Topological Mapper for 3D Volumetric Images*

Daniel H. Chitwood, Mitchell Eithun, Elizabeth Munch, and Tim Ophelders

Michigan State University, East Lansing, MI, USA
{chitwoo9,eithunmi,muncheli,ophelder}@msu.edu

Abstract. Mapper is a topological construction similar to a Reeb graph, and is used to summarize the shape of a dataset as a (generalized) graph. Formally, mapper can be constructed for any connected space and algorithms have been developed to compute mapper for point clouds and 2D images. In this paper, we extend mapper to 3D volumetric images. We use our algorithm to compute mapper for scans of barley generated using computed tomography. We demonstrate the flexibility of the construction by highlighting different aspects of the morphology through different choices of starting parameters. Applying mapper to this type of data provides an integrated means of visualization, segmentation and clustering, and can thus be used to study the topology of any 3D object.

Keywords: Topological Mapper · Image Processing · Computed Tomography · Topological Data Analysis

1 Introduction

Mapper is a construction from topological data analysis [3,10,15] that generates an abstract simplicial complex from data. Traditionally, mapper bins a point cloud based on a filter function, clusters the points inside each bin and builds a simplicial complex based on how the clusters of different bins overlap [27]. Statistically, mapper is an optimal estimator of a Reeb graph [4,16], which summarizes connected components of functions on manifolds [22,26]. For surfaces with genus 0, a Reeb graph reduces to a contour tree [7].

While mapper was originally defined for point-clouds, the idea can be applied to any dataset in a metric space. Some applications of mapper include visualization, clustering and classification. For example, mapper has identified a new subtype of breast cancer [18]. Mapper has also been used to mine data related to spinal chord and traumatic brain injuries [19]. The more general concept of a Reeb graph has been used for phenotyping root architectures [14] and comparing cortical surfaces in brain scans [25]. Mapper graphs can be constructed from Reeb graphs, which can be constructed in $O(n \log n)$ time [7].

* This project was supported by the USDA National Institute of Food and Agriculture, and by Michigan State University AgBioResearch. The work of EM was supported in part by NSF grants DMS-1800446 and CMMI-1800466.

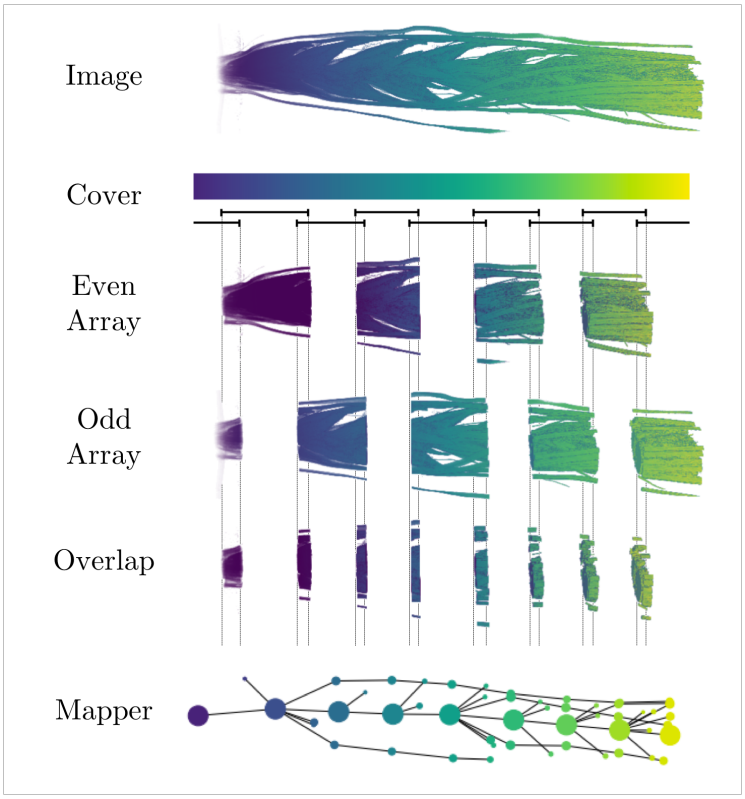


Fig. 1. Mapper applied to a 3D volumetric image, filtered by height. The preimages of cover elements along the range are clustered to form vertices in the mapper graph. Edges are defined by overlap between clusters.

