Computing a Loss Function to Bound the Interleaving Distance for Mapper Graphs

Erin Wolf Chambers
St. Louis University, USA

Ishika Ghosh
Michigan State University, USA

Elizabeth Munch
Michigan State University, USA

Sarah Percival
Michigan State University, USA

Bei Wang
University of Utah, USA

Abstract

Mapper graphs preserve the connected components of the inverse image function $f : \mathbb{X} \to \mathbb{R}$ over any given cover. Inspired by the interleaving distance for Reeb graphs, (Chambers et al. 2024) extends this notion of distance to discretized mapper graphs. The distance is upper-bounded using a loss function. Unlike the NP-hard interleaving distance computation for Reeb graphs, the algorithm of the loss function has polynomial complexity. In this paper, we implement the categorical framework of mapper graphs and compute the loss function to bound the interleaving distance.

2012 ACM Subject Classification Mathematics of computing → Algebraic topology; Theory of computation → Computational geometry

Keywords and phrases Mapper graphs, geometric graphs, interleaving distance

Funding This research was partially supported by a grant from the Department of Energy (DOE) DE-SC0021015 and and grants from the National Science Foundation (NSF) DMS-2301361, CCF-1907591, CCF-2106578, CCF-2142713, CCF-2106578, CCF-2106672, and CCF-1907612.

1 Introduction

Developing efficient and computable metrics to compare graphical representations of data is crucial for data analysis. Computationally, topological descriptors of discretized underlying space are essential, as such input data is common. Often, these datasets are equipped with a function $f : \mathbb{X} \to \mathbb{R}$. Here, we direct our attention to mapper graphs; see Fig. 1 for an example. These graphical data structures keep track of the relationship between connected components of the inverse image of elements of a particular choice of cover. Such mapper graphs can be compared using a variant of the interleaving distance [2, 3, 7]. However, the computation of the interleaving distance is NP-hard in general [1, 2]. Formally, we encode our mapper inputs as functors (see e.g. [5, 6]) of the form $F : \mathbf{Open}(K) \to \mathbf{Set}$ for a space K encoding the cover information. In [3], K is defined for a chosen $\delta > 0$ as a cubical complex over the bounding interval $[-B, B]$ with diameter δ.

The result is that the open sets of K are intervals of the form $(i\delta, j\delta)$ for $i, j \in \{-L, \cdots, L\}$ where $L \cdot \delta = B$. In [3], a 1-thickening on these intervals is introduced, where the thickening

---

This is an abstract of a presentation given at CG:YRF 2024. It has been made public for the benefit of the community and should be considered a preprint rather than a formally reviewed paper. Thus, this work is expected to appear in a conference with formal proceedings and/or in a journal.

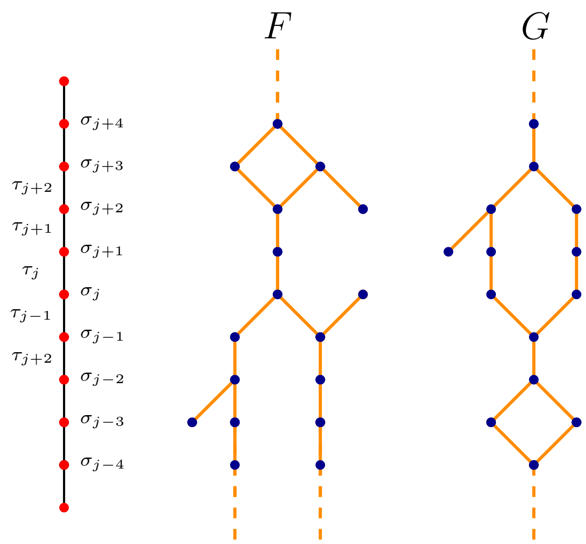

Figure 1 Two input mapper graphs F and G. Discretization on the left.

$(i\delta, j\delta)^n$ is the interval $((i-n)\delta, (j+n)\delta)$. This can be pre-composed with the functor F to result in an n-thickened functor $F^n : \mathbf{Open}(K) \to \mathbf{Set}$ given by $F^n(U) = F(U^n)$. This in turn defines an interleaving distance d_I [4, 7] as follows. An interleaving is a pair of natural transformations $\varphi : F \Rightarrow G^n$ and $\psi : G \Rightarrow F^n$ which must satisfy certain commutativity properties. We will denote the four diagrams required to commute by $\boxbslash_{\varphi}(U,V)$, $\boxslash_{\psi}(U,V)$, $\nabla_{\varphi,\psi}(U)$, and $\triangle_{\varphi,\psi}(U)$; see [3] for details. Then the interleaving distance is the smallest n for which such interleaving exists, otherwise the distance is set to infinity [3].

