Differentiable Mapper for Topological Optimization of Data Representation

Ziyad Oulhaj [1] **Mathieu Carrière** [2] **Bertrand Michel** [1]

Abstract

Unsupervised data representation and visualization using tools from topology is an active and growing field of Topological Data Analysis (TDA) and data science. Its most prominent line of work is based on the so-called *Mapper graph*, which is a combinatorial graph whose topological structures (connected components, branches, loops) are in correspondence with those of the data itself. While highly generic and applicable, its use has been hampered so far by the manual tuning of its many parameters—among these, a crucial one is the so-called *filter*: it is a continuous function whose variations on the data set are the main ingredient for both building the Mapper representation and assessing the presence and sizes of its topological structures. However, while a few parameter tuning methods have already been investigated for the other Mapper parameters (i.e., resolution, gain, clustering), there is currently no method for tuning the filter itself. In this work, we build on a recently proposed optimization framework incorporating topology to provide the first filter optimization scheme for Mapper graphs. In order to achieve this, we propose a relaxed and more general version of the Mapper graph, whose convergence properties are investigated. Finally, we demonstrate the usefulness of our approach by optimizing Mapper graph representations on several datasets, and showcasing the superiority of the optimized representation over arbitrary ones.

1. Introduction

Mapper graphs and TDA. The Mapper graph introduced in (Singh et al., 2007) is an essential tool of Topological Data Analysis (TDA), and has been used many times for visualization purposes on different types of data, including, but not limited to, single-cell sequencing (Wang et al., 2018; Zechel et al., 2014), neural network architectures (Mitra & Rao JV, 2021; Joseph et al., 2021), or 3D meshes (Wang, 2020; Rosen et al., 2018). Moreover, its ability to precisely encode (within the graph) the presence and sizes of geometric and topological structures in the data in a mathematically founded way (through the use of algebraic topology) has also proved beneficial for highlighting subpopulations of interest, which are usually detected as peculiar topological structures of significant sizes, and identifying the key features that best explain such subpopulations against the rest of the Mapper graph. This general pipeline has become a key component in, e.g., biological inference in single-cell data sets, where differentiating stem cells can usually be recovered from branching patterns in the corresponding Mapper graphs (Rizvi et al., 2017a).

Parameter selection. However, it has quickly become clear that the Mapper graph is quite sensitive to its parameters, in the sense that the structure of the graph can vary a lot under (even small) changes of its parameters. As such, several pipelines based on Mapper graphs actually involve brute force optimization: they first compute a grid of Mapper graphs corresponding to many different sets of parameters, and then they pick the best one, either by manual inspection or with arbitrary criteria—leading to prohibitive running times. In order to deal with this issue, several methods have been proposed in the literature for either assessing the statistical robustness of a given Mapper graph with respect to the distribution of the studied dataset (Belchí et al., 2020; Brown et al., 2021), or for tuning the Mapper parameters automatically (Carriere et al., 2018). Unfortunately, most tuning methods involve simple heuristics that only work for some, but not all Mapper parameters; in particular, the so-called *filter parameter* has never been treated, to the best of our knowledge. This is mostly because it is a general continuous function, and can thus vary in a much wilder parameter space than the other Mapper parameters.

In another line of work, ensemble methods have recently been proposed to combine Mapper graphs over multiple parameter sets, rather than trying to find the best one (Kang & Lim, 2021; Fitzpatrick et al., 2023), so as to be able

[1]Nantes Université, École Centrale Nantes, Laboratoire de Mathématiques Jean Leray, CNRS UMR 6629, Nantes, France [2]DataShape, Centre Inria d'Université Côte d'Azur, Sophia Antipolis, France. Correspondence to: Ziyad Oulhaj <ziyad.oulhaj@ec-nantes.fr>.

Proceedings of the 41st International Conference on Machine Learning, Vienna, Austria. PMLR 235, 2024.

to produce outputs that are more robust. However, this imposes to aggregate families of completely different filter functions, with no guarantees on the resulting graph. In this work, we follow a different approach, and rather attempt at identifying an "optimal" filter function by minimizing specific loss functions.

Another approach related to our work is (Bui et al., 2020), where an alternative way of constructing Mapper graphs is proposed using a fuzzy clustering algorithm. Even though we also adopt a probabilistic approach (that allows, e.g., a point to belong to disconnected intervals in the cover of the filter range), the underlying probabilistic formalism that we use is new, while there is none in (Bui et al., 2020). In particular, we introduce stochastic *assignment schemes* and we address the parameter selection problem within this framework.

Contributions. Our contribution is three-fold:

- We introduce *Soft Mapper*: a generalization of the combinatorial Mapper graph in the form of a probability distribution on Mapper graphs,
- We propose a filter optimization framework adapted to a smooth Soft Mapper distribution with provable convergence guarantees,
- We implement and showcase the efficiency of Mapper filter optimization through Soft Mapper on various data sets, with public, open-source code in `TensorFlow`.

The following of the article is organized as follows: in Section 2 we recall the basics on the Mapper algorithm, then in Section 3 we detail the Soft Mapper construction, which is the main focus of this work. We provide several interesting special cases of Soft Mapper in Section 4, before introducing topological losses that are specific to Mapper graphs in Section 5. We then present our optimization setting, in which a parameterized family of Mapper filter functions is optimized, in Section 6, and we apply it on 3-dimensional shapes and single cell RNA-sequencing data in Section 7. Finally, we discuss the results of this article and present possible future work directions in Section 8.