Recently, a version of mapper was developed for 2D images [24]. We extend this algorithm to voxel-based images in 3D, called volumetric images. In particular we clarify how the domain of an image is connected after imposing a lattice graph, which naturally defines clusters as connected components. We also explain different parameters used to generate an open interval cover. To our knowledge, this is the first time mapper has been applied to volumetric images.

As an example we apply mapper to a volumetric image obtained through X-ray computed tomography (CT) on a barley spike. Different choices of filter functions and cover elements reveal different aspects of the morphology and structure of barley.

2 Background

Mapper separates data using a filter function, performs clustering, and outputs a *simplicial complex* describing how clusters overlap. A simplicial complex can

be thought of as a generalization of a graph; that is, a combinatorial object with vertices and edges, but also triangles, tetrahedra, and higher dimensional analogues. However, the reader not familiar with this concept can internally replace "simplicial complex" with "graph" for this paper as we will immediately restrict our constructions of interest so that all simplicial complexes are 1-dimensional.

The general mapper construction uses the concept of the nerve of a cover [13, §3.3]. Given an open cover of a topological space (a collection of open sets whose union contains the space), the nerve is a skeleton that summarizes the intersections of the cover elements. For ease of notation throughout, define $[N] := \{1, 2, \ldots, N\}$.

Definition 1. *Let the collection $\mathcal{U} = \{U_\alpha\}_{\alpha=1}^N$ be an open cover of a topological space $\mathbb{X}$. The **nerve** of $\mathcal{U}$, written $Nrv(\mathcal{U})$, is the simplicial complex with vertex set $[N]$, for which $\{\alpha_1, \alpha_2, \ldots, \alpha_k\} \subseteq [N]$ spans a k-simplex in $Nrv(\mathcal{U})$ if and only if $\bigcap_{i=1}^k U_{\alpha_i} \neq \emptyset$.*

To define mapper, consider a real-valued function f on some topological space (this is commonly called a filter function) and choose an open cover of the range of f. Mapper is the nerve of the components of the preimages (under f) of the cover elements. Note that if there are no triple intersections between these preimages, then all of the simplices in the nerve will be of degree 0 or 1, i.e. the nerve is a graph.

Definition 2. *Let $\mathbb{X}$ be a connected topological space and let $f\colon \mathbb{X} \to \mathbb{R}$ be a continuous function. Choose an open interval cover $\mathcal{U} = \{U_\alpha\}_{\alpha=1}^N$ of $f(\mathbb{X})$ such that $U_\alpha \cap U_{\alpha'} \cap U_{\alpha''} = \emptyset$ for all distinct $U_\alpha, U_{\alpha'}, U_{\alpha''} \in \mathcal{U}$. Let $\mathcal{V}$ be the cover of the domain of f whose elements are the connected components of $f^{-1}(U)$ for all $U \in \mathcal{U}$. The **mapper graph** is*

$$\mathcal{M}(\mathbb{X}, f, \mathcal{U}) := Nrv(\mathcal{V}).$$

One important difference to note between our definition and the original definition from [27] is that we assume our input data is a continuous function on a topological space, while the original definition was given for point cloud data (or more generally, finite metric space data). This actually simplifies the construction as it means that no choice of clustering method is required; this will be further discussed in Sec. 3.1.

3 Topological Mapper for 3D Images

Imaging methods such as magnetic resonance imaging (MRI) and CT generate 3D volumetric images. Volumetric images are 3D arrays of intensity values, which roughly correspond to the density of the object in a particular spot. Each cubical cell in the image is called a *voxel*, which is analogous to the concept of a *pixel* in a 2D image. See Fig. 1 for an example of the full construction.

