifolds with no boundary that for the moment are assumed to have the same dimension m; the manifolds $\mathcal{M}_l$ may have nonempty pairwise intersections, but these are assumed to have measure zero relative to the volume forms of each of the manifolds involved. The distribution μ is assumed to be a mixture model taking the form

$$\mu = w_1\rho_1 d\mathrm{vol}_{\mathcal{M}_1} + \cdots + w_N\rho_N d\mathrm{vol}_{\mathcal{M}_N} \tag{2.1}$$

for smooth density functions $\rho_l : \mathcal{M}_l \to \mathbb{R}$ and positive weights w_i that add to one; henceforth we use $d\mathrm{vol}_{\mathcal{M}_l}$ to denote the integration with respect to the Riemannian volume form associated to $\mathcal{M}_l$. A *tensorized Laplacian* $\Delta_{\mathcal{M}}$ acting on functions f on $\mathcal{M}$ (which will

be written as $f = (f_1, \ldots, f_N)$, where $f_l : \mathcal{M}_l \to \mathbb{R}$) can be defined according to

$$\Delta_{\mathcal{M}} f := (w_1 \Delta_{\mathcal{M}_1} f_1, \ldots, w_N \Delta_{\mathcal{M}_N} f_N), \tag{2.2}$$

where $\Delta_{\mathcal{M}_l}$ is a Laplacian operator mapping regular enough functions $f_l : \mathcal{M}_l \to \mathbb{R}$ into functions $\Delta_{\mathcal{M}} f_l : \mathcal{M}_l \to \mathbb{R}$ according to

$$\Delta_{\mathcal{M}_l} f_l = -\frac{1}{\rho_l} \mathrm{div}_{\mathcal{M}_l} \left(\rho_l^2 \nabla_{\mathcal{M}_l} f_l \right).$$

In other words, the operator $\Delta_{\mathcal{M}}$ acts in a coordinatewise fashion effectively treating each manifold $\mathcal{M}_i$ independently. It is then straightforward to show that eigenfunctions of $\Delta_{\mathcal{M}}$ are spanned by functions of the form $(0, \ldots, f_l, \ldots, 0)$ for some l, where f_l is an eigenfunction of $\Delta_{\mathcal{M}_l}$. This means that the spectrum of $\Delta_{\mathcal{M}}$ splits the geometries of the $\mathcal{M}_l$, and in particular, the different $\mathcal{M}_l$ can be detected by retrieving the eigenfunctions with zero eigenvalue.

3. Result

Let's begin with two sufficient conditions that the weighted graph must satisfy to solve MMC problem.

Definition 3.1 (Fully inner Connected graphs)**.** *Let $X = x_1, \ldots, x_n$ be samples from μ as defined in* (2.1)*. A weighted graph (X, ω) is said to be fully inner connected relative to ε_+ and ε_- which converge to zeros as $n \to \infty$ if with probability $1 - C_1(n)$, where $C_1(n) \to 0$ as $n \to \infty$, for any pair of points x_i, x_j belonging to the same manifold $\mathcal{M}_k$ we have $\omega_{x_i,x_j} = \omega^{\varepsilon_+,\varepsilon_-}_{x_i,x_j}$.*

Definition 3.2 (Sparsely Outer Connected graphs)**.** *Let $X = x_1, \ldots, x_n$ be samples from μ as defined in* (2.1)*, and let (X, ω) be a weighted graph. Let N_{sl} be the number of connections between $x_i \in \mathcal{M}_s$ and $x_j \in \mathcal{M}_l$ such that $\omega_{ij} > 0$, and let*

$$N_0 := \max_{l \neq s} \{N_{ls}\}.$$

The graph is said to be sparsely outer connected relative to ε_+ and ε_- converging to zero as $n \to \infty$ if with probability one, $\frac{N_0}{n^2(\varepsilon_+^{m+2} - \varepsilon_-^{m+2})} \to 0$ as $n \to \infty$. We recall that $m = \max_{l=1,\ldots,N} m_l$.

Full inner connectivity condition guarantees that points within one manifold connect to each other with high probability, and sparse outer connectivity condition guarantees that the number of connections between points from different manifolds cannot be too large.

Our first main results (**Theorem 2.5 and Theorem 2.7**) say that provided that the weights ω_{ij} defining the graph Laplacian operator Δ_n in (1.1) satisfy full inner connectivity and sparse outer connectivity, then the eigenvalues (appropriately scaled) and eigenvectors of Δ_n approximate the eigenvalues and eigenfunctions of the tensorized Laplacian $\Delta_{\mathcal{M}}$; we obtain high probability quantitative bounds for the error of this approximation. The bottom line is that our results imply that the spectral methods studied here are guaranteed, at least for large enough n, to recover the underlying multi-manifold structure of the data;

see Figure 3 for an illustration. Our work extends the growing literature of works that study the connection between graph Laplacians on data sets and their continuum analogues. This literature has mostly focused on the smooth setting where multiple intersecting manifolds are not allowed.

In our second main result (**Theorem 2.8**) we present some results for the case when the dimensions of the manifolds $\mathcal{M}_i$ do not agree. In this more general setting, the spectrum of the graph Laplacian Δ_n does not recover the tensorized geometry captured by $\Delta_{\mathcal{M}}$ as introduced earlier, but rather, only the tensorized geometry of the manifolds with the *largest* dimension, effectively quotienting out the geometric information of manifolds with dimension strictly smaller than the maximum dimension. Detailed discussions and proofs can be seen in Trillos et al. (2021).

The theory above showed that if two sufficient conditions are satisfied for the weighted graph, then spectral clustering is guaranteed to be consistent for MMC. In the following, We also present a graph construction that we refer to as annular proximity graph with angle constraint that satisfies full inner connectivity and sparse outer connectivity conditions. Numerical experiments support and expand our insights on the algorithm's behavior.

4. Contribution and Discussion

- We analyze graph Laplacians on families of proximity graphs when the nodes of the graphs are random data points that are supported on a union of unknown *intersecting* manifolds. The manifolds may all have *different* dimensions.

- We introduce two sufficient conditions that similarity graphs must satisfy in order to recover, from a graph Laplacian operator, the geometric information of the individual smooth manifolds underlying the data set. These conditions are referred to as *full inner connectivity* and *sparse outer connectivity*.

- We introduce and analyze *annular* proximity graphs and their effect on multi-manifold clustering. These are simple extensions of ε-proximity graphs that nonetheless can be shown to be, theoretically and numerically, better than the vanilla ε-graphs for multi-manifold clustering.

- We analyze a family of *annular proximity graphs with angle constraints*. This family is shown to satisfy the full inner connectivity and sparse outer connectivity conditions when their parameters are tuned appropriately. We contrast this construction with other constructions such as those based on local PCA which in general do not satisfy the full inner connectivity condition.

- Through numerical examples and some heuristic computations we provide further insights into the use of spectral methods for multi-manifold clustering.

References

Nicolas Garcia Trillos, Pengfei He, and Chenghui Li. Large sample spectral analysis of graph-based multi-manifold clustering, 2021.

Quantifying barley morphology using the Euler Characteristic Transform

Erik J. Amézquita[1], Michelle Y. Quigley[2], Tim Ophelders[4], Jacob B. Landis[5,6], Daniel Koenig[6], Daniel H. Chitwood[1,2], and Elizabeth Munch[1,3]

[1]*Dept. of Computational Mathematics, Science, & Engineering, Michigan State University*
[2]*Dept. of Horticulture, Michigan State University*
[3]*Dept. of Mathematics, Michigan State University*
[4]*Dept. of Mathematics and Computer Science, TU Eindhoven*
[5]*School of Integrative Plant Science, Cornell University*
[6]*Dept. of Botany & Plant Sciences, University of California—Riverside*

Abstract

Observing and documenting shape has fueled biological understanding as the shape of biomolecules, cells, tissues, and organisms arise from the effects of genetics, development, and the environment. The vision of Topological Data Analysis (TDA), that data is shape and shape is data, will be relevant as biology transitions into a data-driven era where meaningful interpretation of large datasets is a limiting factor. We focus first on quantifying the morphology of X-ray CT scans of barley spikes and seeds using topological descriptors based on the Euler Characteristic Transform. We then successfully train a support vector machine to distinguish and classify 28 different varieties of barley based solely on the 3D shape of their grains. This shape characterization will allow us later to link genotype with phenotype, furthering our understanding on how the physical shape is genetically coded in DNA. (This work has been accepted in *inSilico Plants*.)

1 Introduction

Biologists are accustomed to thinking about how the shape of biomolecules, cells, tissues, and organisms arise from the effects of genetics, development, and the environment. Traditionally, biologists use morphometrics to compare and describe shapes. The shape of leaves and fruits is quantified based on homologous landmarks—similar features due to shared ancestry from a common ancestor—or harmonic series from a Fourier decomposition of their closed contour. While these methods are useful for comparing many shapes in nature, they cannot always be used: there may not be homologous points between samples or a harmonic decomposition of a shape is not appropriate. Topological data analysis (TDA) offers a more comprehensive, versatile way to quantify plant morphology. In particular, Euler characteristic curves (ECC) serve as a succinct, computationally feasible topological signature that allows downstream statistical analyses [12]. For example, ECCs have been used to determine a morphospace for all leaves to then predict plant family and location [5]. Further analysis has determined the genetic basis of 2D leaf shape in apple [9], tomato [6], and cranberry [3]. Here, we show the use of the Euler Characteristic to comprehensively describe the shape of 3D voxel-based X-ray CT scans of barley seeds as a proof of concept.

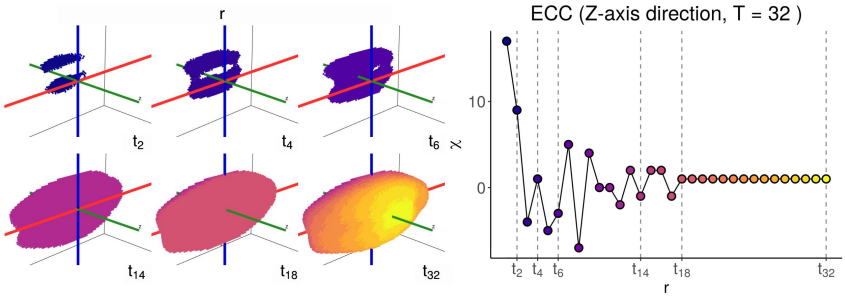


Figure 1: Filtration of a barley seed along the z-axis with 32 thresholds and its corresponding Euler Characteristic Curve.

Table 1: SVM classification accuracy of barley seeds from 28 different accessions after 100 randomized training and testing sets. Classification scores was computed for each accession; the weighted average for each score was taken afterwards, where the weight depended on the number of test seeds used. The use of topological outperforms the use of exclusively traditional descriptors.

Shape descriptors	Dimension reduction	No. of dims	Scores (weighted average $\pm$ standard deviation) Precision	Recall	F_1
Traditional	*	11	0.58 ± 0.050	0.58 ± 0.016	0.57 ± 0.016
Topological	KPCA	2	0.88 ± 0.031	0.87 ± 0.010	0.87 ± 0.011
Topological	UMAP	12	0.75 ± 0.047	0.75 ± 0.016	0.74 ± 0.016
Combined	KPCA	13	0.73 ± 0.052	0.72 ± 0.017	0.71 ± 0.017
Combined	UMAP	23	0.89 ± 0.028	0.89 ± 0.010	0.89 ± 0.010

2 Methods

Consider a cubical complex X of dimension d. For a fixed direction $\nu \in S^{d-1}$, and a height value $h \in \mathbb{R}$, we define

$$X(\nu)_h = \{\Delta \in X \ : \ \langle x, \nu \rangle \leq h \text{ for all } x \in \Delta\},$$

to be the subcomplex containing all cubical cells below height h in the direction ν. The Euler characteristic at height h is $\chi(X(\nu)_h)$, the alternating sum of counts of cells in the subcomplex $X(\nu)_h$. The Euler Characteristic Curve (ECC) of direction ν is defined as $\{\chi(X(\nu)_h)\}_{h\in\mathbb{R}}$, exemplified in Figure 1. The Euler Characteristic Transform (ECT) is defined as the collection of all ECCs corresponding to all possible directions. To be more precise, the ECT of complex X is defined as the function

$$\begin{aligned} ECT(X) : S^{d-1} &\to \mathbb{Z}^{\mathbb{R}} \\ \nu &\mapsto \{\chi(X(\nu)_h)\}_{h\in\mathbb{R}}. \end{aligned}$$

We focus on the shape of barley seeds from a collection of 28 different barley accessions from diverse regions across the Eurasian continent. Using X-ray CT—computed tomography—scanning technology, we have created voxel-based 3D reconstructions of over 875 spikes, from which we have isolated 3121 parental seeds. Since the seeds are oblong in shape, we aligned them according to their three main principal components.

On one hand, we computed 11 traditional quantifiable shape descriptors for each seed, such as length, width, and volume. On the other hand, we computed topological shape descriptors using the ECT. For topological purposes, we treated each voxel-based image as a dual cubical complex where each nonzero voxel is treated as a vertex [13].

We favor the use of the ECT for two reasons. First, the ECT is computationally inexpensive; since it is based on successive alternating sums of counts of cells, a single ECC

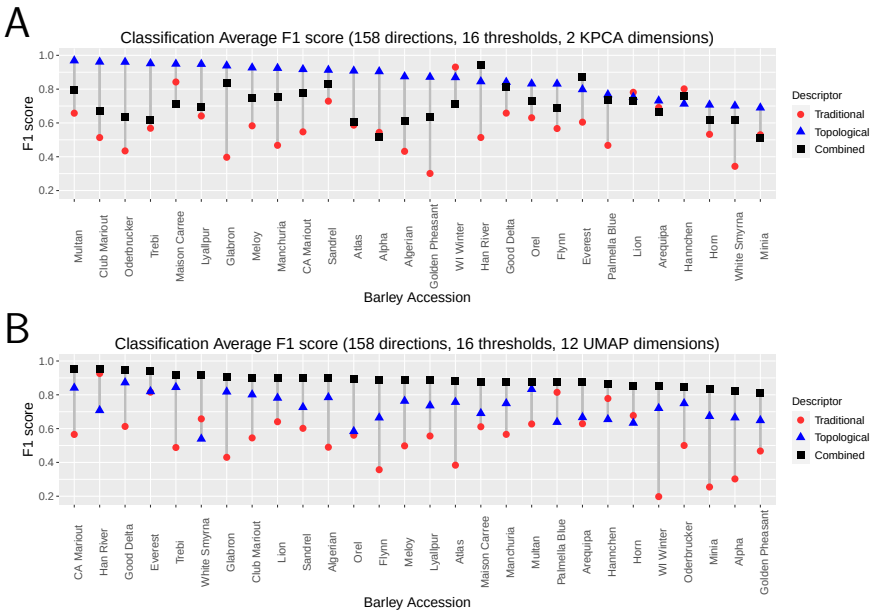


Figure 2: SVM classification results for individual barley accessions. A. Results when using a KPCA 2-dimension reduced topological vector. Accessions ordered according to their classification accuracy determined by the topological shape descriptors. **B.** Results when using a UMAP 12-dimension reduced topological vector. Accessions ordered according to their classification accuracy determined by the combined shape descriptors.

can be computed in linear time with respect to the number of voxels in the image [10]. Second, the ECT effectively summarizes all the morphological features of any 3D complex as it encodes sufficient information to reconstruct the initial complex [12], a result later extended to the n-dimensional case [2]. Nonetheless, the idea of efficiently reconstructing an arbitrary 3D object solely from its ECT remains elusive [1, 4, 8].

In our case, we used 158 different directions with 16 uniformly spaced thresholds. We emphasized directions toward the crease of the seeds. This yielded a 2528-dimensional vector for every seed. These high-dimensional vectors were later reduced in dimension separately using non-linear KPCA [11] and UMAP [7].

The descriptiveness of both traditional and topological measures was tested by training five non-linear support vector machines (SVM). These characterized and predicted the seeds from the 28 distinct barley accessions based on three different collection of descriptors: traditional, topological, and combined. We also varied the dimension-reduction method. In every case, all the descriptors were centered and scaled to variance 1 prior to classification. Given that SVM is a supervised learning method, we first randomly sampled 75% of the seeds from every founder as our training data set. The remaining 25% was used to test the accuracy of our prediction model. This setup was repeated 100 times. Average scores were considered for the overall data set (Table 1), as well as for individual accessions (Fig. 2).

3 Results and Conclusions

The majority of the barley accessions studied are more easily distinguished with the topological lens but not with traditional measures, with few exceptions (Fig. 2). Exceptions like Hannchen, Han River and Palmella Blue have slightly distinctive traditional trait distributions, so seed size does matter and it is important to take it into account. At the same time,

we observe accessions like Alpha, Glabron, Minia, and Wisconsin Winter, that are poorly differentiated with traditional information but report considerably higher classification accuracies whenever using topological information. When looking at a more robust dimension reduction technique like UMAP, classification accuracy is increased when combined with size-related information.

The Euler characteristic is a simple yet powerful way to reveal features not readily visible to the naked eye. There is "hidden" morphological information that traditional and geometric morphometric methods are missing. The Euler characteristic, and Topological Data Analysis in general, can be readily computed from any given image data, which makes it a versatile tool to use in a vast number of biology-related applications. TDA provides a comprehensive framework to detect and compare morphological nuances, nuances that traditional measures fail to capture and that remain unexplored using simple geometric methods. In the specific case of barley seeds presented here, these "hidden" shape nuances provides enough information to not only characterize specific accessions, but the individual spikes from which seeds are derived. Our results suggest a new exciting path, driven by morphological information alone, to explore further the phenotype-genotype relationship.

4 Literature cited

[1] **Betthauser LM**. *Topological reconstruction of grayscale images*. PhD thesis, University of Florida, Gainesville, Florida (2018).

[2] **Curry J, Mukherjee S, Turner K** (2018). How many directions determine a shape and other sufficiency results for two topological transforms. arXiv:1805.09782.

[3] **Diaz-Gárcia L, Covarrubias-Pazaran G, Schlautman B, Grygleski E, Zalapa J** (2018). Image-based phenotyping for identification of QTL determining fruit shape and size in american cranberry (*Vaccinium macrocarpon L.*). PeerJ **6**. doi:10.7717/peerj.5461.

[4] **Fasy BT, Micka S, Millman DL, Schenfisch A, Williams L** (2019). The first algorithm for reconstructing simplicial complexes of arbitrary dimension from persistence diagrams. arXiv:1912.12759.

[5] **Li M, An H, Angelovici R, Bagaza C, Batushansky A, Clark L, Coneva V, Donoghue MJ, Edwards E, Fajardo D, et al.** (2018). Topological data analysis as a morphometric method: Using persistent homology to demarcate a leaf morphospace. Frontiers in Plant Science **9**: 553. doi:10.3389/fpls.2018.00553.

[6] **Li M, Frank MH, Coneva V, Mio W, Chitwood DH, Topp CN** (2018). The persistent homology mathematical framework provides enhanced genotype-to-phenotype associations for plant morphology. Plant Physiology **177**: 1382–1395. doi:10.1104/pp.18.00104.

[7] **McInnes L, Healy J, Melville J** (2020). UMAP: Uniform Manifold Approximation and Projection for Dimension Reduction. arXiv:1802.03426.

[8] **Micka SA**. *Searching and Reconstruction: Algorithms with Topological Descriptors*. PhD thesis, Montana State University, Bozeman, Montana (2020).

[9] **Migicovsky Z, Li M, Chitwood DH, Myles S** (2018). Morphometrics reveals complex and heritable apple leaf shapes. Frontiers in Plant Science **8**: 2185. doi:10.3389/fpls.2017.02185.

[10] **Richardson E, Werman M** (2014). Efficient classification using the Euler characteristic. Pattern Recognition Letters **49**: 99 – 106. doi:10.1016/j.patrec.2014.07.001.

[11] **Schölkopf B, Smola A, Müller KR** (1998). Nonlinear component analysis as a kernel eigenvalue problem. Neural Computation **10**: 1299–1319. doi:10.1162/089976698300017467.