Chambers et al. [3] defined a loss function for structures that have the format of a natural transformation without being provided the commutativity assumptions. They call a collection of maps $\varphi_U : F(U) \to G^n(U)$ and $\psi_U : G(U) \to F^n(U)$ an n-assignment; noting that if φ and ψ satisfy the commutativity properties it would constitute an interleaving. Then the loss function $L_B(\varphi, \psi)$ is defined in a way that results in finding the minimum k such that φ and ψ can be turned into an $(n+k)$-interleaving. When storing the functor information in a graph, the result is that one must check whether two representatives under a particular map are in the same connected component of a slice of the graph; see Fig. 2 for these diagrams.

▶ **Theorem 1** (Chambers et al. [3]). *For an n-assignment, $\varphi : F \Rightarrow G^n$ and $\psi : G \Rightarrow F^n$,*

$$d_I(F,G) \le n + L_B(\varphi, \psi).$$

In this work, we provide additional details of the algorithmic setup for computing this loss function on graph representations of the functor data. See [3] for additional details.

2 Algorithm and Computation

We set up the mapper graph data structure first. Our input is a pair of functors $F, G : \mathbf{Open}(K) \to \mathbf{Set}$ and an n-assignment φ, ψ. Here K (i.e., the discretization of $[-L\delta, L\delta] \subset \mathbb{R}$) consists of vertices $\sigma_i = i\delta$ for $-L \le i \le L$. Additionally, we have edges $\tau_j = (\sigma_j, \sigma_{j+1})$ for $-L \le j < L$. We write a basis for $\mathbf{Open}(K)$ by defining the collection of intervals $U_{\sigma_i} = ((i-1)\delta, (i+1)\delta)$ and $U_{\tau_i} = (i\delta, (i+1)\delta)$ for all i. The vertex set of the graph representation

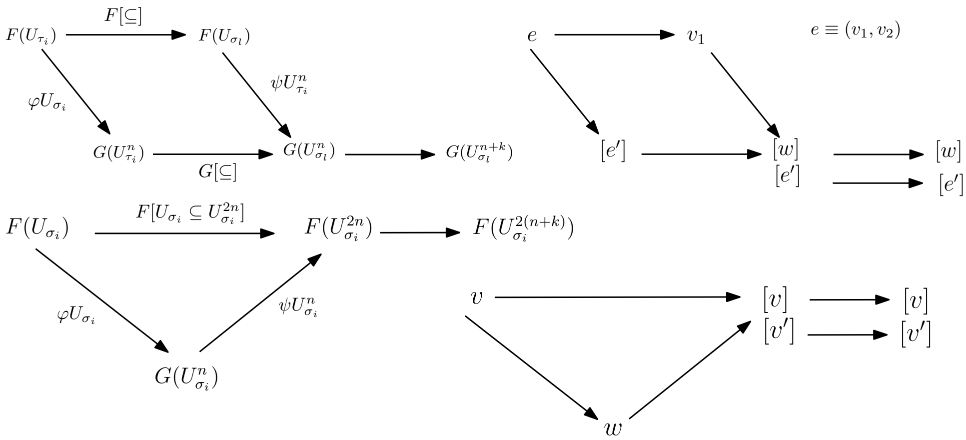

Figure 2 Example diagrams that must be checked for commutativity to determine the loss function. At right are the representatives from the data structures which must be checked for being in the same connected component of the same slice of the representative graph. See [3] for details.

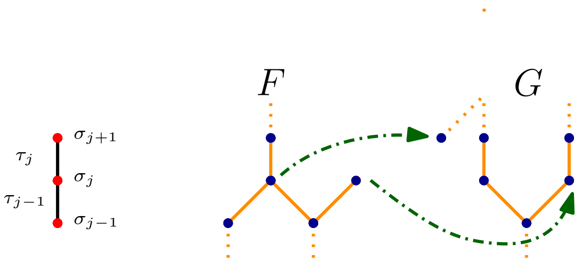

Figure 3 Map the vertices of F at height σ_j with $n = 1$. The G slice contains vertices with heights $\sigma_{j-1}, \sigma_j, \sigma_{j+1}$. Vertices in same connected component in F end up in different components.

of the functor is given by $V = \coprod_i F(U_{\sigma_i})$. The edge set is given by $E = \coprod_i F(U_{\tau_i})$ and are attached to the vertices using the functor. See Fig. 1 for an example and [3] for details.

We implement this structure in Python using the *NetworkX* package. We build a custom *MapperGraph* class to encode the functors F and G which constructed as graphs (V_F, E_F) and (V_G, E_G). We also store the height information for each vertex as node attributes. The *MapperGraph* class also contains some useful functions for visualization and retrieval of data.