Applying mapper to volumetric images follows the basic construction in Section 2, except that the structure of the domain warrants a careful approach to

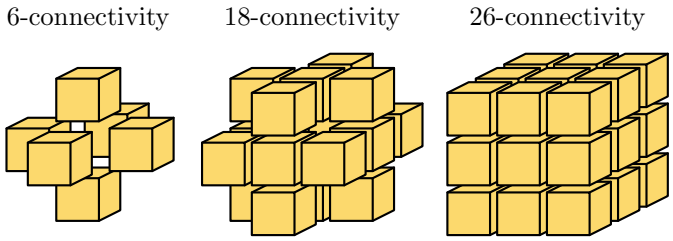


Fig. 2. Standard structuring elements for 3D lattices.

connectivity. Denote an $m \times n \times p$ cubical lattice by $L_{m,n,p} := [m] \times [n] \times [p]$. An $m \times n \times p$ volumetric image $\mathcal{P}$ is a function $\mathcal{P}\colon L_{m,n,p} \to \mathbb{R}$ and a set of input coordinates (i, j, k) is called a *voxel coordinate*. Elements in the range of $\mathcal{P}$ are *intensity values* given by the method used to create the scan (e.g. CT or MRI).

3.1 Structuring Element

There are various ways in which one can define connectivity between the voxel coordinates. We capture such a model of connectivity using a graph on $L_{m,n,p}$ that specifies the pairs of points that are adjacent. In the models of connectivity we consider, the edges of the graphs are based on the relative coordinates between points. That is, there is an edge between two points if the difference between those points lies in a given set D. Following the usual convention in 2D [17], we refer to D as the *structuring element*, and define the *voxel connectivity graph* $G = (V, E)$ by

$$\begin{aligned} V &:= L_{m,n,p}, \\ E &:= \{(u, v) \in V \times V \mid u - v \in D\}. \end{aligned}$$

The L_p-norm of a point $x = (x_1, \ldots, x_k) \in \mathbb{R}^k$ is given by $\|x\|_p = (\sum_{i=1}^{k} |x_i|^p)^{1/p}$. Standard structuring elements for a k-dimensional lattice include the sets $D = \{x \mid 0 < \|x\|_2 \leq \sqrt{r}\}$ for $r \in [k]$. Further natural structuring elements are the sets $D = \{x \mid \|x\|_1 \leq r \text{ and } \|x\|_\infty = 1\}$ of points simultaneously at distance 1 in the L_∞-norm and at distance at most r from the origin in the L_1-norm for $r \in [k]$. For $k \leq 3$, these two models of connectivity are identical for each r. For $r = 1$, 2, and 3, these structuring elements are illustrated in Fig. 2, and are called 6-, 18-, and 26-connectivity, respectively.

In practice, the vertex set V is chosen to be a subset of $L_{m,n,p}$ that only includes the foreground of the image. This is commonly done by thresholding the image (setting intensity values in some range equal to 0). In this paper we use 26-connectivity ("full connectivity") on interior voxel coordinates.

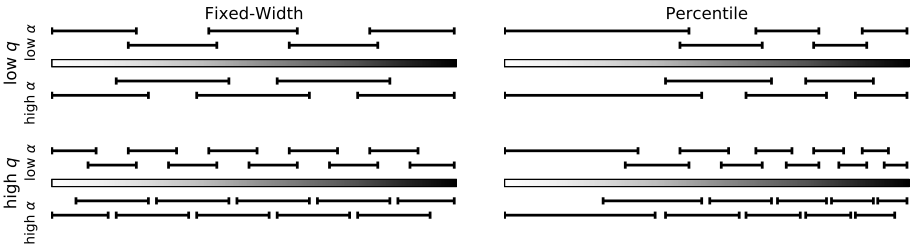


Fig. 3. Examples of fixed-width and percentile intervals over the range of the filter $f\colon [0,1] \to [0,1]$ given by $f(x) = \sqrt{x}$. We consider three parameters for cover elements: the type of scheme (evenly-spaced or percentile), the number of cover elements $(q+1)$ and the amount of overlap, or gain (α).

3.2 Cover Elements

We suggest two heuristics for choosing intervals $\{(a_0, b_0), (a_1, b_1), \ldots, (a_q, b_q)\}$ to cover the range of a filter function $f\colon \mathbb{X} \to \mathbb{R}$. See Fig. 4 for examples of these constructions.