[12] **Turner K, Mukherjee S, Boyer DM** (2014). Persistent homology transform for modeling shapes and surfaces. Information and Inference **3**: 310–344. doi:10.1093/imaiai/iau011.

[13] **Wagner H, Chen C, Vuçini E**, Wagner, Hubert and Chen, Chao and Vuçini, Erald. Efficient computation of persistent homology for cubical data. In Peikert R, Hauser H, Carr H, Fuchs R, editors, *Topological Methods in Data Analysis and Visualization II: Theory, Algorithms, and Applications*, pages 91–106. publisher. doi:10.1007/978-3-642-23175-9_7.

The Two-Squirrel Problem and Its Relatives

Sergey Bereg[1], Manuel Lafond[2], and Binhai Zhu[3]

[1] Department of Computer Science, University of Texas at Dallas, Richardson, TX 75080, USA.
besp@utdallas.edu
[2] Department of Computer Science, Université de Sherbrooke, Sherbrooke, Quebec J1K 2R1, Canada. manuel.lafond@usherbrooke.ca
[3] Gianforte School of Computing, Montana State University, Bozeman, MT 59717, USA.
bhz@montana.edu

Abstract. In this paper, we start with a variation of the star cover problem called the Two-Squirrel problem. Given a set P of $2n$ points in the plane, and two sites c_1 and c_2, compute two n-stars S_1 and S_2 centered at c_1 and c_2 respectively such that the maximum weight of S_1 and S_2 is minimized. This problem is strongly NP-hard by a reduction from Equal-size Set-Partition with Rational Numbers. Then we consider two variations of the Two-Squirrel problem, namely the Dichotomy Two-Squirrel problem and the Two-MST problem, which are both strongly NP-hard. In terms of approximation algorithms, in fact Two-Squirrel and Dichotomy Two-Squirrel both admit a full PTAS (FPTAS) using the traditional methods. For Two-MST, the scenario is quite different and we are only able to obtain a factor-4.8536 approximation.

Keywords: Minimum star/tree cover · NP-hardness · Set-Partition · Approximation algorithms · Minimum spanning tree (MST)

1 Introduction

Imagine that two squirrels try to fetch and divide $2n$ nuts to their nests. Since each time a squirrel can only carry a nut back, this naturally gives the following problem: they should travel along the edges of an n-star, centered at the corresponding nest, such that each leaf (e.g., nut) is visited exactly once (in and out) and the maximum distance they visit should be minimized (assuming that they travel at the same speed, there is no better way to enforce the fair division under such a circumstance). See Figure 1 for an illustration.

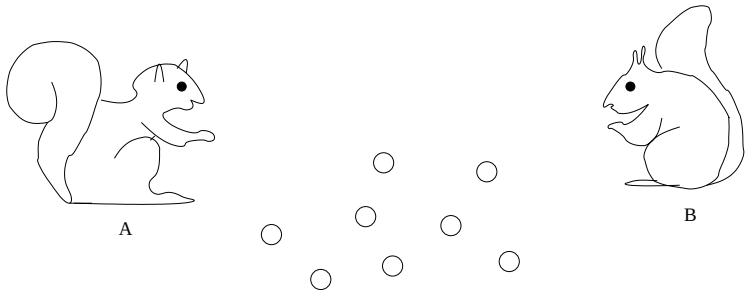


Fig. 1: Two squirrels A and B try to fetch and divide $2n$ nuts.

A star S is a tree where all vertices are leaves except one (which is called the *center* of the star). An n-star is a star with n leaf nodes. When the edges in S carry weights, the weight of S is the sum of weights of all the edges in S. Given two points p, q in the

plane, with $p = (x_p, y_p)$ and $q = (x_q, y_q)$, we define the Euclidean distance between p, q as $d(p,q) = \sqrt{(x_p - x_q)^2 + (y_p - y_q)^2}$.

Formally, the *Two-Squirrel* problem can be defined as: Given a set P of $2n$ points in the plane and two extra point sites c_1 and c_2, compute two n-stars S_1 and S_2 centered at c_1 and c_2 respectively such that each point $p_j \in P$ is a leaf in exactly one of S_1 and S_2; moreover, the maximum weight of S_1 and S_2 is minimized. Here the weight of an edge (c_i, p_j) in S_i is $w(c_i, p_j) = d(c_i, p_j)$ for $i = 1, 2$.

One can certainly consider a variation of the two-squirrel problem where the points are given as pairs (p_{2i-1}, p_{2i}) for $i = 1, ..., n$, and the problem is to split all the pairs (i.e., one to c_1 and the other to c_2) such that maximum weight of the two resulting stars is minimized. We call this version *Dichotomy Two-Squirrel*. A more general version of the problem is when the two squirrels only need to split the $2n$ nuts and each could travel along a Minimum Spanning Tree (MST) of the n points representing the locations of the corresponding nuts, which we call the *Two-MST* problem: Compute a partition of P into n points each, P_1 and P_2, such that the maximum weight of the MST of $P_1 \cup \{c_1\}$ and $P_2 \cup \{c_2\}$, i.e., $\max\{w(P_1 \cup \{c_1\}), w(P_2 \cup \{c_2\})\}$, is minimized.

Covering a (weighted) graph with stars or trees (to minimize the maximum weight of them) is a well-known NP-hard problem in combinatorial optimization [2], for which constant factor approximation is known. Recently, bi-criteria approximations are also reported [3]. In the past, a more restricted version was also investigated on graphs [7]. Our Two-Squirrel problem can be considered a special geometric star cover problem where the two stars are disjoint though are of the same cardinality, and the objective function is also to minimize the maximum weight of them.

It turns out that both Two-Squirrel and Dichotomy Two-Squirrel are strongly NP-hard (under both the Euclidean and L_1 metric, though we focus only on the Euclidean case in this paper). The proofs can be directly from two variations of the famous Set-Partition problem [4, 5], namely, Equal-Size Set-Partition for Rationals and Dichotomy Set-Partition for Rationals, which are both strongly NP-hard with the recent result by Wojtczak [6]. We then show that Dichotomy Set-Partition for Rationals can be reduced to Two-MST in polynomial time, which indicates that Two-MST is also strongly NP-hard.

For the approximation algorithms, both Two-Squirrel and Dichotomy Two-Squirrel admit a FPTAS (note that this does not contradict the known result that a strongly NP-hard problem with an integral objective function cannot be approximated with a FPTAS unless P=NP, simply because our objective functions are not integral). This can be done by first designing a polynomial-time dynamic programming algorithm through scaling and rounding the distances to integers, obtaining the corresponding optimal solutions, and then tracing back to obtain the approximate solutions. The approximation algorithm for Two-MST is more tricky; in fact, with a known lower bound by Chung and Graham related to the famous Steiner Ratio Conjecture [1], we show that a factor 4.8536 approximation can be obtained.

In the next section, we give details for our results for the Two-MST problem. In Section 3, we conclude the paper.

2 Results for the Two-MST Problem

2.1 Preliminaries

In this section, we first define Equal-size Set-Partition for Rationals and Dichotomy Set-Partition for Rationals which are generalizations of Set-Partition [4, 5].

In Dichotomy Set-Partition with Rationals, we are given a set E of $2n$ positive rationals numbers (rationals, for short) with $E = E'_1 \cup E'_2 \cup \cdots E'_n$ such that $E'_i = \{a_{i,1}, a_{i,2}\}$ is a

2-set (or, $E'_i = (a_{i,1}, a_{i,2})$, i.e., as a pair) and the problem is to decide whether E can be partitioned into E_1 and E_2 such that every two elements in E'_i is partitioned into E_1 and E_2 (i.e., one in E_1 and the other in E_2 — clearly $|E_1| = |E_2| = n$) and $\sum_{a \in E_1} a = \sum_{b \in E_2} b$. (Equal-size Set-Partition with Rationals is simply a special case of Dichotomy Set-Partition with Rationals where E is given as a set of $2n$ rationals, i.e., $E = \{a_1, a_2, \cdots, a_{2n}\}$ and E'_i's are not given.)

With integer inputs, both Dichotomy Set-Partition and Equal-size Set-Partition, like their predecessor Set-Partition, can be shown to be weakly NP-complete. Recently, Wojtczak proved that even with rational inputs, Set-Partition is strongly NP-complete [6]. In fact, the proof by Wojtczak implied that Dichotomy Set-Partition and Equal-size Set-Partition are both strongly NP-complete — because in this reduction from a special 3-SAT each pair x_i and $\bar{x}_i$ are associated with two unique rational numbers which must be split in two parts. So we re-state this theorem by Wojtczak.

Theorem 1. *Dichotomy Set-Partition with Rationals and Equal-size Set-Partition with Rationals are both strongly NP-complete.*

2.2 Strong NP-hardness for Two-MST

In this subsection, we prove that the Two-MST problem (2-MST for short), is also strongly NP-hard. Recall that in the 2-MST problem, one is given a set P of $2n$ points in the plane, together with two point sites c_1 and c_2, the objective is to compute two MST T_1 and T_2 each containing n points in P (and c_1 and c_2 respectively) such that the maximum weight of T_1 and T_2, $\max\{w(T_1), w(T_2\}$, is minimized. (Here the weight of any edge (p_i, p_j) or (p_i, c_k) in $T_k, k = 1..2$, is the Euclidean distance between the two corresponding nodes.) We reduce Dichotomy Set-Partition for Rationals to 2-MST in the following.

Given $E = \cup_{i=1..n} E'_i$, with E'_i containing two rationals $a_{i,1}$ and $a_{i,2}$, i.e., $E'_i = \{a_{i,1}, a_{i,2}\}$, for Dichotomy Set-Partition with Rationals we need to partition each of E'_i into two sets E_1 and E_2 such that the rationals in E_1 and E_2 sum the same, i.e., $t = \sum_{a \in E_1} a = \sum_{b \in E_2} b$. We construct $4n$ points in P as well as 2 points c_1 and c_2 as follows.

First set $c_1 = c_2 = (0, 0)$. Then for $i = 1$ to n, construct 4 points corresponding to $E'_i = \{a_{i,1}, a_{i,2}\}$: $p_{i,1} = (i \cdot t, a_{i,1})$, $p_{i,2} = (i \cdot t, -a_{i,2})$, $q_{i,1} = (i \cdot t, 0)$ and $q_{i,2} = (i \cdot t, 0)$. We loosely call these 4 points forming the i-th cusp C_i. (The sketch of an example is shown in Fig. 2.) Since $t >> a_{i,1}$ or $a_{i,2}$, the optimal MST's must first split the points on the x-axis, i.e., $\{c_1, c_2\} \cup (\cup_{i=1..n}\{q_{i,1}, q_{i,2}\})$, evenly. Secondly, between the neighboring cusps C_i and C_{i+1}, the closest distance is t; therefore, $p_{i,1}$ and $p_{i,2}$ must be connected to exactly one of $q_{i,1}$ and $q_{i,2}$. To make the maximum weight of the resulting MST T_1 and T_2 minimum, it comes to how to connect $p_{i,1}$ and $p_{i,2}$ to $q_{i,1}$ and $q_{i,2}$ such that the weight of T_1 and T_2 is the same, i.e., with a value of $(n+1)t$. It is clear that in this case $|P| = 4n$ and $|T_1| = |T_2| = 2n+1$, due to the addition of c_1 and c_2. Hence, we summarize: Dichotomy Set-Partition with Rationals has a solution iff the 2-MST instance $P \cup \{c_1, c_2\}$ admits a solution with optimal weight of $(n + 1)t$. We therefore have the following theorem.

Theorem 2. *Two-MST is strongly NP-hard.*

We comment that with this proof, a variation of 2-MST, e.g., even if c_1 and c_2 are not given in advance, remains strongly NP-hard.

2.3 A 4.8536-Approximation for Two-MST

First let P_1 be the subset of points closer to c_1, and P_2 the subset of points closer to c_2 (ties are broken arbitrarily). Let T be an MST of $P \cup \{c_1, c_2\}$. T_1 is obtained by removing

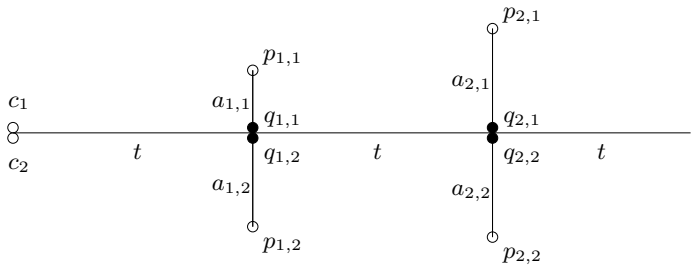


Fig. 2: Illustration for the reduction from Dichotomy Set-Partition with Rationals to 2-MST. Here only the construction for $E'_1 = \{a_{1,1}, a_{1,2}\}$ and $E'_2 = \{a_{2,1}, a_{2,2}\}$ are shown.

c_2 plus any n points of T. Viewing these removed points as Steiner points, by the bound of Chung and Graham [1], we have $w(T_1) \leq (1/0.82416874) \cdot w(T) \leq 1.2134 \cdot w(T)$. Then, obtain T_2 by taking the points not in T_1. We also have $w(T_2) \leq 1.2134 \cdot w(T)$. Return the best solution among (P_1, P_2) or (T_1, T_2). Note that in either case, the approximate solution APP satisfies $APP \leq 1.2134 \cdot w(T)$.

To obtain the final factor, let M_1 and M_2 be the two MST's of the optimal solution, and let OPT be the maximum weight of M_1 or M_2. By taking the union of M_1 and M_2, and adding an edge between c_1 and c_2, we obtain a spanning tree. Thus, $w(M_1) + w(M_2) + d(c_1, c_2) \geq w(T)$, since T is a minimum spanning tree of $P \cup \{c_1, c_2\}$.

Next we show that $OPT \geq d(c_1, c_2)/2$. If the optimal solution splits P into P_1 and P_2, we just return that. Now assume that the optimal solution does not do that. This means that M_1 has a point of P_2, or M_2 has a point of P_1. Let $p \in M_1 \cap P_2$, then the path from c_1 to p in M_1 shows that $OPT \geq d(c_1, c_2)/2$. The same inequality holds if $p \in M_2 \cap P_1$.

Thus we obtain

$$w(M_1) + w(M_2) + d(c_1, c_2) \leq OPT + OPT + 2 \cdot OPT = 4 \cdot OPT.$$

Combined with the above, this gives $APP \leq 1.2134 \cdot w(T) \leq 1.2134 \cdot (4 \cdot OPT) = 4.8536 \cdot OPT$.

Theorem 3. *Two-MST can be approximated with a factor-4.8536 approximation algorithm which runs in $O(n \log n)$ time.*

3 Concluding Remarks

The obvious question is whether we could improve the approximation factor for 2-MST. Even with the current algorithm, we believe that the actual factor should be around 3.

References

1. F.R.K. Chung and R.L. Graham. A new lower bound for Euclidean Steiner minimal trees. *Annals. NY Academy of Sciences*, 440(1):328-346, 1985.
2. G. Even, N. Garg, J. Koenemann, R. Ravi and A. Sinha. Covering graphs using trees and stars. *Proc. APPROX/RANDOM'03*, pp. 24-35, 2003.
3. B. Gamlath and V. Grinberg. Approximating star cover problems. *Proc. APPROX/RANDOM'20*, pp. 57:1-57:19, 2020.
4. M. Garey and D. Johnson. *Computers and Intractability: A Guide to the Theory of NP-completeness*, W.H.Freeman, 1979.
5. R.M. Karp. Reducibility among combinatorial problems. In *Complexity of Computer Computations*, pp. 85-103, Springer US, 1972.
6. D. Wojtczak. On strong NP-completeness of rational numbers. *Proc. CSR'18*, pp. 308-320, 2018.
7. W. Zhao and P. Zhang. Approximation to the Minimum Rooted Star Cover Problem. *Proc. TAMC'07*, pp. 670-679, 2007.

Reverse Shortest Path Problem in Weighted Unit-Disk Graphs*

Haitao Wang† Yiming Zhao‡

Abstract

Given a set P of n points in the plane, a unit-disk graph $G_r(P)$ with respect to a parameter r is an undirected graph whose vertex set is P such that an edge connects two points $p, q \in P$ if the (Euclidean) distance between p and q is at most r (the weight of the edge is 1 in the unweighted case and is the distance between p and q in the weighted case). Given a value $\lambda > 0$ and two points s and t of P, we consider the following *reverse shortest path problem*: Compute the smallest r such that the shortest path length between s and t in $G_r(P)$ is at most λ. The unweighted case of the problem was solved in $O(n^{5/4}\log^{7/4} n)$ time before. In this abstract, we study the weighted case and present an $O(n^{5/4}\log^{5/2} n)$ time algorithm. We also consider the L_1 version of the problem where the distance of two points in the plane is measured by the L_1 metric. We solve the L_1 problem in $O(n\log^3 n)$ time for both the unweighted and weighted cases.

1 Introduction

Given a set P of n points in the plane and a parameter r, the *unit-disk graph* $G_r(P)$ is an undirected graph whose vertex set is P such that an edge connects two points $p, q \in P$ if the (Euclidean) distance between p and q is at most r. The weight of each edge of $G_r(P)$ is defined to be one in the *unweighted* case and is defined to the distance between the two vertices of the edge in the *weighted* case. Alternatively, $G_r(P)$ can be viewed as the intersection graph of the set of congruous disks centered at the points of P with radii equal to $r/2$, i.e., two vertices are connected if their disks intersect. The *length* of a path in $G_r(P)$ is the sum of the weights of the edges of the path.

Computing shortest paths in unit-disk graphs with different distance metrics and different weights assigning methods has been extensively studied, e.g., [3–7, 9–11]. Although a unit-disk graph may have $\Omega(n^2)$ edges, geometric properties allow to solve the single-source-shortest-path problem (SSSP) in sub-quadratic time. Roditty and Segal [9] first proposed an algorithm of $O(n^{4/3+\epsilon})$ time for unit-disk graphs for both unweighted and weighted cases, for any $\epsilon > 0$. Cabello and Jejčič [3] gave an algorithm of $O(n\log n)$ time for the unweighted case. Using a dynamic data structure for bichromatic closest pairs [1], they also solved the weighted case in $O(n^{1+\epsilon})$ time [3]. Chan and Skrepetos [4] gave an $O(n)$ time algorithm for the unweighted case, assuming that all points of P are presorted. Kaplan et al. [7] developed a new randomized result for the dynamic bichromatic closest pair problem; applying the new result to the algorithm of [3] leads to an $O(n\log^{12+o(1)} n)$ expected time randomized algorithm for the weighted case. Recently, Wang and Xue [10] proposed a new algorithm that solves the weighted case in $O(n\log^2 n)$ time.

The L_1 version of the SSSP problem has also been studied, where the distance of two points in the plane is measured under the L_1 metric when defining the graph $G_r(P)$. Note that in the L_1 version a "disk" becomes a diamond. The SSSP algorithms of [3, 4] for the L_2 unweighted version can be easily

*This research was supported in part by NSF under Grant CCF-2005323.

†Department of Computer Science, Utah State University, Logan, UT 84322, USA. haitao.wang@usu.edu

‡Corresponding author. Department of Computer Science, Utah State University, Logan, UT 84322, USA. yiming.zhao@usu.edu

adapted to the L_1 unweighted version. Wang and Zhao [11] recently solved the L_1 weighted case in $O(n \log n)$ time. It is also known $\Omega(n \log n)$ is a lower bound for the SSSP problem in both L_1 and L_2 versions [3, 11]. Hence, the SSSP problem in the L_1 weighted/unweighted case as well as in the L_2 unweighted case has been solved optimally.

In this abstract, we consider the following *reverse shortest path* (RSP) problem. In addition to P, given a value $\lambda > 0$ and two points $s, t \in P$, the problem is to find the smallest value r such that the distance between s and t in $G_r(P)$ is at most λ. Throughout the abstract, we let r^* denote the optimal value r for the problem. The goal is therefore to compute r^*.