The n-assignments φ, ψ are encoded as vertex and edge maps; see [3] for details. To define φ (ψ is similar), for height i of each vertex (or height of lower vertex for an edge) in F, we only look at the n-thickening of G at that height. In other words, we define a *slice* of the functor which only includes the vertices and edges within height $[i-n, i+n]$. Each element in F gets randomly paired with an element in the corresponding slice of G. The resulting map is stored as a dictionary, with (object, image) as key-value pairs. Figure 3 and 4 illustrate some examples.

Given two mapper graphs F, G, and assignments φ, ψ, we compute the loss separately for $L^{U_\tau, U_\sigma}_{\diagdown\!\!\!\square}$ (or, $L^{U_\tau, U_\sigma}_{\diagup\!\!\!\square}$) and $L^{U_\sigma}_{\bigtriangledown}$ (or, $L^{U_\sigma}_{\triangle}$). Fix a k for each step with binary search on $[0, \cdots, 2L]$, where $[-L, L]$ is the bounding box of the functors. For each k, we verify if $L_B(\varphi, \psi) \leq k$. We travel across the diagrams and note if the resulting edges or vertices are in same connected component. We show an example of two types of diagrams in Fig. 2. Loss is the smallest k for which all diagrams commute. If no such k exits, then the loss is deemed infinite.

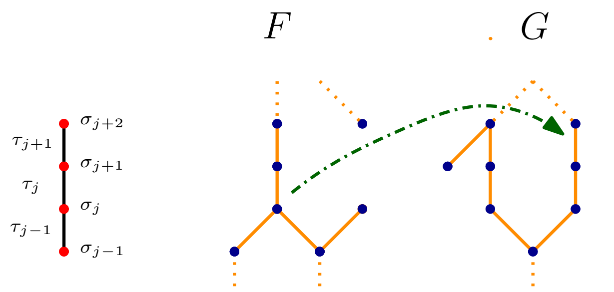

Figure 4 Map edges of F with lower vertex-height σ_j with $n = 1$. The G slice contains edges with lower vertex-heights $\sigma_{j-1}, \sigma_j, \sigma_{j+1}$. Notice how slicing varies from vertex mapping.

3 Discussion

Now that we have set up the data structure to encode 1-dimensional mapper graphs, we are focusing on optimizing the loss function. Given two mapper graph functors F, G and an initial n-assignment φ, ψ, our goal is to perturb the assignments cleverly such that the loss function is minimized. In future work, we will deploy the Metropolis–Hastings algorithm over the space of n-assignments to optimize and improve the bound. Further, our goal is to extend this implementation to higher dimensional mapper graphs; i.e. when the input data is of the form $f : \mathbb{X} \to \mathbb{R}^d$.

References

1 Håvard Bakke Bjerkevik and Magnus Bakke Botnan. Computational Complexity of the Interleaving Distance. In Bettina Speckmann and Csaba D. Tóth, editors, *34th International Symposium on Computational Geometry (SoCG 2018)*, volume 99 of *Leibniz International Proceedings in Informatics (LIPIcs)*, pages 13:1–13:15, Dagstuhl, Germany, 2018. Schloss Dagstuhl – Leibniz-Zentrum für Informatik. URL: `https://drops-dev.dagstuhl.de/entities/document/10.4230/LIPIcs.SoCG.2018.13`, `doi:10.4230/LIPIcs.SoCG.2018.13`.

2 Peter Bubenik, Vin De Silva, and Jonathan Scott. Metrics for generalized persistence modules. *Foundations of Computational Mathematics*, 15:1501–1531, 2015.

3 Erin W Chambers, Elizabeth Munch, Sarah Percival, and Bei Wang. Bounding the interleaving distance for geometric graphs with a loss function. *arXiv preprint arXiv:2307.15130*, 2023.

4 Frédéric Chazal, David Cohen-Steiner, Marc Glisse, Leonidas J Guibas, and Steve Y Oudot. Proximity of persistence modules and their diagrams. In *Proceedings of the twenty-fifth annual symposium on Computational geometry*, pages 237–246, 2009.

5 Justin Curry. *Sheaves, Cosheaves and Applications*. PhD thesis, University of Pennsylvania, 2014. `arXiv:1303.3255`.

6 Emily Riehl. *Category theory in context*. Courier Dover Publications, 2017.

7 V De Silva, E Munch, and A Stefanou. THEORY OF INTERLEAVINGS ON CATEGORIES WITH A FLOW. *Theory and Applications of Categories*, 33(21):583–607, 2018. URL: `http://www.tac.mta.ca/tac/volumes/33/21/33-21.pdf`, `arXiv:1706.04095`.