First we define evenly-spaced intervals over the range. Suppose that $f(\mathbb{X}) \subseteq (a, b)$ and let $a_0 := a$ and $b_q := b$. Let $r = (b-a)/q$ be the interval length and choose $\alpha \in (0, 1]$ to be the proportion of overlap between cover elements. Now for $1 \leq i \leq q-1$, let

$$a_i = a + (i + \alpha)r,$$
$$b_i = a + (i + 1 - \alpha)r.$$

This rule generates $q+1$ overlapping intervals. The parameter r is called the resolution and controls the diameter of the elements; the gain α captures the proportion of overlap [4].

The second scheme is based on the underlying distribution of intensity values. Specifically, define $\mathrm{Perc}(x)\colon [0,1] \to \mathbb{R}$ so that $\mathrm{Perc}(x)$ is the $(100x)^{\text{th}}$ percentile of the range of f. Let $a_0 = \mathrm{Perc}(0)$ and $b_q = \mathrm{Perc}(1)$ and for $1 \leq i \leq q-1$, let

$$a_i = \mathrm{Perc}(i/q) - \alpha(\mathrm{Perc}(i/q) - \mathrm{Perc}((i-1)/q)),$$
$$b_i = \mathrm{Perc}((i+1)/q) + \alpha(\mathrm{Perc}((i+1)/q) - \mathrm{Perc}(i/q)).$$

The cover intervals defined here correspond to subsets of the same size in the underlying dataset.

3.3 Filtering

A filter function used to construct the mapper graph is a choice of real valued function on the vertices. The most natural filter function to use for a volumetric image $\mathcal{P}$ is the input data itself, i.e $f(i,j,k) = \mathcal{P}(i,j,k)$. Filters may also be constructed by viewing the volumetric image as a collection of foreground voxel coordinates. For example, the *z-filter* $f_z\colon V(G) \to \mathbb{R}$ is defined by $f_z((i,j,k)) = k$.

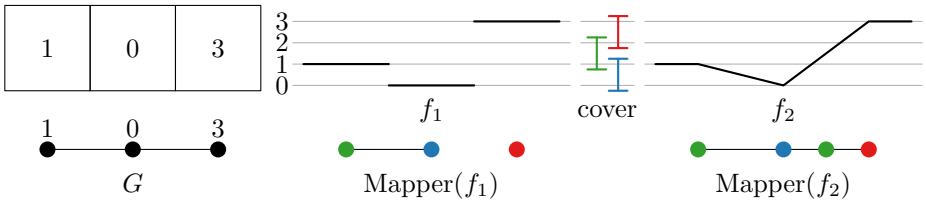


Fig. 4. An example of a 2D image with three voxels. The resulting mapper is disconnected if the function is piecewise-constant, but connected if the function value is interpolated along edges.

Similar filters can be constructed in the x- and y-directions and, more generally, from sweeping a plane with an arbitrary direction through the finite lattice $L_{m,n,p}$.

There are various ways we can extend a filter function to the voxel connectivity graph. A natural way to construct a filter on a 3D volumetric image is to extend a function $f\colon L_{m,n,p} \to \mathbb{R}$ defined on the vertices $V(G)$ to the points interior to edges of G. This can be done through linear interpolation between the function values at the end points of the edge, but also by assigning the function value of the closest end point of that edge, resulting in a piecewise-constant function. Figure 4 illustrates how these two choices may result in a different mapper graph. However, if adjacent voxels always lie in a common element of the cover of the range of f, the resulting mapper is the same.

For our algorithm, we choose the piecewise-constant method of extending the function to edges. In the context of CT images this choice makes sense because the intensity values arise from the X-ray densities of the scanned materials. The overlap between cover elements is often chosen to capture adjacency between certain materials. For instance, we might be interested when a material of intensity 0 is adjacent to a material of intensity 1. However, if we use linear interpolation, adjacent materials of intensities 0 and 3 would (falsely) appear to be separated by a material of intensity 1.