Observe that r^* must be equal to the distance of two points in P in any case (i.e., L_1, L_2, weighted, unweighted). For the L_2 unweighted case, Cabello and Jejčič [3] mentioned a straightforward solution that can solve it in $O(n^{4/3} \log^3 n)$ time, by using the distance selection algorithm of Katz and Sharir [8] to perform binary search on all interpoint distances of P; Wang and Zhao [12] later gave two algorithms with time complexities $O(\lfloor \lambda \rfloor \cdot n \log n)$ and $O(n^{5/4} \log^{7/4} n)$[1], respectively, using the parametric search technique. In particular, the first algorithm is interesting when λ is relatively small and the second algorithm uses the first one as a subroutine.

In this abstract, we study the L_2 weighted case of the RSP problem and present an algorithm of $O(n^{5/4} \log^{5/2} n)$ time. In addition, we also consider the L_1 version and solve the L_1 RSP problem in $O(n \log^3 n)$ time for both the unweighted and weighted cases.

The RSP problem has been studied in the literature under various problem settings. Intuitively, the problem is to modify the graph (e.g., modify edge weights) so that certain desired constraints related to shortest paths can be satisfied, e.g., [2, 13]. As a motivation of our problem, consider the following scenario. Suppose $G_r(P)$ represents a wireless sensor network in which each sensor is represented by a disk centered at a point in P and two sensors can communicate with each other (e.g., directly transmit a message) if they are connected by an edge in $G_r(P)$. The disk radius is proportional to the energy of the sensor. The latency of transmitting a message between two neighboring sensors is proportional to their distance. For two sensors s and t, we want to know the minimum energy for all sensors so that the total latency of transmitting messages between s and t is no more than a target value λ. It is not difficult to see that this is equivalent to our RSP problem.

2 Our algorithms – an overview

In this section, we give a brief overview on our algorithms. We begin with the L_2 weighted RSP problem.

Our algorithm for the L_2 weighted RSP problem follows the parametric search scheme. Let $d_r(s, t)$ denote the distance from s to t in $G_r(P)$. Given any r, the *decision problem* is to decide whether $r^* \leq r$. Observe that $r^* \leq r$ holds if and only if $d_r(s, t) \leq \lambda$. Hence, the shortest path algorithm of Wang and Xue [10] (referred to the WX algorithm) can be used to solve the decision problem in $O(n \log^2 n)$ time. To compute r^*, since r^* is equal to the distance of two points of P, one could first compute all interpoint distances of points of P and then use the WX algorithm to perform binary search among these distances to compute r^*. Clearly, the algorithm takes $\Omega(n^2)$ time. Alternatively, as mentioned in [3], one can perform binary search by using the distance selection algorithm of Katz and Sharir [8] (i.e., given any k with $1 \leq k \leq \binom{n}{2}$, the algorithm finds the k-th smallest distance among all interpoint distances of P) without explicitly computing all these $\Omega(n^2)$ distances. As the algorithm of Katz and Sharir [8] runs in $O(n^{4/3} \log^2 n)$, this approach can compute r^* in $O(n^{4/3} \log^3 n)$ time.

We propose a more efficient parametric search algorithm, by "parameterizing" the decision algorithm, i.e., the WX algorithm. Like typical parametric search algorithms, we run the decision algorithm with a parameter $r \in (r_1, r_2]$ by simulating the decision algorithm on the unknown r^*. At

[1]The time complexity given in [12] is $O(n^{5/4} \log^2 n)$, but can be easily improved to $O(n^{5/4} \log^{7/4} n)$ by changing the threshold for defining large cells from $n^{3/4}$ to $(n/\log n)^{3/4}$ in Section 4 [12].

each step of the algorithm, we call the decision algorithm on certain "critical values" r to compare r and r^*, and the algorithm will proceed accordingly based on the result of the comparison. The interval $(r_1, r_2]$ will also be shrunk after these comparisons but is guaranteed to contain r^* throughout the algorithm. The algorithm terminates once t is reached, at which moment we can prove that r^* is equal to r_2 of the current interval $(r_1, r_2]$.

Specifically, the WX algorithm first builds a grid $\Psi_r(P)$ implicitly on the plane such that for any point $p \in P$, if p is in a cell C of the grid, then all neighboring points of p in $G_r(P)$ lie in a constant number of neighboring cells of C. The WX algorithm follows the basic idea of Dijkstra's algorithm and computes an array $dist[\cdot]$ for each point $p \in P$, where $dist[p]$ will be equal to $d_r(s, p)$ when the algorithm terminates. Different from Dijkstra's shortest path algorithm, which picks a single vertex in each iteration to update the shortest path information of other adjacent vertices, the WX algorithm aims to update in each iteration the shortest path information for all points within one single cell of $\Psi_r(P)$ and pass on the shortest path information to vertices lying in the neighboring cells. More specifically, each iteration of the algorithm picks a vertex z with the minimum $dist$-value. We assume that z lies in cell C of $\Psi_r(P)$. We update the shortest information ($dist$-values) of vertices lying in cell C and vertices lying in a constant number of neighboring cells of C. After that, it can be proved that $dist[p] = d_r(s, p)$ for all points p of P in C [10].

To parameterize the WX algorithm, we maintain an interval $(r_1, r_2]$ and ensure $r^* \in (r_1, r_2]$ during the whole algorithm. Our algorithm follows the workflow of the WX algorithm. In each step of our algorithm, although we do not know r^*, we find some critical values of r in $(r_1, r_2]$ such that behaviors of the WX algorithm change on these critical values. Then the decision algorithm is called to do a binary search on these critical values and shrink the interval $(r_1, r_2]$ to $(r_1', r_2']$ such that no critical values lie in (r_1', r_2'). For any $r \in (r_1', r_2')$ after shrinking, if $r^* \neq r_2'$ (i.e., $r^* \in (r_1', r_2')$), then the behaviors of the WX algorithm running on r are the same as the behaviors of the WX algorithm running on r^*. The algorithm terminates after t is reached, and r^* is equal to r_2 of the final interval $(r_1, r_2]$. As t is reached within $O(n)$ steps and each step takes more than linear time (due to calling the decision algorithm), the total time of the algorithm is at least quadratic. To reduce the time complexity, we borrow an idea from [12] as follows. We classify each cell of the grid built in the WX algorithm as a *large cell* if it contains at least $n^{3/4} \log^{3/2} n$ points of P, and a *small cell* otherwise. We use a subroutine of the distance selection algorithm of Katz and Sharir [8] to preprocess all small cells and compute an interval $(r_1, r_2]$ such that if $r^* \neq r_2$, for any $r \in (r_1, r_2)$, the points lying in neighboring small cells have the same connectivities in both $G_r(P)$ and $G_{r^*}(P)$. More specifically, for any two points $p, q \in P$ lying in a pair of neighboring small cells of the grid, an edge connects p and q in $G_r(P)$ if and only if an edge connects p and q in $G_{r^*}(P)$. Then we follow the same algorithm as above. For small cells, we just pick any $r \in (r_1, r_2)$ and run the original WX algorithm on $G_r(P)$ since the points lying in neighboring small cells have the same connectivities in $G_r(P)$ and $G_{r^*}(P)$. This avoids parametric search (and thus calling the decision algorithm is not needed). For large cells, we run the same parametric search as before. The threshold $n^{3/4} \log^{3/2} n$ is carefully chosen to balance the running time of the parametric search on large cells (e.g., there are only $O(n^{1/4}/\log^{3/2} n)$ large cells in the grid) and the preprocessing on small cells. The total time complexity of our algorithm for the L_2 weighted RSP problem is $O(n^{5/4} \log^{5/2} n)$.

For the L_1 RSP problem, we use an approach similar to the distance selection algorithm of Katz and Sharir [8]. As in the L_2 case, the decision problem can be solved in $O(n \log n)$ time by applying the SSSP algorithms for both the unweighted case and the weighted case [3, 4, 12] (more precisely, for the unweighted case, the decision problem can be solved in $O(n)$ time after $O(n \log n)$ time preprocessing for sorting the points of P [4]). Let Π denote the set of all pairwise distances of all points of P. In light of the observation that r^* is in Π, each iteration of our algorithm computes an interval $(a_j, b_j]$ (initially, $a_0 = -\infty$ and $b_0 = \infty$) such that $r^* \in (a_j, b_j]$ and the number of values of Π in $(a_j, b_j]$ is a constant fraction of the number of values of Π in $(a_{j-1}, b_{j-1}]$. In this way, r^* can be found within

$O(\log n)$ iterations. Each iteration will call the decision algorithm to perform binary search on certain values. The total time of the algorithm is $O(n \log^3 n)$.

A by-product of our technique is an $O(n \log^3 n)$ time algorithm that can compute the k-th smallest L_1 distance among all pairs of points of P, for any given k with $1 \leq k \leq \binom{n}{2}$. As mentioned before, the L_2 version of the problem can be solved in $O(n^{4/3} \log^2 n)$ time [8].

Remarks. Our RSP problem is defined with respect to a pair of points (s, t). Our techniques can be extended to solve a more general "single-source" version of the problem: Given a source point $s \in P$ and a value $\lambda > 0$, compute the smallest value r^* such that the shortest paths lengths from s to all vertices of $G_r(P)$ are at most λ, i.e., $\max_{t \in P} d_{r^*}(s, t) \leq \lambda$. The decision problem (i.e., deciding whether $r^* \leq r$ for a given r) now becomes deciding whether $\max_{t \in P} d_r(s, t) \leq \lambda$. All decision algorithms for our original RSP problem are actually for single-source-shortest-paths and thus can be used directly for solving the new decision problem with asymptotically the same time complexities. With these "new" decision algorithms, to compute r^*, we can follow the same algorithm schemes as before. One difference is that in the L_2 case our original algorithm terminates once t is reached but now we instead halt the algorithm once all points of P are reached, which does not affect the running time asymptotically. As such, the "single-source" version of the RSP problem in the L_2 weighted case can be solved in $O(n^{5/4} \log^{5/2} n)$ time and the L_1 unweighted/weighted case can be solved in $O(n \log^3 n)$ time.

References

[1] P.K. Agarwal, A. Efrat, and M. Sharir. Vertical decomposition of shallow levels in 3-dimensional arrangements and its applications. *SIAM Journal on Computing*, 29:912–953, 1999.

[2] D. Burton and P.L. Toint. On an instance of the inverse shortest paths problem. *Mathematical Programming*, 53:45–61, 1992.

[3] S. Cabello and M. Jejčič. Shortest paths in intersection graphs of unit disks. *Computational Geometry: Theory and Applications*, 48(4):360–367, 2015.

[4] T.M. Chan and D. Skrepetos. All-pairs shortest paths in unit-disk graphs in slightly subquadratic time. In *Proceedings of the 27th International Symposium on Algorithms and Computation (ISAAC)*, pages 24:1–24:13, 2016.

[5] T.M. Chan and D. Skrepetos. Approximate shortest paths and distance oracles in weighted unit-disk graphs. In *Proceedings of the 34th International Symposium on Computational Geometry (SoCG)*, pages 24:1–24:13, 2018.

[6] J. Gao and L. Zhang. Well-separated pair decomposition for the unit-disk graph metric and its applications. *SIAM Journal on Computing*, 35(1):151–169, 2005.

[7] H. Kaplan, W. Mulzer, L. Roditty, P. Seiferth, and M. Sharir. Dynamic planar Voronoi diagrams for general distance functions and their algorithmic applications. In *Proceedings of the 28th Annual ACM-SIAM Symposium on Discrete Algorithms (SODA)*, pages 2495–2504, 2017.

[8] M. Katz and M. Sharir. An expander-based approach to geometric optimization. *SIAM Journal on Computing*, 26(5):1384–1408, 1997.

[9] L. Roditty and M. Segal. On bounded leg shortest paths problems. *Algorithmica*, 59(4):583–600, 2011.

[10] H. Wang and J. Xue. Near-optimal algorithms for shortest paths in weighted unit-disk graphs. *Discrete and Computational Geometry*, 64:1141–1166, 2020.

[11] H. Wang and Y. Zhao. An optimal algorithm for L_1 shortest paths in unit-disk graphs. In *Proceedings of the 33rd Canadian Conference on Computational Geometry (CCCG)*, pages 211–218, 2021.

[12] H. Wang and Y. Zhao. Reverse shortest path problem for unit-disk graphs. In *Proceedings of the 17th International Symposium of Algorithms and Data Structures (WADS)*, pages 655–668, 2021. Full version available at https://arxiv.org/abs/2104.14476.

[13] J. Zhang and Y. Lin. Computation of the reverse shortest-path problem. *Journal of Global Optimization*, 25(3):243–261, 2003.

Maximum Matchings on co-Unit Disk Graphs

Shuhao Tan
Department of Computer Science, University of Maryland, College Park, USA

David M. Mount
Department of Computer Science, University of Maryland, College Park, USA

1 Introduction

The unit disk graph is a classical object of interest in computational geometry. It has many applications, notably in the field of communication [3], where each edge models ability to transmit a signal locally. Computing matchings on unit disk graphs is a topic of fundamental interest [2, 5]. We consider the problem of computing a maximum cardinality matching on the *complement* of a unit disk graph, or *co-unit disk graph*. A motivating application is to pair-up people "diversely", in the sense that we want to avoid matchings between individuals that are too similar. Other motivating applications include distributing backup data to distant servers and allocating resources to non-interfering agents.

We consider the problem in a multi-dimensional setting. Let $d(\cdot, \cdot)$ denote the Euclidean distance function. Given a point set $P = \{p_1, p_2, \ldots, p_n\} \subset \mathbb{R}^d$, where d is a constant, the objective is to compute a maximum cardinality matching M on P such that for any matched pair (p_i, p_j), $d(p_i, p_j) > 1$. Here, we consider an approximate variant: Given a constant $0 < \varepsilon < 1$, compute a matching M of cardinality no less than the optimal (in the exact case), such that for any matched pair (p_i, p_j), $d(p_i, p_j) > 1 - \varepsilon$.

Prior Work

Maximum matching on general graphs has been extensively studied. The first breakthrough of the problem was Edmond's blossom algorithm [4], which computes successive augmenting paths in the graph as if it is bipartite and contracts any odd-length cycles it encounters.

The current best bound for general graphs is due to Micali and Vazirani [7, 8]. The algorithm is very similar to the bipartite matching algorithm of Hopcroft and Karp [6], where a maximal partial augmenting matching is found in each iteration. The algorithm runs in time $O(|E|\sqrt{|V|})$, where V and E are the vertex and edge sets, respectively.

The line of study on matchings has generally focused on blossom-based approaches, but Norbert Blum proposed an alternative algorithm by duplicating the graph to form a directed bipartite graph [1]. Finding an augmenting path reduces to a simple reachability problem, but with some additional constraints. A modified depth-first search (DFS) is employed to solve the problem. To achieve the best running time, this is combined with a Hopcroft-Karp style approach.

Matching for geometric objects is also well-studied, but the focus is more on minimizing the weight of matching [9–11] or on geometric intersection graphs [2]. The longest perfect matching problem [5], in which the goal is to maximize the minimum distance between matched pairs, is closely related to the problem we consider.

Our Contribution

Our technique involves computing the well-separated pair decomposition and obtaining a compressed representation of the ε-approximation of the co-unit disk graph. We then modify a reachability-based matching algorithm by Norbert Blum [1] to run on the compressed representation.

The standard quadtree-based WSPD construction yields a quadtree with a list of pairs of nodes representing the well-separated pairs. We consider the general case where a tree and a list of pairs of nodes are given. The corresponding graph is obtained by expanding a pair to a bipartite clique between the leaves of the two subtrees rooted at the two nodes in the pair. We show that a matching corresponds to a flow-network-like structure on the tree, and finding an augmenting path corresponds to finding an augmenting flow.

We then modify Blum's matching algorithm on this structure. In Blum's algorithm, the original graph is duplicated to represent inner/outer vertices, and directed edges between them are added to represent matched/unmatched edges. Blum showed that an alternating path in the original corresponds to a simple path in the modified graph that visits at most one copy of each duplicated vertex. Augmentation reduces to finding such a path. Our duplication is more involved since our modified graph is not bipartite, the tree structure is preserved in both copies and edges corresponding to a pair of nodes are added between copies. We also realize that in our model, the same vertex from both copies can be both visited in a valid augmenting flow. We generalize Blum's definition of strong simplicity to restrict double visit to a given set of edges. We show that the invariants for Blum's algorithm still hold, and adapting the algorithm solves the matching problem on our compressed representation.

▶ **Theorem 1.** *An ε-approximation of maximum cardinality matching on a co-unit disk graph of n points in $\mathbb{R}^d$ can be computed in $O(n^2/\varepsilon^d)$ time.*

Remark Solving the exact problem using Micali-Vazirani algorithm yields a running time of $O(n^{2.5})$. We believe further adapting the algorithm to Hopcraft-Karp style could result in a $O(n^{1.5}/\varepsilon^d)$ algorithm.

2 Hierarchical Bipartite Clique Decomposition

A *hierarchical bipartite clique decomposition* (HBCD hereafter) is defined as (T, B) where

1. T is a tree.
2. B is a set of edges between tree nodes such that neither node is an ancestor of the other. We will call them *bridges*.
3. If $(u, v) \in B$, then for any $x \in S(u), y \in S(v)$, where $S(u)$ is the subtree rooted at u, $(x, y) \neq (u, v) \implies (x, y) \notin B$.

Given an HBCD (T, B), the corresponding general graph $G(V, E)$ is constructed as follows:

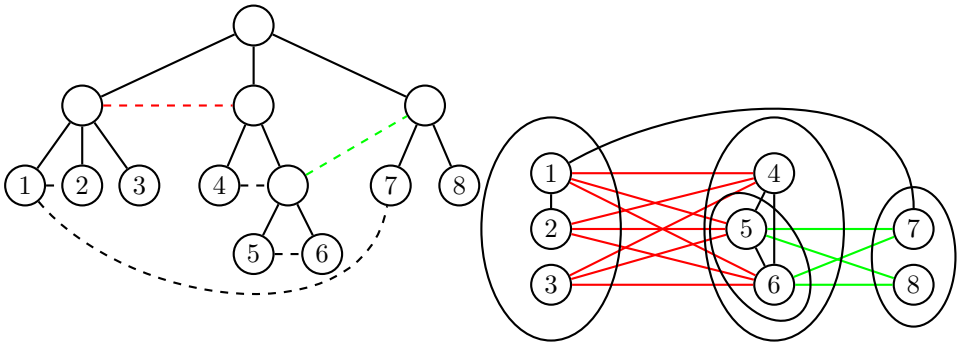
1. V is the set of leaves of T.
2. $(u, v) \in E \iff \exists x \in P(u), \exists y \in P(v), (x, y) \in B$. e.g., there exists a bridge connecting u and v via their ancestor.

A *matching* in an HBCD is a pairing relation between leaves such that the relation is a matching in the corresponding general graph. The *cardinality* of a matching is the number of pairs.

A *flow network* on an instance of HBCD (T, B) is a function $f : E(T) \cup B \to \mathbb{N}$ where $E(T)$ is the edge set of T such that:

1. Flow conservation: For each leaf node u, $\sum_{e \in E(u)} f(e) \leq 1$ where $E(u)$ denotes the set of edges incident to u.
2. Uncapacitated: For every other node u, let A be the set of edges that connect u and its children, and let B be the remaining edges incident to u, $\sum_{e \in A} f(e) = \sum_{e \in B} f(e)$.

Intuitively, we have an undirected flow network where Property 2 mandates that flows are generated at leaf nodes and each leaf node can only contribute to no flow or a single unit



(a) HBCD, dashed edges are bridges **(b)** Equivalent general graph

Figure 1 Correspondence between HBCD and general graph

of flow in the network. Property 1 is the usual flow conservation property where we treat each node as a tunnel between edges from children and other edges.

We call a leaf node *exposed* if there is no flow attached to it.

▶ **Lemma 2.** *Let (T, B) be an instance of HBCD. There is a matching of cardinality n if and only if there is a flow network with $2n$ non-exposed leaf nodes.*

Like an augmenting path for a matching on general graph, we have similar definition for the flow network. The augmenting path in a flow network is a path equipped with ± 1 values, starting and ending at two distinct exposed leaf nodes, and preserve flow conservation locally and validity of flow network globally when added to the network.

Remark In contrast to an augmenting path for general graph, an augmenting path for flow network is not necessarily simple. Moreover, same edge could appear multiple times in a path. However, we will show that it is almost simple after some simplification.

It is also easy to see that adding the values of an augmenting path to a flow network f should produce a flow network that is still valid, but with two fewer exposed leaf nodes.

▶ **Definition 3.** *A simplified augmenting path for a flow network f is a path $\mathcal{P}$ such that:*

1. *$\mathcal{P}$ is an augmenting path.*
2. *Consider the edges on $\mathcal{P}$ to be directed, any edge (u, v) appears at most once in $\mathcal{P}$.*
3. *Consider the edges on $\mathcal{P}$ to be directed, if (u, v) and (v, u) both appear in $\mathcal{P}$, their associated values are the same.*

▶ **Lemma 4.** *If there exists an augmenting path $\mathcal{P}$ in a flow network f, then there also exists a simplified augmenting path $\widehat{\mathcal{P}}$*

▶ **Lemma 5.** *A flow network f is maximum if and only if there is no augmenting path for f.*

3 Algorithm to Find an Augmenting Path

We will modify the maximum cardinality matching algorithm by Blum [1]. The distinct feature of Blum's algorithm is that it does not explicitly deal with blossoms from the traditional Edmond's blossom algorithm. Instead, he reduced the problem to a reachability problem with constraints on paths constructed. We will follow the same idea here.

The main difference between our algorithm from Blum's algorithm is that a valid augmenting path in our model could contain the same edge twice. We need to relax the notion of

"strongly simple" path from Blum's algorithm (meaning that the final path does not contain duplicate vertices). We tackle this by modifying the algorithm with additional notion of one-way/two-way edges.

The general idea is to naturally add tree structure on the two copies that correspond to the possible flow on the edge. We identify pairs of edges that correspond to -1 on an edge with exactly 1 flow to be "one-way", so that such edges cannot be both traversed in the two copies. Blum's algorithm is modified in a way that instead of avoiding visiting the same vertex in both copies, it avoids visiting the same "one-way" edges in both copies.

Due to space limitations, we omit the presentation of the algorithm. These details will be presented in the full version. We obtain the following result.

▶ **Theorem 6.** *Maximum cardinality matching on HBCD can be solved in* $O(|T|(|T| + |B|)$.

We can now prove Theorem 1. Given a point set P, we first compute a quadtree based $(4/\varepsilon)$-WSPD of P, then prune the pairs with closest distance smaller than $1 - \varepsilon$, and perform the maximum cardinality matching described in the previous section. Computing the WSPD takes $O(n/\varepsilon^d + n \log n)$ time, producing a quad tree of size $O(n)$ and a list of pairs of size $O(n/\varepsilon^d)$. The matching takes time $O(n^2/\varepsilon^d)$, and so the overall running time is $O(n^2/\varepsilon^d)$.

References

1 N. Blum. A new approach to maximum matching in general graphs. *Proc. 17th Internat. Colloq. on Autom., Lang. and Program.*, pages 586–597, 1990. Note: A newer and more complete version can be found at `https://arxiv.org/abs/1509.04927`. `doi:10.1007/BFb0032060`.

2 É. Bonnet, S. Cabello, and W. Mulzer. Maximum Matchings in Geometric Intersection Graphs. *Proc. 37th Internat. Sympos. on Theoret. Aspects of Comp. Sci.*, 154:31:1–31:17, 2020. URL: `https://drops.dagstuhl.de/opus/volltexte/2020/11892`, `doi:10.4230/LIPIcs.STACS.2020.31`.

3 B. N. Clark, C. J. Colbourn, and D. S. Johnson. Unit disk graphs. *Discrete Math.*, 86(1):165–177, 1990. URL: `https://www.sciencedirect.com/science/article/pii/0012365X90903580`, `doi:https://doi.org/10.1016/0012-365X(90)90358-O`.

4 J. Edmonds. Paths, trees, and flowers. *Canad. J. Math*, 17:449–467, 1965. `doi:10.4153/CJM-1965-045-4`.

5 A. Efrat, A. Itai, and M. J. Katz. Geometry helps in bottleneck matching and related problems. *Algorithmica*, 31(1):1–28, September 2001. `doi:10.1007/s00453-001-0016-8`.

6 J. E. Hopcroft and R. M. Karp. A $n^{5/2}$ algorithm for maximum matchings in bipartite graphs. In *12th Annual Symposium on Switching and Automata Theory (swat 1971)*, pages 122–125, 1971. `doi:10.1109/SWAT.1971.1`.

7 S. Micali and V. V. Vazirani. An $O(\sqrt{|v|} \cdot |E|)$ algoithm for finding maximum matching in general graphs. In *21st Annual Symposium on Foundations of Computer Science (sfcs 1980)*, pages 17–27, 1980. `doi:10.1109/SFCS.1980.12`.

8 P. A. Peterson and M. C. Loui. The general maximum matching algorithm of Micali and Vazirani. *Algorithmica*, 3(1):511–533, Nov 1988. `doi:10.1007/BF01762129`.

9 R. Sharathkumar and P. K. Agarwal. A near-linear time ϵ-approximation algorithm for geometric bipartite matching. *Proc. 44th Annu. ACM Sympos. Theory Comput.*, page 385–394, 2012. `doi:10.1145/2213977.2214014`.

10 P. M. Vaidya. Approximate minimum weight matching on points in k-dimensional space. *Algorithmica*, 4(1):569–583, June 1989. `doi:10.1007/BF01553909`.

11 K. R. Varadarajan and P. K. Agarwal. Approximation algorithms for bipartite and non-bipartite matching in the plane. *Proc. Tenth Annu. ACM-SIAM Sympos. Discrete Algorithms*, page 805–814, 1999. `doi:10.5555/314500.314918`.

Catching Polygons

Bradley McCoy * Eli Quist † Anna Schenfisch†

Abstract

Consider an arrangement of k lines intersecting the unit square. There is some minimum scaling factor so that any placement of a rectangle with aspect ratio $1 \times p$ with $p \geq 1$ must non-transversely intersect some portion of the arrangment or unit square. Assuming the lines of the arrangement are axis-aligned, we show the optimal arrangement depends on the aspect ratio of the rectangle. In particular, the optimal arranment is either evenly spaced parallel lines or an evenly spaced grid of lines. We present the precise aspect ratios of rectangles for which each of the two nets are optimal.

1 Introduction

During the open problem session of CCCG20, Joseph O'Rourke suggested the following problem. Consider an arrangement of k lines, all of which intersect the unit square. For a fixed polygon P, there is some minimum scale factor $c > 0$ such that the scaled polygon cP cannot be embedded in the unit square without intersecting any of the k lines of the arrangement non-transversely (allowing translation and rotation). That is, the scaled polygon will 'just touch' at least one line of the arrangement. Can we compute the minimum such c over all possible arrangements of this type? Can we describe an arrangement that realizes this minimum? See Figure 1 for an example.

This problem can be described using an analogy to catching fish. In this analogy, the polygon is a fish that lives in the unit square. The fish can vary in size but always has the same shape. The goal is to catch fish, that is, create a net so that the largest fish that can get through any hole in the net is as small as possible.

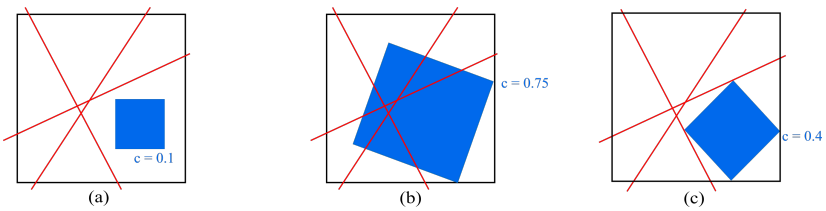


Figure 1: (a) If c is small, the polygon can avoid the net. (b) If c is large, any rotation and translation of the polygon is caught in the net. (c) The minimum c such that the polygon is caught in the net.

Here, we consider the special case where the lines are axis-aligned and P is a rectangle (a lemma towards justifying the setting of axis-aligned lines is given in Appendix B) We observe that, depending on the aspect ratio of the rectangle being considered, the optimal arrangement is either evenly spaced parallel lines or a grid of lines. See Figure 2 for an example. O'Rourke asked, *for what aspect ratios of rectangles is each of the two nets optimal?* In this work, we answer this question precisely. We are only aware of one work that directly considers this problem [1], however the author has a different interpretation of the problem. The problem we consider is closely related to the laser based localization problems considered in [3]. Using the results from [2], given any net and aspect ratio for the fish one can compute the optimal scale factor.

*School of Computing, Montana State U.
†Dept. of Mathematical Sciences, Montana State U.
bradleymccoy@montana.edu, eli.quist@student.montana.edu, annaschenfisch@montana.edu

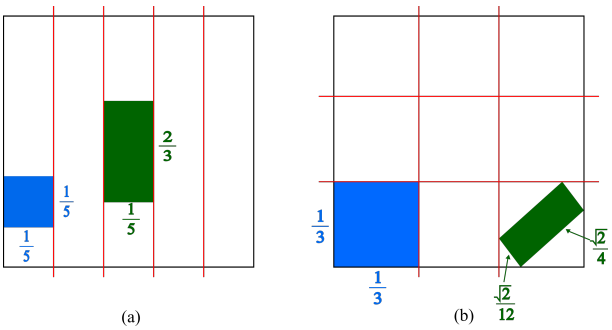


Figure 2: (a) For a square the evenly spaced vertical lines are optimal but for a rectangle with a (1×3) aspect ratio evenly spaced vertical lines is not optimal. (b) For a square the evenly spaced grid is not optimal but for a (1×3) rectangle the evenly spaced grid is optimal.

2 Rectangular Fish in Rectangular Nets

When P is a rectangle and the k lines are axis-aligned, the problem of calculating the optimal scale factor c reduces to finding the largest rectangle inscribed inside of another rectangle. Inscribing rectangles inside rectangles has been studied in [4, 5, 6]. In this section, we construct a curve that describes the optimal scale factor c for rectangular fish in rectangular nets for all spect ratios of the inscribed rectangle.

We fix the aspect ratio of the hole in the net to be $1 \times n$ with $n \geq 1$. Let the aspect ratio of the fish be $1 \times p$ with $p \geq 1$. We express the optimal scale factor c as a function of the aspect ratio p. We denote this curve by $\mathcal{C}_n(p)$. See Figure 3 for an example.

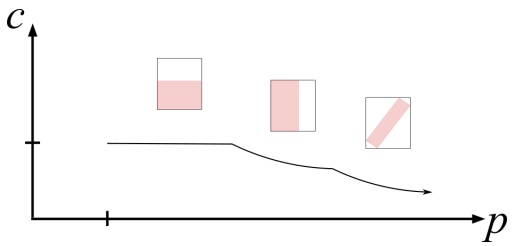


Figure 3: The inscribing curve for some $n \in \mathbb{R}^{\geq 1}$. The x-axis is the aspect ration of the fish and the y-axis is the optimal scale factor.

The curve $\mathcal{C}_n(p)$ consists of three parts. For small $p \leq n$, placing the shorter side of the fish parallel to the shorter side of the net is optimal and $c = 1$. For medium size p values relative to n, the value of c is limited by the height of the net and the longer side of the fish is scaled to equal the longer side of the net. This gives $pc = n$ or $c = \frac{n}{p}$.

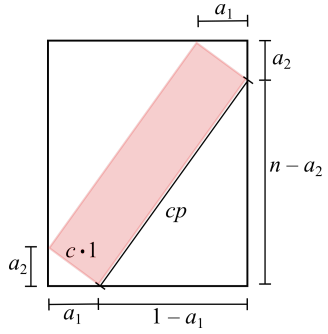


Figure 4: A diagonally inscribed rectangle.

For large p relative to n, placing the fish diagonally is optimal. Using the variables indicated in Figure 4, we have the following three equations

$$\frac{a_1}{a_2} = \frac{n - a_2}{1 - a_1} \tag{1}$$

$$a_1^2 + a_2^2 = c^2 \tag{2}$$

$$(1 - a_1)^2 + (n - a_2)^2 = (cp)^2 \tag{3}$$

along with the natural constraints of the problem, $0 < c$, $0 < a_1 < 1$, $0 < a_2 < n$. Equation 1 is due to all triangles being similar. Equation 2 and Equation 3 are applications of Pythagorean's theorem.

Solving for c gives

$$c = \sqrt{\left(\frac{n-p}{1-p^2}\right)^2 (p^2+1) - 2\left(\frac{n-p}{1-p^2}\right) p + 1}.$$

Therefore, we are able to give an explicit formula for $\mathcal{C}_n(p)$ for a given n, namely,

$$\mathcal{C}_n(p) = \begin{cases} 1 & 1 \le p \le n \\ \frac{n}{p} & n < p \le w_n \\ \sqrt{\left(\frac{n-p}{1-p^2}\right)^2 (p^2+1) - 2\left(\frac{n-p}{1-p^2}\right) p + 1} & w_n < p \end{cases}$$

where w_n is the solution to $\sqrt{\left(\frac{n-p}{1-p^2}\right)^2 (p^2+1) - 2\left(\frac{n-p}{1-p^2}\right) p + 1} = \frac{n}{p}$ for $1 < n < p$. The value w_n represents the scale factor where the vertically inscribed rectangle has the same scale factor as the diagonally inscribed rectangle.

When k is even, the grid with $\frac{k}{2}$ horizontal lines and $\frac{k}{2}$ vertical lines has square holes with side length $\frac{1}{\frac{k}{2}+1}$. The arrangement with k vertical and 0 horizontal lines has rectangular holes with dimension $1 \times \frac{1}{k+1}$. The curves $\frac{1}{\frac{k}{2}+1}\mathcal{C}_1(p)$ and $\frac{1}{k+1}\mathcal{C}_{k+1}(p)$ intersect at $p = \frac{k+1}{\frac{k}{2}+1}$. For even values of k, we define the *base curve* to be

$$B_k(p) = \min\left\{\frac{1}{k+1}\mathcal{C}_{k+1}(p), \frac{1}{\frac{k}{2}+1}\mathcal{C}_1(p)\right\}$$

The case for odd values of k is similar and included in Appendix A.

3 Optimality Results

In this section, we show that for any axis-aligned net $\mathcal{N}$ with an even number of lines and any aspect ratio of the fish p, the base curve has a scale factor that is less than or equal to the scale factor of $\mathcal{N}$. Let $\mathcal{N}(k,0)$ denote the net with k evenly spaced parallel lines and let $\mathcal{N}(\frac{k}{2},\frac{k}{2})$ denote the net with $\frac{k}{2}$ evenly spaced vertical lines and $\frac{k}{2}$ evenly spaced horizontal lines. The rectangle aspect ratio where the base curve switches from $\mathcal{N}(k,0)$ to $\mathcal{N}(\frac{k}{2},\frac{k}{2})$ is $p = \frac{k+1}{\frac{k}{2}+1}$. We now state our main theorem:

Theorem 1 (Regular is Optimal)**.** *For k even, the optimal axis aligned net for rectangular polygons is $\mathcal{N}(k,0)$ for aspect ratio $p \le \frac{k+1}{\frac{k}{2}+1}$ and $\mathcal{N}(\frac{k}{2},\frac{k}{2})$ for aspect ratio $p \ge \frac{k+1}{\frac{k}{2}+1}$.*

Proof. First, notice that regularly spaced lines give a smaller scale factor than irregularly spaced lines. This is because a rectangular hole generated by regular spacing has height and width equal to the average. A rectangular hole generated by irregular spacing has a hole with height and width greater than or equal to the average.

Consider any axis aligned net over the square, let v be the number of vertical lines and h be the number of horizontal lines call the net $\mathcal{N}(v,h)$. Notice $v+h=k$. If $v=n$ then we have a $\frac{k}{2} \times \frac{k}{2}$ net and if $n=0$ (or $v=0$), then we have k parallel lines. The base curve is defined to be the minimum scale factor of $\mathcal{N}(\frac{k}{2},\frac{k}{2})$ and $\mathcal{N}(k,0)$. Thus, the scale factor of the base curve is less than or equal to either of these nets.

Consider any other v and h. Without loss of generality, assume $h < v$ we have $0 < h < \frac{k}{2} < v < k$. There are $(v+1)(h+1)$ holes in the net and the average size of a hole is $\frac{1}{v+1} \times \frac{1}{h+1}$. There exists one hole at least as big as the average, so, we have a hole at least as big as the hole with aspect ratio $\frac{v+1}{h+1}$ scaled so the smaller side has length $\frac{1}{v+1}$. The optimal scale factor of a rectangle with aspect ratio p that fits inside this hole is given by $\frac{1}{v+1}\mathcal{C}_{\frac{v+1}{h+1}}(p)$. We compare $\frac{1}{v+1}\mathcal{C}_{\frac{v+1}{h+1}}(p)$ to $B_k(p)$. Both $\frac{1}{v+1}\mathcal{C}_{\frac{v+1}{h+1}}(p)$ and $B_k(p)$ are piecewise functions, we directly examine all $p \geq 1$ to show $B_k(p) \leq \frac{1}{v+1}\mathcal{C}_{\frac{v+1}{h+1}}(p)$, see Figure 5 for intuition.

For $1 \leq p \leq \frac{k+1}{\frac{k}{2}+1}$, we have $B_k(p) = \frac{1}{k+1}$ and $\frac{1}{v+1}\mathcal{C}_{\frac{v+1}{h+1}}(p) = \frac{1}{v+1}$. since $v < k$, we have $\frac{1}{k+1} < \frac{1}{v+1}$. For $\frac{k+1}{\frac{k}{2}+1} \leq p \leq \frac{v+1}{h+1}$, $B_k(p)$ decreases and $\frac{1}{v+1}\mathcal{C}_{\frac{v+1}{h+1}}(p) = \frac{1}{k+1}$ is constant.

For $\frac{v+1}{h+1} < p \leq w_{\frac{v+1}{h+1}}$, $B_k(p) = \min\left\{\frac{1}{k+1}\mathcal{C}_{k+1}(p), \frac{1}{\frac{k}{2}+1}\mathcal{C}_1(p)\right\} \leq \left(\frac{1}{\frac{k}{2}+1}\right)\left(\frac{1}{p}\right)$ and $\frac{1}{v+1}\mathcal{C}_{\frac{v+1}{h+1}}(p) = \frac{1}{v+1}\left(\frac{v+1}{h+1}\right)\frac{1}{p}$. Since $h \leq \frac{k}{2}$, we have

$$\left(\frac{1}{\frac{k}{2}+1}\right)\left(\frac{1}{p}\right) < \frac{1}{h+1}\left(\frac{1}{p}\right) = \frac{1}{v+1}\left(\frac{v+1}{h+1}\right)\frac{1}{p}.$$

For $p \geq w_{\frac{v+1}{h+1}}$ the interior rectangle is placed diagonally. Let c' be the scale factor value of the rectangle placed diagonally in $\mathcal{N}(\frac{k}{2}, \frac{k}{2})$. Consider the length of the rectangle, with shorter side equal to c', placed diagonally in a rectangle with sides $\frac{1}{v+1} \times \frac{1}{h+1}$. Let a_1 and a_2 be the legs of the small right triangle formed by c', the squared length of this inscribed rectangle is $\ell^2(v, h, a_1, a_2) = (\frac{1}{v+1} - a_2)^2 + (\frac{1}{h+1} - a_1)^2$. The minimum of this function along the constraints Equation 1, Equation 2, and $v + h = k$ occurs when $v = h$. We omit the details, but this can be done with Lagrange multipliers. So, when $v \neq h$ we can fit the diagonal rectangle of $\mathcal{N}(\frac{k}{2}, \frac{k}{2})$ inside the diagonal of the rectangle with dimensions $\frac{1}{v+1} \times \frac{1}{h+1}$ so the optimal scale factor must be larger. □

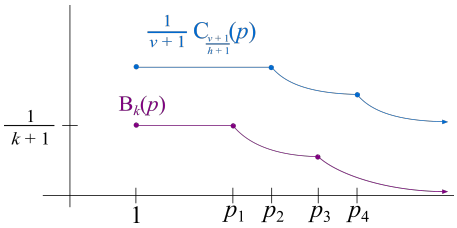


Figure 5: The curves $B_k(p)$ and $\frac{1}{v+1}\mathcal{C}_{\frac{v+1}{h+1}}(p)$. Here, $p_1 = \frac{k+1}{\frac{k}{2}+1}$, $p_2 = \frac{v+1}{h+1}$, $p_3 = \sqrt{2}+1$, and $p_4 = w_{\frac{v+1}{h+1}}$.