In practice the filtering step takes advantage of the fact that no three cover elements intersect. Consider a filter function f with the cover intervals

$$\{(a_0, b_0), (a_1, b_1), \ldots, (a_q, b_q)\}.$$

Assume that $a_0 \leq a_1 \leq \cdots \leq a_q$, as in our construction of fixed-width or percentile intervals. By avoiding triple intersections between cover intervals we can assume that $f^{-1}((a_\ell, b_\ell))$ and $f^{-1}((a_{\ell+2}, b_{\ell+2}))$ are disjoint. Furthermore, the preimages of intervals with even index are disjoint; similarly the preimage of intervals with odd index are disjoint. Following [24], we exploit this fact to construct two 3D label arrays to filter the data: L_0 and L_1, called the even and odd arrays respectively. For $b \in \{0, 1\}$, define

$$L_b(i,j,k) = \begin{cases} 2\ell + b & \text{if } (i,j,k) \in f^{-1}((a_{2\ell+b}, b_{2\ell+b})) \text{ for some } \ell \in \mathbb{Z} \\ 0 & \text{otherwise} \end{cases}.$$

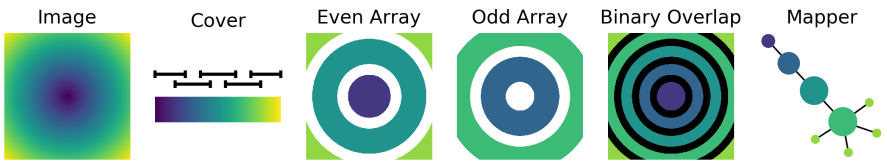


Fig. 5. Mapper on a 2D circular gradient image. The even (odd) array contains the voxels corresponding to cover elements with even (odd) indices. Each region (black) in the overlap between the two arrays suggests an edge in the mapper graph.

In other words, a non-zero value in L_0 or L_1 at voxel coordinate (i, j, k) is the index of the cover interval that contains $\mathcal{P}(i, j, k)$. Examples of even and odd arrays are shown in Fig. 5.

3.4 Clustering

In the point-cloud version of mapper, a clustering algorithm (e.g. k-means) is required to separate points within the pullback of each cover element. In this version for 3D volumetric images, clusters are defined by connected components in the voxel connectivity graph. These components are dependent on the chosen structuring element, as described in Sec. 3.1. Further, they are sensitive to the choice of filter function interpolation as described in Sec. 3.3.

In practice, label the disjoint regions in L_0 and L_1 to generate arrays L'_0 and L'_1. These arrays simply choose a labeling for each connected component in L_i and give each voxel in the connected component the same entry. The connected components in these images are what will be used to determine the vertices of the mapper graph. Note that the preimage of a cover element may have several connected components.

3.5 Mapper Graph Construction

To construct the mapper graph, define an overlap array $\mathcal{O}$ as the logical `AND` of L'_0 and L'_1. The disjoint regions in this image represent the overlap between clusters. To identify the edges in the mapper graph let (i, j, k) be a voxel coordinate in a connected component of $\mathcal{O}$. The mapper graph has the edge $(L'_0(i, j, k), L'_1(i, j, k))$.

3.6 Implementation

While there is available open and closed-source code for point cloud input data, to the best of our knowledge, no previous code is available for applying mapper to 3D images. Open-source software to apply mapper to point clouds includes an implementation in R [21], Kepler Mapper in `Scikit-TDA` for Python 3 [2], and Python Mapper, which has a GUI for Python 2 [1]. The private company

Ayasdi also offers closed-source software to generate mapper graphs for point clouds, but not volumetric images.

Our implementation for 3D images is done in Python 3, using `numpy` [20] and `skimage` [28]. It is open-source and available on Github [8]. This algorithm is based on extending the ideas of [24] which is restricted to 2D images.

4 Example: The Shape of Barley

We apply our 3D mapper algorithm to a CT reconstruction of barley (*Hordeum vulgare* L.). The specimen, a reconstructed spike, is shown in Figure 1. A spike is an inflorescence bearing sessile flowers. Spikelets consist of single flowers and are attached to the rachis (the backbone of the inflorescence) in groups of three at each node, alternately arranged. Subtending each spikelet are two glumes (bracts), and enclosing the seed of each flower are upper (palea) and lower (lemma) bracts. From each lemma extends a prominent awn (a bristle-like appendage) [23]. The seed of each spikelet has a conspicuously high X-ray density. The surrounding organs of each seed, the awns, and rachis are less X-ray dense. We use this example to show how different choices of parameters in the mapper construction highlight different aspects of the data in question. An interactive visualization of mapper applied to this barley scan is available at `egr.msu.edu/~eithunmi/js_mapper`.

4.1 Parameters

Our construction of mapper for images requires three main parameters: a choice of structuring element, a filter function and an open cover of its range. In this example we adopt 26-connectivity and give examples using both the intensity filter and the z-filter. For each filter function, we consider different numbers of evenly-spaced intervals and percentile intervals with different levels of overlap.

4.2 z-filter

The output of mapper with the z-filter is shown in Fig. 6. Following [27], the graphs are drawn to reflect the underlying data. Vertices are colored according to the average value of the clusters they represent and their size is proportional to the square root of the size of each cluster. Additionally, vertices are plotted at the centroids of their underlying clusters in a 2D projection of the volumetric image, which suggests the overall structure of the spike. The bristle-like awns extend out from the nodes of the spike, and the individual spikelets from which the awns arise are segmented along the central axis representing the rachis. The correspondence of the mapper graph to the architecture of the barley spike is because of the choice of direction of the z-axis along the rachis, which is sensitive to the separation of the spikelets and awns from the main body of the inflorescence.

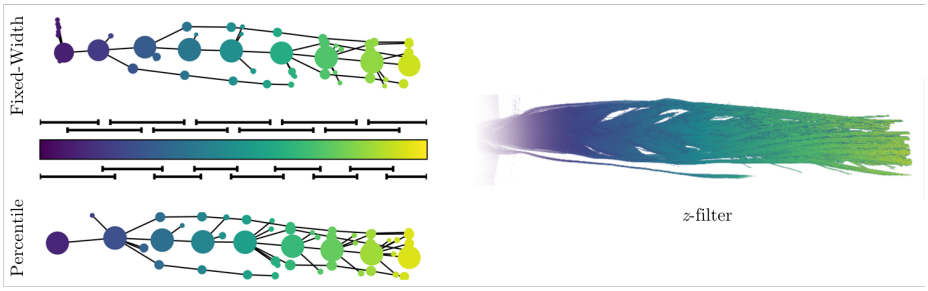


Fig. 6. Mapper graphs generated using the z-filter. The color bar shows the cover elements on the range of the filter. The nodes are drawn in line with the average function value of vertices represented.

4.3 Intensity Filter

The output of mapper with the intensity filter is shown in Figure 7. Like the z-filter, the intensity filter summarizes the anatomical features of the barley spike, but as defined by X-ray density, yielding a different graph. Clusters corresponding to X-ray dense seeds all connects to a central node, representing the rachis. Similarly, there are separate hubs connected to numerous edges corresponding to awns and anatomical features surrounding the seed.

5 Discussion

New data analysis tools from the field of applied topology have provided previously unavailable insights on the shape of data. Applying mapper to 3D volumetric images has implications for studying the shape and structure of 3D images. In the future we plan to study other intricate plant structures including maize and citrus using X-ray CT. Extending this algorithm to 4D images or using time-varying mapper graphs could also be used to track morphological changes over time or across species. Further structural information (at the expense of computational cost) could be also constructed by removing the restriction on triple intersections and producing a higher-dimensional simplicial complex. Recently proposed parallelization schemes might make this more feasible [12,11]. Multiscale mapper approaches may also be useful for capturing structural details at different scales [6].

Another important area for future work is parameter selection, which may reduce noise in resultant mapper graphs. Recently a bootstrap method was proposed to automatically tune mapper parameters by estimating the distance between mapper graph and the underlying Reeb graph [4]. Along with the rise and the gain parameters used to generate a fixed-width interval cover, a δ parameter is introduced which corresponds to our choice of structuring element. To our knowledge no process has been proposed to choose parameters for a percentile-based interval scheme. Parameters could also be chosen based on some desirable outcome, such as the connectivity or diameter of the output graph.