4 Discussion

In this work, we showed that for axis-aligned nets with k lines and rectangular fish with aspect ratio $1 \times p$, the optimal net is evenly spaced parallel lines for $p \leq \frac{k+1}{\frac{k}{2}+1}$ and a grid of $\frac{k}{2} \times \frac{k}{2}$ evenly spaced lines for $p \geq \frac{k+1}{\frac{k}{2}+1}$. As far as we know, the problem is still open for non axis-aligned nets and rectangular fish. Our hope is that someone can show, that for any net, one can construct an axis-aligned net that does change the optimal scale factor. Non-rectangular fish would also be interesting to explore.

References

[1] S. Aghamolaei, *Catching a polygonal fish with a minimum net*, arXiv:2008.06337, (2020).

[2] H. Alt, D. Hsu, and J. Snoeyink, *Computing the largest inscribed isothetic rectangle*, 7th Canad. Conf. Comput. Geom, (1995), pp. 67–72.

[3] E. M. Arkin, R. Das, J. Gao, M. Goswami, J. S. B. Mitchell, V. Polishchuk, and C. D. Tóth, *Cutting polygons into small pieces with chords: Laser-based localization*, 28th Euro. Symp. on Algorithms, 173 (2020), pp. 7:1–7:23.

[4] W. Carver, *Solution to problem E1225*, Amer. Math. Monthly, 64 (1957), pp. 114–116.

[5] O. Dunkel, *Solution I to problem 416*, Amer. Math. Monthly, 27 (1920), pp. 327–330.

[6] J. E. Wetzel, *Rectangles in rectangles*, Mathematics Magazine, 73 (2000), pp. 204–211.

A An odd number of lines

In this section, we prove that, when k is odd, the net $\mathcal{N}(k,0)$ is optimal for $1 \le p \le \frac{(k+1)\lfloor \frac{k}{2} \rfloor}{\lceil \frac{k}{2} \rceil^2}$ and the net $\mathcal{N}(\lceil \frac{k}{2} \rceil, \lfloor \frac{k}{2} \rfloor)$ is optimal for $p \ge \frac{(k+1)\lfloor \frac{k}{2} \rfloor}{\lceil \frac{k}{2} \rceil^2}$. The net $\mathcal{N}(\lceil \frac{k}{2} \rceil, \lfloor \frac{k}{2} \rfloor)$ has $\frac{1}{\lceil \frac{k}{2} \rceil} \times \frac{1}{\lfloor \frac{k}{2} \rfloor}$ holes and the net $\mathcal{N}(k,0)$ has $1 \times \frac{1}{k+1}$ holes. We consider the curves $\frac{1}{\lceil \frac{k}{2} \rceil} \mathcal{C}_{\frac{\lceil \frac{k}{2} \rceil}{\lfloor \frac{k}{2} \rfloor}}(p)$ and $\frac{1}{k+1}\mathcal{C}_{k+1}(p)$. See Figure 6. These curves intersect at $p = \frac{(k+1)\lfloor \frac{k}{2} \rfloor}{\lceil \frac{k}{2} \rceil^2}$. We define the *base curve* for k odd to be

$$D_k(p) = \min\left\{ \frac{1}{k+1}\mathcal{C}_{k+1}(p), \frac{1}{\lceil \frac{k}{2} \rceil} \mathcal{C}_{\frac{\lceil \frac{k}{2} \rceil}{\lfloor \frac{k}{2} \rfloor}}(p) \right\}.$$

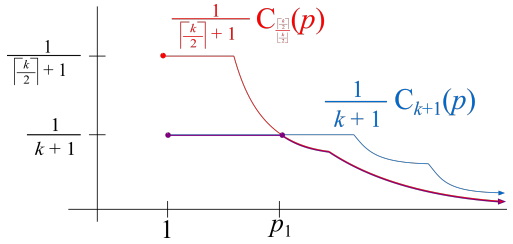


Figure 6: The minimum curve for k odd, with $p_1 = \frac{(k+1)\lfloor \frac{k}{2} \rfloor}{\lceil \frac{k}{2} \rceil^2}$.

Theorem 2 (Regular is Optimal Odd)**.** *For k odd, the optimal axis aligned net for rectangular polygons is $\mathcal{N}(k,0)$ for aspect ratio $p \le \frac{(k+1)\lfloor \frac{k}{2} \rfloor}{\lceil \frac{k}{2} \rceil^2}$ and $\mathcal{N}(\lceil \frac{k}{2} \rceil, \lfloor \frac{k}{2} \rfloor)$ for $p \ge \frac{(k+1)\lfloor \frac{k}{2} \rfloor}{\lceil \frac{k}{2} \rceil^2}$.*

Proof. Consider any axis aligned net over the square, let v be the number of vertical lines and h be the number of horizontal lines call the net $\mathcal{N}(v,h)$. Notice $v + h = k$.

Recall evenly spaced lines give a smaller scale factor than irregularly spaced lines. If $v = \lceil \frac{k}{2} \rceil$ and $\lfloor \frac{k}{2} \rfloor = n$ then we have a $\lceil \frac{k}{2} \rceil \times \lfloor \frac{k}{2} \rfloor$ net. If $n = 0$ (or $v = 0$), then we have k parallel lines. The base curve is defined to be the minimum scale factor of $\mathcal{N}(k,0)$ and $\mathcal{N}(\lceil \frac{k}{2} \rceil, \lfloor \frac{k}{2} \rfloor)$ so the base curve has scale factor less than or equal to $\mathcal{N}(k,0)$.

Consider any other v and h. Without loss of generality assume $h < v$ we have $0 < h < \lfloor \frac{k}{2} \rfloor < \lceil \frac{k}{2} \rceil < v < k$

There are $(v+1)(h+1)$ holes in the net and the average size of a hole is $\frac{1}{v+1} \times \frac{1}{h+1}$. There exits one hole at least as big as the average, that is, with width at least $\frac{1}{v+1}$ and height at least $\frac{1}{h+1}$. So we have a hole with aspect ratio $\frac{v+1}{h+1}$ scaled so the smaller side has length $\frac{1}{v+1}$. The maximum scale factor of a rectangle

with aspect ratio p that fits inside this hole is given by $\frac{1}{v+1}\mathcal{C}_{\frac{v+1}{h+1}}(p)$. We compare $\frac{1}{v+1}\mathcal{C}_{\frac{v+1}{h+1}}(p)$ to $D_k(p)$. Both $\frac{1}{v+1}\mathcal{C}_{\frac{v+1}{h+1}}(p)$ and $D_k(p)$ are piecewise functions, we directly examine all $p \geq 1$ to show $D_k(p) < \frac{1}{v+1}\mathcal{C}_{\frac{v+1}{h+1}}(p)$.

For $1 \leq p \leq \frac{(k+1)\lfloor \frac{k}{2} \rfloor}{\lceil \frac{k}{2} \rceil^2}$, we have $D_k(p) = \frac{1}{k+1}$ and $\frac{1}{v+1}\mathcal{C}_{\frac{v+1}{h+1}}(p) = \frac{1}{v+1}$. since $v < k$, we have $\frac{1}{k+1} < \frac{1}{v+1}$. Then, for $\frac{(k+1)\lfloor \frac{k}{2} \rfloor}{\lceil \frac{k}{2} \rceil^2} \leq p \leq \frac{v+1}{h+1}$, $D_k(p)$ decreases and $\frac{1}{v+1}\mathcal{C}_{\frac{v+1}{h+1}}(p) = \frac{1}{k+1}$ is constant.

For $\frac{v+1}{h+1} < p \leq w_{\frac{v+1}{h+1}}$, $D_k(p) = \min\left\{\frac{1}{k+1}\mathcal{C}_{k+1}(p), \frac{1}{\lceil \frac{k}{2} \rceil}\mathcal{C}_{\frac{\lceil \frac{k}{2} \rceil}{\lfloor \frac{k}{2} \rfloor}}(p)\right\} \leq \frac{1}{\lceil \frac{k}{2} \rceil}\left(\frac{1}{p}\right)$ and

$\frac{1}{v+1}\mathcal{C}_{\frac{v+1}{h+1}}(p) = \frac{1}{v+1}\left(\frac{v+1}{h+1}\right)\frac{1}{p}$. Since $h \leq \lfloor \frac{k}{2} \rfloor$, we have

$$\left(\frac{1}{\lceil \frac{k}{2} \rceil}\right)\left(\frac{1}{p}\right) < \frac{1}{h+1}\frac{1}{p} = \frac{1}{v+1}\left(\frac{v+1}{h+1}\right)\frac{1}{p}.$$

For $p \geq w_{\frac{v+1}{h+1}}$, we again solve the same constrained optimization problem as in the even case. The minimum occurs when $v = h$. This is a global minimum, so in the odd case the minimum occurs when me make v as close to h as possible. So, when $h < \lfloor \frac{k}{2} \rfloor < \lceil \frac{k}{2} \rceil < v$ we can fit the diagonal rectangle of $\mathcal{N}(\lceil \frac{k}{2} \rceil, \lfloor \frac{k}{2} \rfloor)$ inside the diagonal of the rectangle with dimensions $\frac{1}{v+1} \times \frac{1}{h+1}$ and the optimal scale factor must be larger.

□

B Towards Axis-Aligned Net Optimality

In this appendix, we show that, for square fish, evenly spaced axis-aligned vertical lines are generally a local optimum. This is a first step toward our conjecture that axis-aligned lines are a global optimum for rectangular fish.

Lemma 1. *Let P be a fish with aspect ration $p = 1$, i.e., a square. Then evenly spaced axis-aligned vertical lines are a local optimum when the number of lines is $k > 2$.*

Proof. Denote the k lines from left to right by $\ell_1, \ell_2, \ldots, \ell_k$. Assume, towards a contradiction, that, given some small $\epsilon > 0$, there are small shift and pivot values for each line that describe the translation of lines to an arrangement that results in a lower overall scaling factor c. Specifically, let these values be denoted $s_1, s_2, \ldots, s_k$ and $p_1, p_2, \ldots, p_k$, respectively, where $s_i, p_i \in [0, \epsilon]$, and at least one of these values s_i or p_i are nonzero. Notice that, regardless of the pivot value, shifting any line decreases the maximum fish size for one neighboring face, but increases it for the other neighboring face, leading to a higher c-value overall. Thus, we must have $s_1 = s_2 = \ldots = s_k = 0$. This means we must consider the effect of pivot values on unshifted lines. Notice that pivoting a single line that is adjacent to a vertical (un-pivoted) line will increase the maximum fish size for the face between them. Since the lines p_1 and p_k are always adjacent to the vertical edges of the bounding square, we must have $p_1 = p_k = 0$, otherwise the leftmost and rightmost faces would cause the arrangement to have a higher c-value. But then we must also have $p_2 = p_{k-1} = 0$, or else the second to leftmost and second to rightmost faces would cause the arrangement to have a higher c-value. Continuing this line of argument, we eventually see that $p_1 = p_2 = \ldots = p_k = 0$, contradicting our assumption that some s_i or p_i be nonzero. □

Measuring Length-Preserving Fréchet Correspondence for Graphs in $\mathbb{R}^2$

Kevin Buchin, Brittany Terese Fasy*,
Erfan Hosseini Sereshgi†, and Carola Wenk†

Abstract

Finding a comprehensive way to compare two road networks has been a challenge for quite a long time. One frequently-used comparison uses a discrete (easy to compute) sample from each graph to define a similarity measure based on a correspondence between the samples. In practice, this takes into account both the geometry and the topology of the graphs. However, this sampling approach introduces many variables along the way and can result in undesirable correspondences. In this paper, we introduce a continuous alternative to this method that uses Fréchet distance in a length-preserving setting.

1 Introduction

Finding a comprehensive way to compare two roadmaps or embedded graphs has been a challenge for quite a long time. One discrete way that has been used frequently in the literature is the graph sampling method. Graph sampling was originally defined in [3] as a sampling-based comparison between graphs embedded in the plane that takes into account both the geometry and the topology of the immersed graphs. Graph sampling follows two steps: sampling two embedded graphs with points and matching the corresponding points to each other in a one-to-one manner. The proportion of the matched pairs can be used later to explain the similarities and differences between the two said graphs. Despite its simple definition, Graph sampling introduces many variables along the way that can affect the final results. Sampling interval, sampling function, matching function and whether to take bearing into consideration are some examples of these variables. Furthermore, while graph sampling is a fairly effective approach for map comparison, it is still a discrete method and simply counting the number of matched samples is not a reliable measurement in many cases. A possible artifact of this method is that a single road can be matched to several roads (see Fig. 1). Another common problem with graph sampling on reconstructed maps is finding a suitable ground truth map that only contains roads that are covered in the GPS data. This particular issue results in inconsistent evaluations among experiments in the literature [2].

In this paper we introduce a continuous alternative to this method that uses Fréchet distance in a length-preserving setting and avoids the issues that are mentioned above. For this purpose, one needs to match (continuous) segments of two graphs. There is a related work that matches segments of two curves using Fréchet distance, the so-called partial Fréchet matching [4]. The authors give a polynomial-time dynamic programming algorithm if L_1 or L_∞ are used as the underlying metrics. Here, we put a length-preserving constraint on the matched segments and our goal is to maximize the total length of them. Furthermore, we generalize this definition for graphs in $\mathbb{R}^2$.

2 Measuring Length-Preserving Fréchet Correspondence

In this paper, we consider paths and graphs in $\mathbb{R}^2$, where we define the distance between two points x and y as the two-norm of their difference, denoted $||x - y||$. We denote the (open) metric ball centered at $x \in \mathbb{R}^2$ with radius $\delta \in \mathbb{R}_{\geq 0}$ by: $\mathbb{B}(x, \delta) := \{y \in \mathbb{R}^2 \mid ||x - y|| < \delta\}$.

*Partially supported by National Science Foundation grant 2046730.
†Partially supported by National Science Foundation grant CCF 2107434.

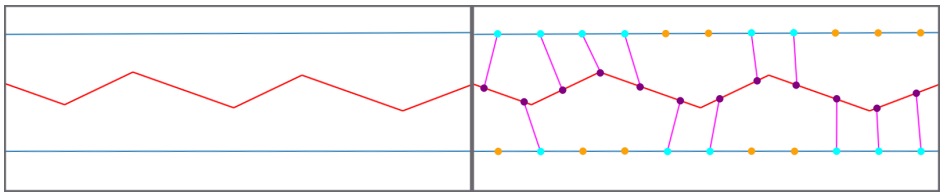


Figure 1: Graph sampling on two road maps G and H shown with blue and red respectively. Cyan and purple points are matched samples on G and H, while orange points indicate samples on G that are not matched. A magenta line shows a matching between a pair of samples.

Let $\mathcal{G}$ be the set of all pairs (G, ϕ), where G is an abstract graph and $\phi\colon G \to \mathbb{R}^2$ is a continuous map, up to the following equivalence: we say two pairs (G, ϕ_G) and (H, ϕ_H) are equivalent if there exists a homeomorphism $h\colon G \to H$ such that $\phi_G = \phi_H \circ h$. Throughout, we assume that each abstract graph G is comprised of a finite set of vertices, denoted $V(G)$, and a finite set of edges, denoted by $E(G)$. Given two points $x, y \in G$, we measure their distance by considering all paths in G and find the path whose Lebesgue measure (or, length) under ϕ is smallest:

$$L(x, y; (G, \phi)) := \inf_{p\colon [0,1] \to G} \text{len}(\phi(p)),$$

where p ranges over all continuous maps such that $p(0) = x$ and $p(1) = y$. We measure the *length* of (G, ϕ) as the total Lebesgue measure of all edges in the graph: $\text{len}(G) = \sum_{e \in E(G)} \text{len}\, \phi(e)$.

Let $(A, \phi_A), (B, \phi_B) \in \mathcal{G}$. Let $f\colon A \to B$ be a function that is homeomorphic onto its image. We say that f is *length-preserving* if for each $x, y \in A$, the length of the shortest path in (A, ϕ_A) from $\phi_A(x)$ to $\phi_A(y)$ is equal to the length of the shortest path in (B, ϕ_B) from $\phi_B(f(x))$ to $\phi_B(f(y))$. More formally, f is length-preserving iff for all $x, y \in A$, $L(x, y; (A, \phi)) = L(f(x), f(y); (B, \phi_B))$.

Let $(G, \phi_G), (H, \phi_H) \in \mathcal{G}$. Let $\varepsilon > 0$. Let C be a connected subgraph of G, and let $h\colon C \to H$ be a continuous map that is homeomorphic onto its image. We measure the length of the subgraph of C that maps within ε of $h(C)$. More formally, let

$$C_h^\varepsilon := \{x \in C \mid ||\phi_G(x) - \phi_H(h(x))|| \leq \varepsilon \text{ and } \exists \delta > 0\colon h|_{\mathbb{B}(x,\delta)} \text{ is length-preserving}\}. \tag{1}$$

We can interpret this as a subgraph of (G, ϕ_G). Specifically, $(C_h^\varepsilon, \phi_G|_{C_h^\varepsilon})$ is in $\mathcal{G}$, but we note that it may not

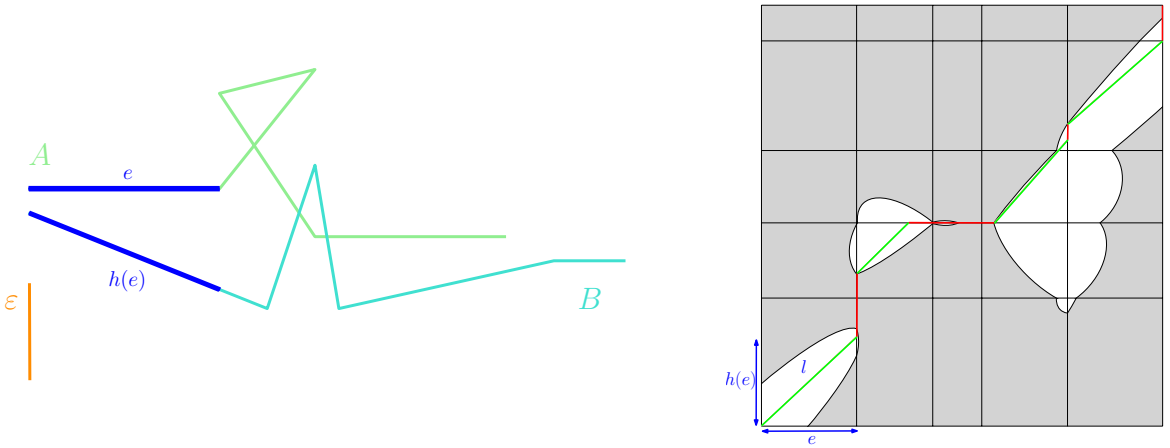


Figure 2: Free-space diagram of two curves A and B for a given ε. The green (slope-one) line segments correspond to length-preserving matchings. e is a line segment on A that was matched with $h(e)$ on B.

be a path-connected graph. To highlight the relation between length-preserving and paths in the free space diagram, consider Fig. 2. Similar to [4], in this example, we are looking for a monotone path from bottom-left to top-right but only maximizing the length of slope-one segments in the white space. A line segment l

indicates a length-preserving matching because the corresponding matched line segments, e and $h(e)$ on the two curves have the same length if and only if l is slope-one. Fig. 3 shows an example of C_h^ε. Note that C_h^ε itself is not necessarily a connected graph.