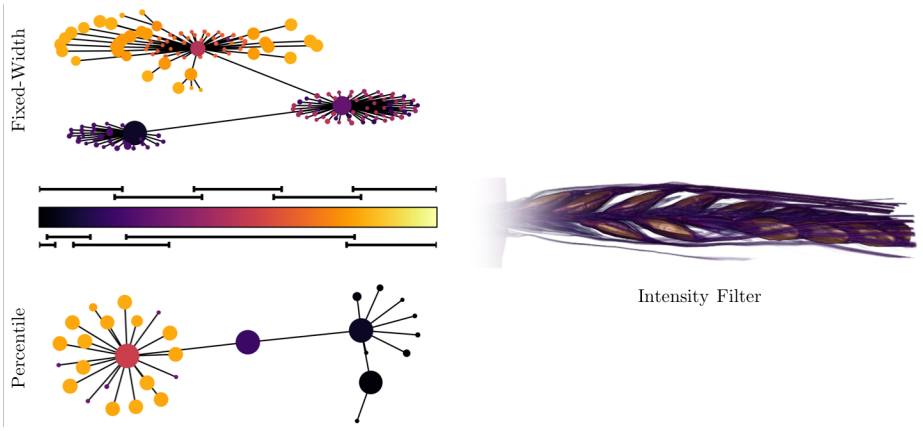


Fig. 7. Mapper graphs for the barley shown in Fig. 6 generated using the intensity filter with two different choices of cover. The locations of the vertices are drawn with the Fruchterman-Reingold spring layout [9].

Concerning the input data, while the scheme proposed here solves a key problem related to connectivity by imposing a graph structure on the data, the clustering scheme is highly susceptible to noise. For example, if a connected graph component in a preimage of a cover element has connectivity 1, the removal of one edge would disconnect the graph. A way to combat this might be to use graph connectivity measures to test each cluster. If a cluster is not connected by sufficiently many edges it can be cut to form multiple clusters.

Acknowledgments

The authors thank Jacob Landis and Daniel Koenig for providing the barley spike and X-ray Computed Tomography data. The data set is available on the figshare repository [5].

References

1. Python mapper, http://danifold.net/mapper/
2. Sckit-tda, https://github.com/scikit-tda/scikit-tda
3. Carlsson, G.: Topological pattern recognition for point cloud data. Acta Numerica **23**, 289–368 (5 2014). https://doi.org/10.1017/S0962492914000051, http://journals.cambridge.org/article_S0962492914000051, survey
4. Carrière, M., Michel, B., Oudot, S.: Statistical analysis and parameter selection for mapper. arXiv:1706.00204 (2017)
5. Chitwood, D.H., Eithun, M., Koenig, D., Landis, J., Munch, E., Ophelders, T.: CT scan of barley (Hordeum vulgare L.) (2019). https://doi.org/10.6084/m9.figshare.7590833

6. Dey, T., Mmoli, F., Wang, Y.: Multiscale Mapper: Topological Summarization via Codomain Covers. In: Proceedings of the Twenty-Seventh Annual ACM-SIAM Symposium on Discrete Algorithms, pp. 997–1013. Proceedings, Society for Industrial and Applied Mathematics (Dec 2015). https://doi.org/10.1137/1.9781611974331.ch71, https://epubs.siam.org/doi/10.1137/1.9781611974331.ch71
7. Doraiswamy, H., Natarajan, V.: Computing reeb graphs as a union of contour trees. IEEE Transactions on Visualization and Computer Graphics **2**, 249–262 (2013)
8. Eithun, M.: eithun/3D-mapper: Topological mapper for 3D volumetric images (2019). https://doi.org/10.5281/zenodo.2602059
9. Fruchterman, T.M., Reingold, E.M.: Graph drawing by force-directed placement. Software: Practice and experience **21**(11), 1129–1164 (1991)
10. Ghrist, R.: Elementary Applied Topology. CreateSpace Independent Publishing Platform (2014)
11. Hajij, M., Assiri, B., Rosen, P.: Distributed Mapper. arXiv:1712.03660 [cs, stat] (Dec 2017), http://arxiv.org/abs/1712.03660, arXiv: 1712.03660
12. Hajij, M., Rosen, P.: An Efficient Data Retrieval Parallel Reeb Graph Algorithm. arXiv:1810.08310 [cs] (Oct 2018), http://arxiv.org/abs/1810.08310, arXiv: 1810.08310
13. Hatcher, A.: Algebraic Topology. Cambridge University Press (2001)
14. Janusch, I.: Reeb Graph Based Image Representation for Phenotyping of Plants. Master's thesis, Vienna University of Technology (2014)
15. Munch, E.: A user's guide to topological data analysis. Journal of Learning Analytics **4**(2) (2017). https://doi.org/10.18608/jla.2017.42.6
16. Munch, E., Wang, B.: Convergence between categorical representations of Reeb space and Mapper. In: Fekete, S., Lubiw, A. (eds.) 32nd International Symposium on Computational Geometry (SoCG 2016). Leibniz International Proceedings in Informatics (LIPIcs), vol. 51, pp. 53:1–53:16. Schloss Dagstuhl–Leibniz-Zentrum fuer Informatik, Dagstuhl, Germany (2016). https://doi.org/http://dx.doi.org/10.4230/LIPIcs.SoCG.2016.53, http://drops.dagstuhl.de/opus/volltexte/2016/5945
17. Najman, L., Cousty, J.: A graph-based mathematical morphology reader. Pattern Recognition Letters **47**, 3–17 (Oct 2014). https://doi.org/10.1016/j.patrec.2014.05.007, https://linkinghub.elsevier.com/retrieve/pii/S0167865514001548
18. Nicolau, M., Levine, A.J., Carlsson, G.: Topology based data analysis identifies a subgroup of breast cancers with a unique mutational profile and excellent survival. Proceedings of the National Academy of Sciences p. 201102826 (Apr 2011). https://doi.org/10.1073/pnas.1102826108, https://www.pnas.org/content/early/2011/04/07/1102826108
19. Nielson, J.L., Paquette, J., Liu, A.W., Guandique, C.F., Tovar, C.A., Inoue, T., Irvine, K.A., Gensel, J.C., Kloke, J., Petrossian, T.C., Lum, P.Y., Carlsson, G.E., Manley, G.T., Young, W., Beattie, M.S., Bresnahan, J.C., Ferguson, A.R.: Topological data analysis for discovery in preclinical spinal cord injury and traumatic brain injury. Nature Communications **6** (Oct 2015). https://doi.org/10.1038/ncomms9581, https://www.ncbi.nlm.nih.gov/pmc/articles/PMC4634208/
20. Oliphant, T.E.: Guide to numpy (2006–)
21. Piekenbrock, M., Doran, D., Kram, R.: Efficient multi-scale simplicial complex generation for mapper (2018), https://peekxc.github.io/resources/indexed_mapper.pdf

22. Reeb, G.: Sur les points singuliers d'une forme de pfaff complèment intégrable ou d'une fonction numérique. Comptes Rendus de L'Académie ses Séances **222**, 847–849 (1946)
23. Reid, D.A.: Morphology and anatomy of the barley plant. Barley (barley), 73–101 (1985)
24. Robles., A., Hajij., M., Rosen., P.: The shape of an image - a study of mapper on images. In: Proceedings of the 13th International Joint Conference on Computer Vision, Imaging and Computer Graphics Theory and Applications - Volume 4: VISAPP,. pp. 339–347. INSTICC, SciTePress (2018). https://doi.org/10.5220/0006574803390347
25. Shi, Y., Li, J., Toga, A.W.: Persistent reeb graph matching for fast brain search. In: Machine Learning in Medical Imaging, pp. 306–313. Springer International Publishing (2014). https://doi.org/10.1007/978-3-319-10581-9_38
26. de Silva, V., Munch, E., Patel, A.: Categorified Reeb Graphs. Discrete & Computational Geometry **55**(4), 854–906 (Jun 2016). https://doi.org/10.1007/s00454-016-9763-9, https://doi.org/10.1007/s00454-016-9763-9
27. Singh, G., Memoli, F., Carlsson, G.: Topological Methods for the Analysis of High Dimensional Data Sets and 3D Object Recognition. In: Botsch, M., Pajarola, R., Chen, B., Zwicker, M. (eds.) Eurographics Symposium on Point-Based Graphics. The Eurographics Association (2007). https://doi.org/10.2312/SPBG/SPBG07/091-100
28. van der Walt, S., Schönberger, J.L., Nunez-Iglesias, J., Boulogne, F., Warner, J.D., Yager, N., Gouillart, E., Yu, T., the scikit-image contributors: scikit-image: image processing in Python. PeerJ **2**, e453 (Jun 2014). https://doi.org/10.7717/peerj.453, https://doi.org/10.7717/peerj.453