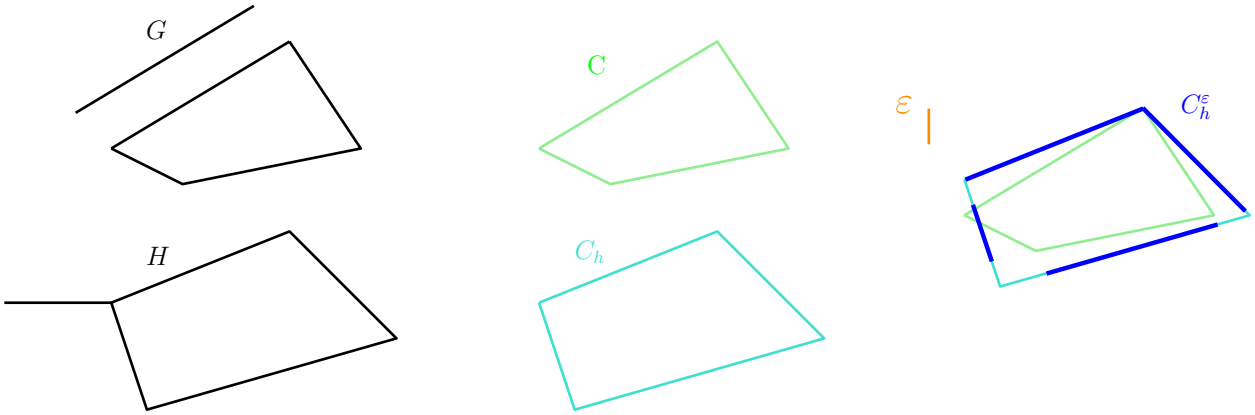


Figure 3: Computing C_h^ε on two given graphs, G and H. C is a connected subgraph of G. $C_h := h(C)$ is a subgraph of H such that C and C_h are homeomorphic. For a given ε, dark blue segments are C_h^ε.

Given this set-up, the length of the maximum *length-preserving* Fréchet correspondence is

$$L_{FC}(G, H; \varepsilon) \mapsto \sup_{C \subset G} \sup_{h: C \mapsto H} \operatorname{len}(C_h^\varepsilon).$$

While C_h^ε only requires small portions of h to be length-preserving, we show that indeed all connected components are length-preserving.

Lemma 1 (Path-Connected Components Are Length-Preserving)**.** *Let C_h^ε be defined as in Equation (1). Then, each restriction of h to a path-connected component of C_h^ε is a length-preserving map.*

Proof. Let $\widetilde{C}$ be a path-connected component of C_h^ε. Let $x, y \in \widetilde{C}$. Since $\widetilde{C}$ is a path-connected space, let $p\colon [0,1] \to \widetilde{C}$ be a path that starts at x and ends at y. Then, by the definition of C_h^ϵ, for each $t \in [0,1]$, there exists a $\delta_t > 0$ such that h restricted to $\mathbb{B}(p(x), \delta_t)$ is length-preserving. Let $\mathcal{U} := \{\mathbb{B}(p(x), \delta_t)\}_{t\in[0,1]}$. Since $\operatorname{Im}(p)$ is a compact subspace of $\mathbb{R}^2$, there exists a finite subcover $\widehat{\mathcal{U}}$ of $\mathcal{U}$. Then, there exists a decomposition of p such that each subpath lies entirely in at least one open set in $\widehat{\mathcal{U}}$. Let $\{p_i\}_{i=1}^n$ be one such decomposition. Then, we know that $\operatorname{len}(p) = \sum_i \operatorname{len} p_i$. Since each $\operatorname{Im} p_i$ is contained in some $U \in \widehat{\mathcal{U}}$ and since h restricted to U is length-preserving, we know that $\operatorname{len} p_i = \operatorname{len} h(p_i) = \operatorname{len} h(p)$. Hence, $\operatorname{len}(p) = \operatorname{len}(h(p))$ for each path from x to y, which means that $L(x, y; (G, \phi_G)) = L(h(x), h(y); (H, \phi_H)$. Thus, we have shown that h restricted to $\widetilde{C}$ is length-preserving, as was to be shown. □

3 NP-Hardness

Unfortunately, deciding whether an optimal Length-Preserving Fréchet Correspondence is above a given threshold is NP-hard.

Theorem 2 (Maximum Length-Preserving Fréchet Correspondence is NP-hard)**.** *Deciding $L_{FC}(G, H; \varepsilon) > L$ is NP-hard, even if G consists of only one edge and H is a plane graph.*

Proof. We reduce from the Hamiltonian path problem in grid graphs, which is known to be NP-hard [5], even for induced grid graphs of degree at most three [6]. The vertex set of such a grid graph is a finite subset of $\mathbb{Z}^2$, and there is an edge between two vertices u, v if and only if $\|u - v\| = 1$.

Given such a grid graph $H' = (V', E')$, we construct the graph H as follows: for every vertex we add an edge to a new degree-1 vertex at distance > 1, see Figure 4. Formally, let $V'' = V' + (3/4, 3/4)$ be the set V' translated by $(3/4, 3/4)$ and $E'' = \{(v', v'') \in V' \times V'' \mid v'' = v' + (3/4, 3/4)\}$. We note that the edges in E''

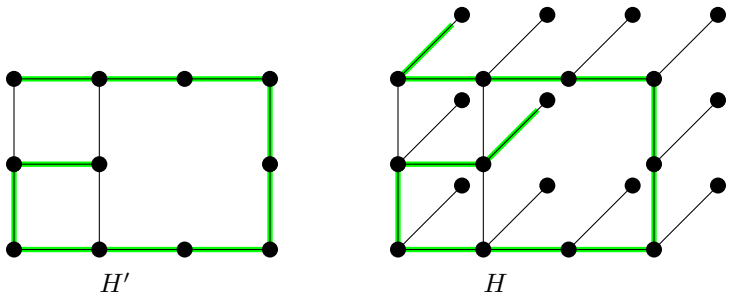


Figure 4: A grid graph H' with Hamiltonian path in green. The graph H and the image of G corresponding to the Hamiltonian path.

have length $\sqrt{2} \cdot 3/4 \approx 1.06$. We choose $H = (V' \cup V'', E' \cup E'')$. Without loss of generality, we can assume that the coordinates of the vertices of H are between 0 and $n = |V'| > 1$, since we can assume that H' is connected. Let G consist of only one edge $(0, n)$. We choose $\varepsilon = n$. We claim that if H' has a Hamiltonian path then $L_{FC}(G, H; \varepsilon) = n + 1$, and otherwise $L_{FC}(G, H; \varepsilon) < n + 1/5$. This then implies the theorem.

We have chosen ε sufficiently large such that h can map G onto any simple path in H (not necessarily ending at vertices). If H' has a Hamiltonian path, then we can map G length-preserving onto the corresponding path in H extended by parts (of length 1) of edges in E'' at the beginning and end. Thus, $L_{FC}(G, H; \varepsilon)$ is the length of the edge $(0, n)$, i.e, $n + 1$. If H' does not have a Hamiltonian path, then the longest simple path in H' that starts and ends at vertices has length at most $n - 2$. This implies that the longest simple path in H, not necessarily ending at vertices, has length at most $n - 2 + 2\sqrt{2} \cdot 3/4 \approx n + 0.12 < n + 1/5$. □

4 Discussion

For the simpler setting of comparing two curves, we expect that the framework from [4] can be applied to develop a dynamic programming algorithm for computing the length of an optimal correspondence for a given $\varepsilon > 0$. Similar to [4], we also need to compute a partial matching of maximum length inside the free space, just that we only measure the length-preserving portions. If we have two points (x, y) and $(x + \Delta x, y + \Delta y)$ on boundaries of the white space within a cell, then the the length-preserving portion that we can achieve between those two points is $\min(\Delta x, \Delta y)$. Thus, the main difference to [4] is that we have a piecewise linear function in Δx and Δy rather than just $\Delta x + \Delta y$. We therefore expect that their algorithm can be modified to our setting for two curves under the L_1 or L_∞ distances.

Assuming that we can compute $L_{FC}(G, H; \varepsilon)$ by dynamic programming for two curves, we expect that such an algorithm then generalizes to the case that H is a tree (or even the case that H has small treewidth). Another interesting case is when G also is a tree; it is related to the subtree isomorphism problem for which efficient algorithms exist; see e.g., [1].

The original aim of this research was to avoid issues with discontinuities in the graph sampling method. As an intermediate measure, we could also consider a version of L_{FC} where vertices have to be mapped to vertices, i.e., a length-preserving discrete Fréchet correspondence. For this measure NP-hardness follows even more directly, but also the algorithms become much simpler. The hardness proof assumes that there is a large number of vertices within a distance ε, and it would be interesting to develop algorithms for the case where the number of points in any ε-ball is constant.

References

[1] Amir Abboud, Arturs Backurs, Thomas Dueholm Hansen, Virginia Vassilevska Williams, and Or Zamir. Subtree isomorphism revisited. *ACM Transactions on Algorithms (TALG)*, 14(3):1–23, 2018.

[2] Jordi Aguilar, Kevin Buchin, Maike Buchin, Erfan Hosseini Sereshgi, Rodrigo I. Silveira, and Carola Wenk. Graph sampling for map comparison. *In submission*, 2021.

[3] James Biagioni and Jakob Eriksson. Inferring road maps from global positioning system traces. *Transportation Research Record: Journal of the Transportation Research Board*, 2291(1):61–71, 2012.

[4] 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*, SODA '09, page 645654, USA, 2009. Society for Industrial and Applied Mathematics.

[5] Alon Itai, Christos H. Papadimitriou, and Jayme Luiz Szwarcfiter. Hamilton paths in grid graphs. *SIAM Journal on Computing*, 11(4):676–686, 1982.

[6] Christos H. Papadimitriou and Umesh V. Vazirani. On two geometric problems related to the travelling salesman problem. *Journal of Algorithms*, 5(2):231–246, 1984.

SWEEPING POLYGONS WITH A VARIABLE-LENGTH LINE SEGMENT

Kien C. Huynh
Department of Computer Science
Stony Brook University
USA
kchuynh@cs.stonybrook.edu

Joseph S. B. Mitchell
Department of Applied Mathematics and Statistics
Stony Brook University
USA
joseph.mitchell@stonybrook.edu

ABSTRACT

We study the problem of sweeping every point in a polygon using a variable-length line segment. The entire line segment is constrained to stay inside the polygon at all times. The endpoints can move with different (bounded) speeds. Our objective is to minimize makespan: find a pair of trajectories for the endpoints so that the sweeping can be concluded as soon as possible. If the polygons are simple, we present a polynomial time $(4 + \epsilon)$-approximation algorithm.

1 Introduction

Given a polygon P in the plane, we want to sweep every point in it using a line segment. A point is consider swept only if it is touched by the segment. One can consider there to be two agents moving within P that are required to remain covisible at all times; the agents are the endpoints of a sweeping line segment, which is required to sweep over every point of P. Initially, the agents are assumed to start at the same position, $s \in P$. Without the loss of generality, we can also assume that each endpoint can move with speed at least 0 (stationary) and at most 1. The general goal is to find a good sweeping solution that covers every point in P while minimizing some cost functions. These can be overall sweeping duration, the maximum distance traversed between the two agents, or total distance traveled. In this abstract, we are focusing on finding a short makespan, i.e. minimizing the sweeping duration.

This problem is motivated by the problem introduced in [1], which involves sweeping a polygon with a connected chain of guards. The main difference is, in that problem, the guards at the beginning and the end of the chain must always stay on the boundary of P at all times. With such a setup, it is possible for the chain to always catch any moving target inside the polygon, as the chain maintains a "cleaned" portion of the domain.

Our current problem is similar but with limited resource: we only have two agents (guards), but we still want to efficiently sweep the polygon, searching for a static target. A closely related problem is to minimize the sum of distances traveled by two endpoints, which can be solved in polynomial time [4]. The same authors later provided an $O(n^2)$ algorithm to minimize the maximum length of the line segment [5]. Another related problem is the watchman route problem, in which an agent moves in order to see every point in the domain; see, e.g., [2].

In this abstract, our main contribution is a polynomial time approximation algorithm for any simple polygon. We also present our ideas behind proving the hardness of the version with holes and finding an approximation solution for the case with multiple line segments.

2 Minimizing makespan in a simple polygon

In this version, we want to minimize the overall time that is needed for the agents to finish sweeping every point in P. This can be seen as minimizing the maximum time spent by each agent (endpoint of the segment) during the sweep schedule.

We begin with a straightforward observation:

Lemma 2.1. *To completely sweep P, it is necessary that all convex vertices must be visited by at least one endpoint.*

This also applies to polygon with holes. The same is not true for reflex vertices, as those can be swept by the interior of a sweeping segment (when the reflex vertex falls interior to the segment, and the segment becomes tangent at the vertex). Moreover, it is possible for the line segment to sweep an edge of P without requiring any endpoint to move along that edge. For example, the line segment can already be parallel to an edge, to sweep that edge each endpoint only needs to move to its closest vertex of that edge. However, in any optimal solution, the segment can only sweep an edge in such a way at most once, which leads to the following lemma:

Lemma 2.2. *Let T be the minimum geodesic Steiner tree spanning every convex vertex of P, then $OPT \geq \frac{1}{2}|T|$.*

Proof. Since the line segment starts at the same location at the beginning, the trajectories of the two endpoints would make a connected graph. Using Lemma 2.1, this graph must connect all convex vertices. It follows that $OPT \geq \frac{1}{2}|T|$ by the definition of T. □

Before introducing our approximation algorithm, the following definition is necessary:

Definition 2.1. *A polygon is called a* crescent *if it consists of one convex chain and one reflex chain, both chains connect at their two vertices at the ends.*

Such a polygon can be swept by one segment in time at most equal to the length of the convex chain, i.e. the longer chain. To sweep it, the segment will start at one of the shared vertices of the chains and will continue sweeping until it reaches the other vertex. In the entire motion, one endpoint of the segment will always stay on the convex chain and the other on the reflex chain. This type of polygon is employed in proving the following theorem:

Theorem 2.1. *There exists a $(4+\epsilon)$-approximation algorithm that can be computed in polynomial time.*

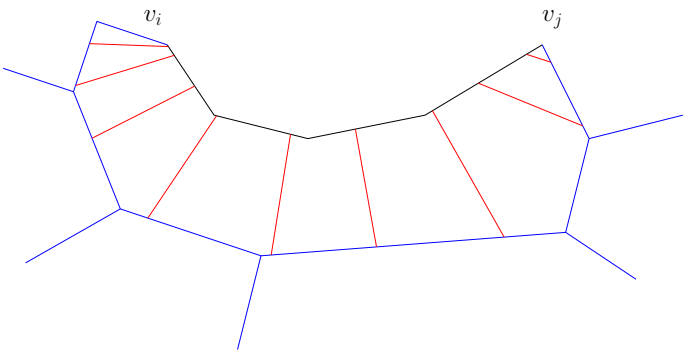


Figure 1: A crescent formed by the minimum geodesic Steiner tree T, colored in blue, and a reflex chain v_iv_j, colored in black. This type of subpolygon will always be sweepable by the line segment, depicted in red, with one endpoint moving along the path of T connecting v_i to v_j and the other along the corresponding path of the boundary.

Proof. Recall that T is the minimum geodesic Steiner tree that spans every convex vertices of P. Any two consecutive convex vertices v_i, v_j (with a possible reflex chain in between), along with the edges of T connecting v_i to v_j, will form a crescent. See Figure 1 for an illustration. The reason is that the path in T connecting the two convex vertices will always be left-turning (if we are traversing the polygon in clockwise order) since T is a minimum Steiner tree. The subpolygon made of this path of T, which is a convex chain, and the reflex chain from v_i to v_j will be a crescent.

Using the FPTAS from [3] we can compute a $(1+\epsilon')$-approximate minimum Steiner tree spanning all convex vertices in any simple polygon. The running time is $O(n_{\epsilon'}k^2(n_{\epsilon'}+k))$, where $n_{\epsilon'}$ is the number of vertices of the polygon plus the number of grid points used in the approximation and k is the number of convex vertices. In case a subpolygon is not a crescent, i.e. there exists a path from v_i to v_j that is not always left-turning, then the corresponding part of the tree is locally optimal. We can fix that by adjusting any right-turning vertex so that it is left-turning, or at least colinear with the two vertices before and after it.

Our sweep schedule will be as follows: starting from any convex vertex, the two agents will sweep from one convex vertex to the next using the corresponding walkable subpolygon generated by the above method. One agent will always follow the edges of T while the other agent will follow the boundary of the polygon. The total duration of this sweeping schedule will be the time it takes an agent, moving at full speed, to traverse twice the length of the tree: $2(1+\epsilon')|T| \leq 4(1+\epsilon')OPT$. If we set $\epsilon' = \epsilon/4$ then the theorem follows. □

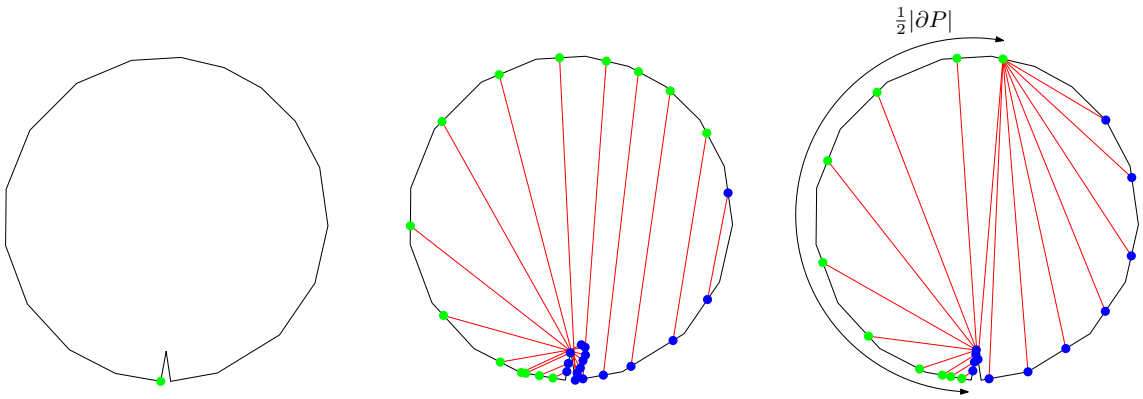


Figure 2: An example in which the optimal solutions for the makespan minimization, shown in the middle, and maximum distance minimization, shown on the right, are different. This assumes that both endpoints start at the same location, depicted as a green dot in the left figure. In the makespan minimization solution, the green endpoint will continue to move even if its traveled distance is already longer than $|\partial P|/2$. In contrast, if minimizing the maximum distance traveled by an agent, the solution has each agent traveling distance $|\partial P|/2$, while the overall duration of the sweep is greater than that of the minimum makespan solution.

It is interesting to note that minimizing the makespan of the sweeping motion is not the same as minimizing the maximum distance traveled by each endpoint. This can be seen in the example shown in Figure 2. The problem of min-maxing the distance traveled can be useful if we consider each endpoint as an agent with limited fuel, in that case we might not want one agent to travel much more than the other.

3 Sweeping a polygon with holes

The case of sweeping a polygon with holes is work in progress. In the talk and full paper, we expect to be able to show:

Claim 3.1. *The problem of minimizing the makespan to sweep a polygon with holes is NP-hard, even if the holes are convex.*

The (still incomplete) proof attempt uses a reduction from Hamiltonian path in a (triangular) grid graph.

Given the hardness of this case, we explore polynomial time approximation algorithms, based on lower bounds in terms of the total length of the boundary of the domain, and a minimum Steiner tree that spans all holes.

4 Sweeping with multiple line segments

In this problem, we are given k segments ($2k$ endpoints), each of which can start at any point on the boundary of P. We again seek to minimize the makespan of the sweeping schedule, i.e. minimizing the maximum time traveled by any endpoint of any segment. If we apply the reasoning similar to Lemma 2.2, then one lower bound of the optimal solution to this problem is $(|\partial P| - \sum_{e \in E_k} |e|)/(2k)$ where E_k is the set of k largest edges of P. This bound implies that we can find k largest edges that also partition the boundary of P into k parts of equal length. But that might not always be true, hence we believe that this bound could be improved.

Our approach to approximating OPT is to reuse the result in Theorem 2.1. We can partition the minimum geodesic Steiner tree T into k smaller trees of the same length. For each such tree, we can apply the same motion plan for each line segment and let them sweep from one crescent to another.

References

[1] Alon Efrat, Leonidas J Guibas, Sariel Har-Peled, David C Lin, Joseph SB Mitchell, and TM Murali. Sweeping simple polygons with a chain of guards. In *Proceedings of the eleventh annual ACM-SIAM symposium on Discrete algorithms*, pages 927–936, 2000.

[2] Joseph SB Mitchell. Approximating watchman routes. In *Proceedings of the twenty-fourth annual ACM-SIAM symposium on Discrete algorithms*, pages 844–855. SIAM, 2013.

[3] J Scott Provan. An approximation scheme for finding steiner trees with obstacles. *SIAM Journal on Computing*, 17 (5):920–934, 1988.

[4] Xuehou Tan and Bo Jiang. Optimum sweeps of simple polygons with two guards. In *International Workshop on Frontiers in Algorithmics*, pages 304–315. Springer, 2010.

[5] Xuehou Tan and Bo Jiang. Minimization of the maximum distance between the two guards patrolling a polygonal region. In *Frontiers in Algorithmics and Algorithmic Aspects in Information and Management*, pages 47–57. Springer, 2012.

Topological Graph Neural Networks

Max Horn[1, 2, *] **Edward De Brouwer**[3, *] **Michael Moor**[1, 2] **Yves Moreau**[3]
Bastian Rieck[1, 2, 4, †] **Karsten Borgwardt**[1, 2, †]

[1]Department of Biosystems Science and Engineering, ETH Zurich, 4058 Basel, Switzerland
[2]SIB Swiss Institute of Bioinformatics, Switzerland
[3]ESAT-STADIUS, KU LEUVEN, 3001 Leuven, Belgium
[4]B.R. is now with the Institute of AI for Health, Helmholtz Zentrum München, Neuherberg, Germany
[*]These authors contributed equally.
[†]These authors jointly supervised this work.

Abstract

Graph neural networks (GNNs) are a powerful architecture for tackling graph learning tasks, yet have been shown to be oblivious to eminent substructures such as cycles. We present TOGL, a novel *layer* that incorporates global topological information of a graph using persistent homology. TOGL can be easily integrated into *any type* of GNN and is strictly more expressive in terms of the Weisfeiler–Lehman graph isomorphism test. Augmenting GNNs with our layer leads to improved predictive performance for graph and node classification tasks, both on synthetic data sets (which can be classified by humans using their topology but not by ordinary GNNs) and on real-world data.

1 Introduction

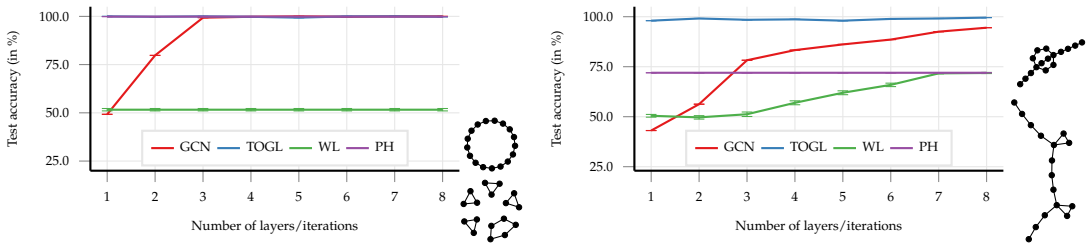
Graphs are a natural description of structured data sets in many domains, including bioinformatics, image processing, and social network analysis. Numerous methods address the two dominant graph learning tasks of graph classification or node classification. In particular, *graph neural networks* (GNNs) describe a flexible set of architectures for such tasks and have seen many successful applications over recent years [15]. At their core, many GNNs are based on iterative message passing schemes. Since these schemes are collating information over the neighbours of every node, GNNs cannot necessarily capture certain topological structures in graphs, such as cycles [1]. These structures, however, are relevant for certain applications, such as the analysis of molecular graphs, whose classification necessitates knowledge about connectivity information [7, 12].

In this paper, we address this issue by proposing a Topological Graph Layer (TOGL) that can be easily integrated into any GNN to make it 'topology-aware.' Our method is rooted in the nascent field of topological data analysis (TDA), which focuses on describing coarse structures that can be used to describe the shape of complex structured and unstructured data sets. We thus obtain a generic way to augment existing GNNs and increase their expressivity in graph learning tasks. Figure 1 provides a motivational example that showcases the potential benefits of using topological information: (i) high predictive performance is reached *earlier* for a *smaller* number of layers, and (ii) learnable topological representations outperform fixed ones if more complex topological structures are present in a data set.

2 Computational Topology

We consider undirected graphs of the form $G = (V, E)$ with a set of vertices V and a set of edges $E \subseteq V \times V$. The basic topological features of such a graph G are the number of connected components β_0 and the number of cycles β_1. These counts are also known as the 0-dimensional and 1-dimensional *Betti numbers*,

This is a shortened version of our work 'Topological Graph Neural Networks' (arXiv:2102.07835), which is currently under review at ICLR 2022. This version has been significantly condensed.



(a) CYCLES (b) NECKLACES

Figure 1: As a motivating example, we introduce two topology-based data sets whose graphs can be easily distinguished by humans; the left data set can be trivially classified by all topology-based methods (since access to β_0 or β_1 is sufficient for classification), while the right data set necessitates *learnable* topological features. We show the performance of using a GCN *and* TOGL, our method, as compared to a 'pure' GCNs, the Weisfeiler–Lehman (WL) graph kernel, and a static topological method (PH) based on a degree filtration of a graph. When there is more than one layer, we use TOGL as the second layer, while all remaining layers are regular GCN layers (we still denote this hybrid model by 'TOGL' in the legend).

respectively, and can be computed efficiently. Betti numbers are invariant under graph isomorphism [5, pp. 103–133]. The expressivity of Betti numbers can be increased by assuming the existence of a *graph filtration*, i.e. a sequence of nested subgraphs of G such that $\emptyset = G^{(0)} \subseteq G^{(1)} \subseteq G^{(2)} \subseteq \cdots \subseteq G^{(n-1)} \subseteq G^{(n)} = G$. A filtration makes it possible to obtain more insights into the graph by 'monitoring' topological features of *each* $G^{(i)}$ and calculating their topological relevance, also referred to as their *persistence*. If a topological feature appears for the first time in $G^{(i)}$ and disappears in $G^{(j)}$, we assign this feature a persistence of $j - i$. Equivalently, we can represent the feature as a tuple (i, j), which we collect in a *persistence diagram* $\mathcal{D}$. This process was formalised and extended to a wider class of structured data sets, namely simplicial complexes, and is known under the name of *persistent homology*. One of its core concepts is the use of a filtration function $f\colon V \to \mathbb{R}^d$, with $d = 1$ typically, to accentuate certain structural features of a graph. This replaces the aforementioned tuples of the form (i, j) by tuples based on the values of f, i.e. (f_i, f_j). Persistent homology has shown excellent promise in different areas of machine learning research (see Hensel et al. [6] for a recent survey); choosing or learning an appropriate filtration function f is crucial for high predictive performance [7, 18].

Notation. We denote the calculation of persistence diagrams of a graph G under some filtration f by $\mathbf{ph}(G, f)$. This will result in two persistence diagrams $\mathcal{D}_0, \mathcal{D}_1$, containing information about topological features in dimension 0 (connected components) and dimension 1 (cycles). The cardinality of $\mathcal{D}_0$ is equal to the number of nodes n in the graphs and each tuple in the 0-dimensional diagram is associated with the *vertex* that created it. The cardinality of $\mathcal{D}_1$ is the number of cycles.

3 TOGL: A Topological Graph Layer

TOGL is a new type of graph neural network layer that is capable of utilising multi-scale topological information of input graphs. In this section, we give a brief overview of the components of this layer before discussing algorithmic details, theoretical expressivity, computational complexity, and limitations. The layer takes as input a graph $G = (V, E)$ equipped with a set of n vertices V and a set of edges E, along with a set of d-dimensional node attribute vectors $x^{(v)} \in \mathbb{R}^d$ for $v \in V$. These node attributes can either be node features of a data set or hidden representations learnt by some GNN. We employ a family of k vertex filtration functions of the form $f_i\colon \mathbb{R}^d \to \mathbb{R}$ for $i = 1, \ldots, k$. Each filtration function f_i can focus on different properties of the graph. The image of f_i is finite and results in a set of node values $a_i^{(1)} < \cdots < a_i^{(n)}$ such that the graph G is filtered according to $\emptyset = G_i^{(0)} \subseteq G_i^{(1)} \subseteq \cdots \subseteq G_i^{(n)} = G$, where $G_i^{(j)} = \big(V_i^{(j)}, E_i^{(j)}\big)$, with

$V_i^{(j)} := \left\{ v \in V \mid f_i\left(x^{(v)}\right) \leq a_i^{(j)} \right\}$, and $E_i^{(j)} := \left\{ v, w \in E \mid \max\left\{ f_i\left(x^{(v)}\right), f_i\left(x^{(w)}\right) \right\} \leq a_i^{(j)} \right\}$. Given this filtration, we calculate a set of persistence diagrams, i.e. $\mathbf{ph}(G, f_i) = \left\{ \mathcal{D}_i^{(0)}, \ldots, \mathcal{D}_i^{(l)} \right\}$. We fix $l = 1$ (i.e. we are capturing connected components and cycles) to simplify our current implementation, but our layer may in principle be extended to arbitrary values of l. In order to benefit from representations that are trainable end-to-end, we use an *embedding function* $\Psi^{(l)} \colon \left\{ \mathcal{D}_1^{(l)}, \ldots, \mathcal{D}_k^{(l)} \right\} \to \mathbb{R}^{n' \times d}$ for embedding persistence diagrams into a high-dimensional space that will be used to obtain the vertex representations, where n' is the number of *vertices* n if $l = 0$ and the number of *edges* if $l = 1$. This step is crucial as it enables us to use the resulting topological features as node features, making TOGL a layer that can be integrated into arbitrary GNNs. We experimented with different embedding functions Ψ; the results described in this work are based on a `DeepSets` approach [17], making it possible to take interactions between different points in the persistence diagram into account. We compute our family of k vertex-based filtrations using $\Phi \colon \mathbb{R}^d \to \mathbb{R}^k$, an MLP with a single hidden layer, such that $f_i := \pi_i \circ \Phi$, i.e. the projection of Φ to the ith dimension. We apply Φ to the hidden representations $x^{(v)}$ of all vertices in the graph. Moreover, we treat the resulting persistence diagrams in dimension 0 and 1 differently. For dimension 0, we have a bijective mapping of tuples in the persistence diagram to the vertices of the graph, which was previously exploited in topological representation learning [10]. Therefore, we aggregate $\Psi^{(0)}$ with the original node attribute vector $x^{(v)}$ of the graph in a residual fashion, i.e. $\tilde{x}^{(v)} = x^{(v)} + \Psi^{(0)}\left(\mathcal{D}_1^{(0)}, \ldots, \mathcal{D}_k^{(0)}\right)[v]$, where $\Psi^{(0)}[v]$ denotes taking vth row of $\Psi^{(0)}$ (i.e the topological embedding of vertex v). The output of our layer for dimension 0 therefore results in a new representation $\tilde{x}^{(v)} \in \mathbb{R}^d$ for each vertex v, making it compatible with any subsequent (GNN) layers. By contrast, $\Psi^{(1)}$ is pooled into a graph-level representation, to be used in the final classification layer of a GNN. This is necessary because there is no bijective mapping to the vertices, but rather to edges.

Expressive Power. The expressive power of graph neural networks is well-studied [3, 16] and typically assessed via the iterative Weisfeiler–Lehman label refinement scheme, denoted as WL[1]. Given a graph with an initial set of vertex labels, WL[1] collects the labels of neighbouring vertices for each vertex in a multiset and 'hashes' them into a new label, using a perfect hashing scheme so that vertices/neighbourhoods with the same labels are hashed to the same value. This procedure is repeated and stops either when a maximum number of iterations has been reached or no more label updates happen. The result of each iteration h of the algorithm for a graph G is a feature vector $\phi_G^{(h)}$ that contains individual label counts. Originally conceived as a test for graph isomorphism [14], WL[1] has been successfully used for graph classification [11]. Surprisingly, Xu et al. [16] showed that standard graph neural networks based on message passing are *no more powerful* than WL[1]. It turns out that persistent homology (and TOGL by transitivity) extend the expressivity of WL[1]. We first state a 'lower-bound' type of result.

Theorem 1. *Persistent homology is* at least *as expressive as WL[1], i.e. if the WL[1] label sequences for two graphs G and G' diverge, there exists an injective filtration f such that the corresponding 0-dimensional persistence diagrams $\mathcal{D}_0$ and $\mathcal{D}_0'$ are not equal.*

Proof sketch. We first assume the existence of a sequence of WL[1] labels and show how to construct a filtration function f from this. While f will result in persistence diagrams that are different, thus serving to distinguish G and G', it does not necessarily satisfy injectivity. We therefore show that there is an injective function $\tilde{f}$ that is arbitrarily close to f and whose corresponding persistence diagrams $\widetilde{\mathcal{D}_0}$, $\widetilde{\mathcal{D}_0'}$ do *not* coincide. Please refer to the extended version of this preprint[1] for more details. □

To prove that persistent homology and TOGL are more expressive than a GCN, we show that there are pairs of graphs G, G' that cannot be distinguished by WL[1] but that *can* be distinguished by $\mathbf{ph}(\cdot)$ and by TOGL, respectively: let G be a graph consisting of the disjoint union of two triangles, i.e.  , and let G' be a graph consisting of a hexagon, i.e.  . WL[1] will be unable to distinguish these two graphs because all multisets in every iteration will be the same. Persistent homology, by contrast, can distinguish G from G'

[1] `https://arxiv.org/abs/2102.07835`

Table 1: Predictive performance of our method. We depict the test accuracy obtained on various benchmark data sets when only considering structural information (i.e. the network has access to *uninformative* node features).

	Graph classification				*Node classification*	
METHOD	DD	ENZYMES	MNIST	PROTEINS	CLUSTER	PATTERN
GAT-4	63.3±3.7	21.7± 2.9	63.2±10.4	67.5± 2.6	16.7± 0.0	58.3±8.8
GIN-4	**75.6±2.8**	21.3± 6.5	83.4± 0.9	**74.6± 3.1**	16.4± 0.1	84.8±0.0
GCN-4 (*baseline*)	68.0±3.6	22.0± 3.3	76.2± 0.5	68.8± 2.8	16.7± 0.0	85.6±0.0
GCN-3-TOGL-1	75.1±2.1	**30.3± 6.5**	**84.8± 0.4**	73.8± 4.3	**16.8± 0.0**	**86.7±0.0**
GCN-3-TOGL-1 (static)	68.0±2.4	23.7± 5.4	82.9± 0.0	71.2± 5.1	**16.8± 0.0**	85.8±0.0

using their Betti numbers. We have $\beta_0(G) = \beta_1(G) = 2$, because G consists of two connected components and two cycles, whereas $\beta_0(G') = \beta_1(G') = 1$ as G' only consists of one connected component and one cycle. Together with Theorem 1, this example implies that persistent homology is *strictly* more powerful than WL[1].

4 Experiments

Next to the synthetic data sets shown in Figure 1, we also applied our method to standard graph/node classification benchmarking data sets, albeit with a slight 'twist:' we remove all node attribute features and replace them by random features. This ensures that classification performance is driven *only* by topological information and nothing else. We compare a hybrid model, consisting of a GCN with three layers and a single TOGL layer as the *second* layer,[2] with several well-known graph neural network architectures [2, 8, 13]. Table 1 depicts the results for graph and node classification tasks on such graphs; we observe that the performance of GCN-3-TOGL-1 *always* outperforms the GCN-4 baseline. For MNIST, the gains are substantial, with an increase of more than 8%, but other data sets exhibit similar improvements. This demonstrates the utility of TOGL in making additional structural information available to improve classification performance. While this was not the primary goal of this experiment, we also see that we perform favourably compared to other graph neural networks (in the case of ENZYMES, we even outperform them by a margin of 8%).

5 Conclusion

We presented TOGL, a generically-applicable layer that incorporates topological information into any GNN architecture. We proved that TOGL, due to its filtration functions (i.e. input functions) being learnable, is more expressive than WL[1], the Weisfeiler–Lehman test for graph isomorphism. On data sets with pronounced topological structures, we found that our method helps a GCN achieve high predictive performance. For future work, we are most interested in the investigation of additional regularisation strategies for improving the training process. Furthermore, we hypothesise that the use of different filtration types [9], together with improved persistent homology algorithms [4], will prove beneficial.

[2] This hybrid model has approximately the same number of parameters as its comparison partner, a GCN with four layers, thus ensuring that the comparison is fair.

References

[1] G. Bouritsas, F. Frasca, S. Zafeiriou, and M. M. Bronstein. Improving graph neural network expressivity via subgraph isomorphism counting. *arXiv preprint 2006.09252*, 2021.

[2] X. Bresson and T. Laurent. Residual gated graph convnets. *arXiv preprint arXiv:1711.07553*, 2017.

[3] Z. Chen, L. Chen, S. Villar, and J. Bruna. Can graph neural networks count substructures? *arXiv preprint arXiv:2002.04025*, 2020.

[4] D. Cohen-Steiner, H. Edelsbrunner, and J. Harer. Extending persistence using Poincaré and Lefschetz duality. *Foundations of Computational Mathematics*, 9(1):79–103, 2009.

[5] A. Hatcher. *Algebraic topology*. Cambridge University Press, 2002.

[6] F. Hensel, M. Moor, and B. Rieck. A survey of topological machine learning methods. *Frontiers in Artificial Intelligence*, 4, 2021. ISSN 2624-8212. doi: 10.3389/frai.2021.681108.

[7] C. D. Hofer, F. Graf, B. Rieck, M. Niethammer, and R. Kwitt. Graph filtration learning. In *Proceedings of the 37th International Conference on Machine Learning (ICML)*, volume 119 of *Proceedings of Machine Learning Research*, pages 4314—4323. PMLR, 2020.

[8] T. N. Kipf and M. Welling. Semi-supervised classification with graph convolutional networks. In *Proceedings of the 5th International Conference on Learning Representations (ICLR)*, 2017.

[9] N. Milosavljević, D. Morozov, and P. Skraba. Zigzag persistent homology in matrix multiplication time. In *Proceedings of the 27th Annual Symposium on Computational Geometry (SoCG)*, pages 216–225, 2011.

[10] M. Moor, M. Horn, B. Rieck, and K. Borgwardt. Topological autoencoders. In *Proceedings of the 37th International Conference on Machine Learning (ICML)*, volume 119 of *Proceedings of Machine Learning Research*, pages 7045–7054. PMLR, 2020.

[11] N. Shervashidze and K. Borgwardt. Fast subtree kernels on graphs. In Y. Bengio, D. Schuurmans, J. Lafferty, C. Williams, and A. Culotta, editors, *Advances in Neural Information Processing Systems*, volume 22, pages 1660–1668. Curran Associates Inc., 2009.

[12] N. Swenson, A. S. Krishnapriyan, A. Buluc, D. Morozov, and K. Yelick. PersGNN: Applying topological data analysis and geometric deep learning to structure-based protein function prediction. *arXiv preprint arXiv:2010.16027*, 2020.

[13] P. Veličković, G. Cucurull, A. Casanova, A. Romero, P. Liò, and Y. Bengio. Graph attention networks. In *International Conference on Learning Representations*, 2018.

[14] B. Weisfeiler and A. A. Lehman. The reduction of a graph to canonical form and the algebra which appears therein. *Nauchno–Technicheskaja Informatsia*, 9:12–16, 1968.

[15] Z. Wu, S. Pan, F. Chen, G. Long, C. Zhang, and P. S. Yu. A comprehensive survey on graph neural networks. *IEEE Transactions on Neural Networks and Learning Systems*, 32(1):4–24, 2021.

[16] K. Xu, W. Hu, J. Leskovec, and S. Jegelka. How powerful are graph neural networks? In *International Conference on Learning Representations*, 2019.

[17] M. Zaheer, S. Kottur, S. Ravanbakhsh, B. Poczos, R. R. Salakhutdinov, and A. J. Smola. Deep sets. In *Advances in Neural Information Processing Systems*, volume 30, pages 3391–3401. Curran Associates, Inc., 2017.

[18] Q. Zhao and Y. Wang. Learning metrics for persistence-based summaries and applications for graph classification. In *Advances in Neural Information Processing Systems 32 (NeurIPS)*, pages 9855–9866. Curran Associates, Inc., 2019.

THE COMBINATORIAL GENERATION OF OBJECTIVE TARGETS AND CONSTRAINTS FOR LARGE-SCALE TESTING OF OPTIMIZATION ROUTINES

ORIT DAVIDOVICH

Abstract. Our primary motivation is the large-scale testing and performance analysis of constrained optimization algorithms. To that end, we wish to randomly generate pairs (f, Ω) consisting of a continuous objective target f and a convex feasibility region Ω contained in its domain. Our challenge is to produce (f, Ω) in such a way that the true solution of the associated constrained optimization problem can be established combinatorially without recourse to an optimization algorithm.

1. Problem Statement

We wish to randomly generate pairs (f, Ω), where f is a continuous piecewise linear (PWL) function defined in a bounded, closed, convex domain $\mathrm{dom}(f) \subseteq \mathbb{R}^d$ and $\Omega \subseteq \mathrm{dom}(f)$ is a convex d-polytope. Our challenge is to produce (f, Ω) in such a way that the constrained optimization problem

$$\text{find } x^* \in \arg\min_{x \in \Omega} f(x) \tag{1}$$

can be resolved combinatorially without recourse to an optimization algorithm (similarly, for a maximization problem).

Our motivation is the large-scale testing and performance analysis of constrained optimization algorithms that seek to solve (1). Of particular interest to us are data-driven optimization algorithms that take a pair (D, Ω) of data D and constraints Ω as input. Such algorithms take an end-to-end approach that combines learning and optimization [7, 10, 11, 22, 23]. Given (f, Ω) and a hypothesis on the distribution of data points in $\mathrm{dom}(f)$, a pair (D, Ω) can easily be derived.

A continuous PWL function is in essence a combinatorial gadget. The domain of each of its affine pieces is a polytope. Subdividing polytopes into simplices, a continuous PWL function can be derived from a triangulation of its domain and set of real values, one for each triangulation vertex. The random generation of continuous PWL functions then becomes the random generation of triangulations [3, 6, 14, 15].

2. Triangulations

Let $S \subseteq \mathbb{R}^d$ be an n point configuration with $n \geq d+1$ and assume the affine span of S is $\mathbb{R}^d$. A full triangulation of S is an abstract d-dimensional *simplicial complex* $\mathcal{T}$ with vertex set S. The (finite) set of i-simplices of $\mathcal{T}$ is denoted by $\mathcal{T}^i$. A full triangulation satisfies $\mathcal{T}^0 = S$. The geometric realization of $\mathcal{T}$, defined as an abstract topological space, is denoted by $\|\mathcal{T}\|$.

Date: October 12, 2021.

We add a requirement that relates triangulations as abstract combinatorial gadgets to concrete embeddings in $\mathbb{R}^d$. We consider only those triangulations for which the following map $\iota : \|\mathcal{T}\| \to \mathbb{R}^d$ is an embedding: for each $\tau \in \mathcal{T}^i$ and $v \in \tau$, ι maps the vertex of the standard i-simplex $\|\tau\|$ labeled by v to v and restricts to an affine map on $\|\tau\|$. By abuse of notation, we use $\|\mathcal{T}\|$ also for the image of ι. Hence, we say that $\mathcal{T}$ is a *convex* triangulation when $\|\mathcal{T}\|$ is.

3. Piece-wise Linear Functions

Consider a pair $(\mathcal{T}, \mathcal{V})$ consisting of a triangulation $\mathcal{T}$ and a set $\mathcal{V} = \{r_v \in \mathbb{R} \,|\, v \in \mathcal{T}^0\}$. It gives rise to a continuous PWL function, which we will denote by $f_{(\mathcal{T},\mathcal{V})}$. Its domain is the bounded, closed $\mathrm{dom}(f) = \|\mathcal{T}\|$. To evaluate f at $v \in \mathrm{dom}(f)$, consider a simplex $\tau = \{v_1, \ldots, v_{d+1}\} \in \mathcal{T}^d$ such that $v \in \|\tau\|$ together with its associated values $r = (r_1, \ldots, r_{d+1})$ from $\mathcal{V}$. Define the homogenization of $v \in \mathbb{R}^d$ to be $\bar{v} = (v, 1)$. When simplices are non-degenerate, we are guaranteed a unique solution $\bar{x} \in \mathbb{R}^{d+1}$ to $\bar{A}_\tau \, \bar{x} = r$, where $\bar{A}_\tau$ is the matrix with rows $\{\bar{v}_1, \ldots, \bar{v}_{d+1}\}$, and we set $f(v) = \bar{x} \cdot \bar{v}$.

The random generation of continuous PWL functions $f_{(\mathcal{T},\mathcal{V})}$ is thus tantamount to the random generation of triangulations $\mathcal{T}$. There is a rich literature on triangulating polygons, polytopes, and point configurations [1, 4, 8, 9, 21]. The software TOPCOM defines the state of the art of triangulation generation and enumeration given a fixed point configuration [19, 13], and there are other triangulation generation schemes that produce optimal triangulations in some desired sense [2, 5]. Our purpose is different; we do not seek to solve the combinatorial problem of listing or enumerating all triangulations for a given point configuration, nor do we seek to produce optimal triangulations. Instead, our challenge is to generate sufficiently rich, that is, sufficiently random collections of continuous PWL functions.

In a computation environment, a simplex $\tau \in \mathcal{T}^d$ is numerically degenerate with respect to system tolerance tol, if it has effective zero volume, that is, $\mathrm{vol}(\tau) = |\det(\bar{A}_\tau)|/d! < \mathrm{tol}$. This prompts the need for some form of triangulation regularization [20]. Consider the set of volumes $\{\mathrm{vol}(\tau)/\mathrm{vol}(\mathcal{T})\}_{\tau \in \mathcal{T}^d}$ of a triangulation $\mathcal{T}$. It can be matched to a partition of $[0, 1]$, albeit not uniquely. Regardless of the partition produced, the distribution of the minimal volume will be equivalent to the distribution of the minimum in a random choice of $\#\mathcal{T}^d - 1$ points in $[0, 1]$, which is well known to be $\mathrm{Beta}(1, \#\mathcal{T}^d - 2)$. We formulate triangulation regularization as follows.

Definition 3.1 (Regularization Condition)**.** Given $\epsilon \in (0, 1)$, a triangulation $\mathcal{T}$ is subject to the ϵ-regularization condition if

$$\mathrm{CDF}_{\mathrm{MinVol}}(\min\{\mathrm{vol}(\tau)/\mathrm{vol}(\mathcal{T})\}_{\tau \in \mathcal{T}^d}) > \epsilon \tag{2}$$

where $\mathrm{MinVol} \sim \mathrm{Beta}(1, \#\mathcal{T}^d - 2)$.

4. Feasibility Region

We now turn to the generation of convex polytopes $\Omega \subseteq \mathrm{dom}(f)$, i.e., the feasibility region introducing constraints into the optimization problem

(1). Scenarios $\Omega \subsetneq \mathrm{dom}(f)$ are of particular interest when extending the generation of (f, Ω) to the generation of $(\mathcal{D}, \Omega)$. In real-world applications, governed by changing business consideration, historical business record may contain data points no longer considered feasible. Another reason to incorporate unfeasible data points into a synthetic data generation scheme is the testing of algorithms that seek to learn, perturb, or optimize the feasibility region itself.

Let $f = f_{(\mathcal{T},\mathcal{V})} : \|\mathcal{T}\| \to \mathbb{R}$ and consider a convex triangulation $\mathcal{T}_\circ$ that satisfies the property:

$$\forall \tau \in \mathcal{T}_\circ^d,\ \exists \tau' \in \mathcal{T}^d : \|\tau\| \subseteq \|\tau'\| \tag{3}$$

Setting $\Omega = \|\mathcal{T}_\circ\|$, we can easily solve (1), since

$$\arg\min_{x\in\Omega} f(x) = \arg\min_{v\in\mathcal{T}_\circ^0} f(v)$$

Given $\mathcal{T}$, our goal is then to produce $\mathcal{T}_\circ$ that satisfies Property (3). Generating f and Ω separately, and then producing $\mathcal{T}_\circ$ via intersection $\{\Omega \cap \|\tau\|\}_{\tau\in\mathcal{T}}$ is akin to solving multiple LP problems, since the intersection of affine hyperplanes must be constrained [16, 17]. This defeats the purpose of our combinatorial generation scheme. We therefore require a different approach.

5. Algorithm

Our proposed algorithm melds the generation of f and Ω with successive ϵ-regularized pulling and placing (or pushing) of new vertices. Crucially, the triangulation, updated at each incremental step, is maintained hierarchically.

Definition 5.1 (DAG of Simplices)**.** A DAG $i, t : E \to V$ of d-simplices has nodes $\tau = \{v_1, \ldots, v_{d+1}\} \subseteq \mathbb{R}^d$ in general position and strict inclusion of geometric realizations $\|\tau\| \subsetneq \|\tau'\|$ for edges $e = (\tau', \tau) \in E$ with $i(e) = \tau', t(e) = \tau$.

Our algorithm takes as input: (i) a random variable X over $\mathbb{R}^d$ to sample triangulation vertices, (ii) a random variable y over $\mathbb{R}$ to sample their respective values, (iii) an integer $n \geq d+1$ for the number of triangulation vertices, (iv) $p \in [0, 1]$ for Bernoulli trials, and (v) $\epsilon \in (0, 1)$ for regularization. It produces a list $L = [G_1, \ldots, G_N]$ of DAGs of d-simplices and a dictionary $\mathcal{V}$ of values, one for each unique vertex of some node $\tau \in V(G_i)$. The list L is constructed so that, for every $1 \leq m \leq N$, we have a convex triangulation $\mathcal{T}_m$ with $\mathcal{T}_m^d = \cup_{i\leq m}\mathrm{leaves}(G_i)$. In addition, $\mathcal{T}_m$ is a subtriangulation of $\mathcal{T}_{m'}$ for $m \leq m'$. The sequence $[\mathcal{T}_1, \ldots, \mathcal{T}_N]$ allows us to establish Theorem 5.2, which delivers what we have set out to do.

Theorem 5.2. *For $\rho \in [0, 1]$ and input $n \geq d + 1$, Algorithm 1 gives rise to (f, Ω) and $v_{\min}, v_{\max} \in \Omega$, where f is a continuous PWL function defined over a bounded, closed, convex domain $dom(f) \subseteq \mathbb{R}^d$ and $\Omega \subseteq dom(f)$ is a convex d-polytope, such that,*

(i) $vol(\Omega)/vol(dom(f)) \geq \rho$,
(ii) $\min_\Omega f = f(v_{\min})$ *and* $\max_\Omega f = f(v_{\max})$,
(iii) the solutions $v_{\min}, v_{\max}$ *are produced in* $\mathcal{O}(n)$.

Algorithm 1 PWL Functions with Constraints

Input: X r.v. in $\mathbb{R}^d$, y r.v. in $\mathbb{R}$, $n \geq d+1$, $p \in [0,1], \epsilon \in (0,1)$
Output: $(L = [G_1, \ldots, G_N], \mathcal{V})$

$\tau \leftarrow$ sample $\{v_1, \ldots, v_{d+1}\}$ from X
$\mathcal{V} \leftarrow \{v_i : r_i\}_{i=1}^{d+1}$, r_i sampled from y
$L \leftarrow [\text{DAG}(\text{roots} = \tau)]$
$i \leftarrow 0$
while $i < n - (d+1)$ **do**
  sample c from $\text{Bern}(p)$
  if $c = 0$ **then**
    sample v from X
    if $\exists G \in L : v \in G$ **then**
      $\tau \leftarrow$ minimal in G w.r.t. v
      $\tau_v \leftarrow$ subdivide τ w.r.t. v
      if $\text{RegCond}(\tau_v, \epsilon)$ **then**
        $\mathcal{V} \leftarrow \mathcal{V} + \{v : \text{ sample } y\}$
        $\text{children}(\tau) \leftarrow \tau_v^d$
        $i \leftarrow i + 1$
      end if
    else
      $\text{facets}(v) \leftarrow \{F \mid F \text{ facet of } \text{Conv}(L) \text{ visible to } v\}$
      $\text{nodes}(F) \leftarrow \{\tau \mid \exists G \in L : \tau \in G \text{ minimal w.r.t } F\}$
      $\text{leaves}(F) = \cup_{\tau \in \text{nodes}(F)} \text{leaves}(\tau)$
      $\text{facets}(F) \leftarrow \{F' \mid \exists \varsigma \in \text{leaves}(F) : F' \in \varsigma^{d-1}, \|F'\| \subseteq \|F\|\}$
      $\tau_{F'} \leftarrow F' \cup \{v\}, F' \in \text{facets}(F), F \in \text{facets}(v)$
      if $\text{RegCond}(\text{leaves}(F) \cup \{\tau_{F'}\}_{F' \in \text{facets}(F)}, \epsilon), \forall F \in \text{facets}(v)$ **then**
        $\mathcal{V} \leftarrow \mathcal{V} + \{v : \text{ sample } y\}$
        $L \leftarrow L + [\text{DAG}(\text{roots} = \{\tau_{F'} | F' \in \text{facets}(F), F \in \text{facets}(v)\})]$
        $i \leftarrow i + 1$
      end if
    end if
  else
    sample $v \in \cup_i \|G_i\|^{d-1}$
    $G_v \leftarrow \{G_i \mid v \in G_i\}$
    $\min(G_v) \leftarrow \{\tau \mid \exists G \in G_v : \tau \in G \text{ minimal w.r.t. } v\}$
    $\tau_v \leftarrow$ subdivide τ w.r.t. v, $\forall \tau \in \min(G_v)$
    if $\text{RegCond}(\tau_v, \epsilon), \forall \tau \in \min(G_v)$ **then**
      $\mathcal{V} \leftarrow \mathcal{V} + \{v : \text{ sample } y\}$
      $\text{children}(\tau) \leftarrow \tau_v^d, \forall \tau \in \min(G_v)$
      $i \leftarrow i + 1$
    end if
  end if
end while

References

1. D. Avis and H. ElGindy, *Triangulating simplicial point sets in space*, Proceedings of the Second Annual Symposium on Computational Geometry (New York, NY, USA), SCG '86, Association for Computing Machinery, 1986, pp. 133–141.
2. C. B. Barber, D. P. Dobkin, and H. Huhdanpaa, *The quickhull algorithm for convex hulls*, ACM Trans. Math. Softw. **22** (1996), no. 4, 469–483.
3. P. Caputo, F. Martinelli, A. Sinclair, and A. Stauffer, *Random lattice triangulations: Structure and algorithms*, Proceedings of the Forty-Fifth Annual ACM Symposium on Theory of Computing (New York, NY, USA), STOC '13, Association for Computing Machinery, 2013, pp. 615—-624.
4. B. Chazelle, *Triangulating a simple polygon in linear time*, Discrete & Computational Geometry **6** (1991), no. 3, 485–524.
5. B. Delaunay, *Sur la sphère vide. a la mémoire de georges voronoï*, Bulletin de l'Académie des Sciences de l'URSS. Classe des sciences mathématiques et na **6** (1934), 793–800.
6. M. Demirtas, L. McAllister, and A. Rios-Tascon, *Bounding the kreuzer-skarke landscape*, arXiv preprint arXiv:2008.01730 (2020).
7. P. L. Donti, B. Amos, and J. Z. Kolter, *Task-based end-to-end model learning in stochastic optimization*, Proceedings of the 31st International Conference on Neural Information Processing Systems (Red Hook, NY, USA), NIPS'17, Curran Associates Inc., 2017, pp. 5490–5500.
8. H. Edelsbrunner, *Algorithms in combinatorial geometry*, Monographs in Theoretical Computer Science. An EATCS Series, vol. 10, Springer-Verlag Berlin Heidelberg, 1987.
9. H. Edelsbrunner, F. P. Preparata, and D. B. West, *Tetrahedrizing point sets in three dimensions*, Symbolic and Algebraic Computation (Berlin, Heidelberg) (P. Gianni, ed.), Springer Berlin Heidelberg, 1989, pp. 315–331.
10. A. N. Elmachtoub and P. Grigas, *Smart "predict, then optimize"*, arXiv preprint arXiv:1710.08005 (2020).
11. F. Fang, T. Nguyen, R. Pickles, W. Lam, G. Clements, B. An, A. Singh, M. Tambe, and A. Lemieux, *Deploying PAWS: Field optimization of the protection assistant for wildlife security*, Proceedings of the Twenty-Eighth AAAI Conference on Innovative Applications of Artificial Intelligence (Palo Alto, CA, USA), IAAI'16, The AAAI Press, 2016, pp. 3966–3973.
12. J. E. Goodman, J. O'Rourke, and C. D. Tóth (eds.), *Handbook of discrete and computational geometry*, 3rd ed., Chapman and Hall/CRC, 2017.
13. C. Jordan, M. Joswig, and L. Kastner, *Parallel enumeration of triangulations*, The Electronic Journal of Combinatorics **25** (2018), no. 3, P3.6.
14. L. McShine and P. Tetali, *On the mixing time of the triangulation walk and other catalan structures*, in Pardalos et al. [18], pp. 147–160.
15. M. Molloy, B. Reed, and W. Steiger, *On the mixing rate of the triangulation walk*, in Pardalos et al. [18], pp. 179–190.
16. D. M. Mount, *Geometric intersection*, in Goodman et al. [12], pp. 1113–1134.
17. D. E. Muller and F. P. Preparata, *Finding the intersection of two convex polyhedra*, Theoretical Computer Science **7** (1978), no. 2, 217–236.
18. P. Pardalos, S. Rajasekaran, and J. Rolim (eds.), *Randomization methods in algorithm design*, DIMACS Series in Discrete Mathematics and Theoretical Computer Science, vol. 43, DIMACS-AMS, 1999.
19. J. Rambau, *TOPCOM: Triangulations of point configurations and oriented matroids*, Mathematical Software (A. M. Cohen, X.-S. Gao, and N. Takayama, eds.), World Scientific, 2002, pp. 330–340.
20. J. Ruppert, *A delaunay refinement algorithm for quality 2-dimensional mesh generation*, Journal of Algorithms **18** (1995), no. 3, 548–585.
21. R. Seidel, *A simple and fast incremental randomized algorithm for computing trapezoidal decompositions and for triangulating polygons*, Computational Geometry **1** (1991), no. 1, 51–64.

22. H. Wang, H. Xie, L. Qiu, Y. R. Yang, Y. Zhang, and A. Greenberg, *COPE: Traffic engineering in dynamic networks*, Proceedings of the 2006 Conference on Applications, Technologies, Architectures, and Protocols for Computer Communications (New York, NY, USA), SIGCOMM '06, Association for Computing Machinery, 2006, pp. 99—-110.
23. Bryan Wilder, Bistra Dilkina, and Milind Tambe, *Melding the data-decisions pipeline: Decision-focused learning for combinatorial optimization*, Proceedings of the AAAI Conference on Artificial Intelligence, vol. 33, 2019, pp. 1658–1665.

IBM Research Lab, Haifa, Israel
E-mail address: orit.davidovich@ibm.com