2. Background on Reeb and Mapper graphs

Reeb graphs. Mapper graphs can be understood as numerical approximations of *Reeb graphs*, that we now define. Let X be a topological space and let $f : X \to \mathbb{R}$ be a continuous function called *filter function*. Let $\sim_f$ be the equivalence relation between two elements x and y in X defined by: $x \sim_f y$ if and only if x and y are in the same connected component of $f^{-1}(z)$ for some z in $f(X)$. The Reeb graph $\mathrm{R}_f(X)$ of X is then simply defined as the quotient space $X/\sim_f$.

Mapper graphs. The Mapper was introduced in (Singh et al., 2007) as a discrete and computable version of the Reeb graph $\mathrm{R}_f(\mathcal{X})$. Assume that we are given a point cloud $\mathbb{X}_n = \{X_1, \dots, X_n\} \subseteq \mathcal{X}$ with known pairwise dissimilarities, as well as a filter function f computed on each point of $\mathbb{X}_n$. The Mapper graph can then be computed with the following generic version of the Mapper algorithm:

1. Cover the range of values $\mathbb{Y}_n = f(\mathbb{X}_n)$ with a set of consecutive intervals $I_1, \dots, I_r$ that overlap, i.e., one has $I_i \cap I_{i+1} \neq \varnothing$ for all $1 \leq i \leq r-1$.
2. Apply a clustering algorithm to each pre-image $f^{-1}(I_j)$, $j \in \{1, ..., r\}$. This defines a *pullback cover* $\mathcal{C} = \{\mathcal{C}_{1,1}, \dots, \mathcal{C}_{1,k_1}, \dots, \mathcal{C}_{r,1}, \dots, \mathcal{C}_{r,k_r}\}$ of $\mathbb{X}_n$.
3. The Mapper graph is defined as the *nerve* of $\mathcal{C}$. Each node $v_{j,k}$ of the Mapper graph corresponds to an element $\mathcal{C}_{j,k}$ of $\mathcal{C}$, and two nodes $v_{j,k}$ and $v_{j',k'}$ are connected by an edge if and only if $\mathcal{C}_{j,k} \cap \mathcal{C}_{j',k'} \neq \varnothing$.

3. Soft Mapper construction

In this section, we introduce our new construction *Soft Mapper*, which generalizes Mapper graphs and can be used for non-convex optimization. In order to do so, we first provide a general formalization of the Mapper construction that does *not* require overlapping intervals and filter functions. Then, we use this formalization to define *Soft Mapper*, which essentially consists in a distribution on regular Mapper graphs.

3.1. Mapper graphs built on latent cover assignments

Let $\mathbb{X}_n = \{x_1, ..., x_n\}$ be a point cloud lying in a metric space (X, d) and let $r \in \mathbb{N}^\star$. For instance, $\mathbb{X}_n$ can be obtained from sampling X^n according to some distribution μ. Then, let Clus be a clustering algorithm on (X, d), that is assumed to be fixed in the following of this work.

Latent cover assignments. Any binary matrix $e \in \{0,1\}^{n \times r}$ is then called an *r-latent cover assignment* of $\mathbb{X}_n$, where $e_{i,j} = 1$ must be understood as point x_i belonging to the j-th element of a *latent cover* of the data. For instance, in the standard version of Mapper presented in Section 2, the latent cover is obtained from a family of pre-images of intervals that cover the range of the filter function.

The procedure to compute a Mapper graph given an r-latent cover assignment $e \in \{0,1\}^{n \times r}$ is straightforward: simply replace $f^{-1}(I_j)$ by $\{x_i : e_{i,j} = 1\}$ in the generic Mapper algorithm in Section 2, then derive the pullback cover using the clustering algorithm Clus, and finally compute the Mapper graph as the nerve of the pullback cover.

Mapper function. Let $\mathbb{K}$ be the set of simplicial complexes of dimension less or equal to 1 (i.e., graphs) and such

that their sets of vertices (i.e., their $0-$skeletons) are subsets of the power set $\mathcal{P}(\mathbb{X}_n)$. We define the Mapper complex generating function as:

$$\text{MapComp}\colon \{0,1\}^{n\times r} \longrightarrow \mathbb{K},$$

where MapComp takes a latent cover assignment as input and creates the corresponding Mapper graph.

3.2. Cover assignment scheme and Soft Mapper

Now, we define stochastic schemes for generating latent cover assignments, that we call *cover assignment schemes*.

Definition 3.1. A *cover assignment scheme* is a double indexed sequence of random variables

$$A = (A_{i,j})_{\substack{1\leq i\leq n\\ 1\leq j\leq r}}$$

such that each $A_{i,j}$ is a Bernoulli random variable conditionally to $\mathbb{X}_n$. Let $p_{i,j}(\mathbb{X}_n)$ be the parameter of the Bernoulli distribution of $(A_{i,j}|\mathbb{X}_n)$, which is thus a function of the point cloud $\mathbb{X}_n$.

Note that, in Definition 3.1, the Bernoulli variables $A_{i,j}$ are not assumed to be independent nor identically distributed. Moreover, $p_{i,j}(\mathbb{X}_n)$ can depend only on its associated point x_i, or on the whole point cloud $\mathbb{X}_n$.

Definition 3.2. Let A be a cover assignment scheme. The *Soft Mapper* of A is defined as the associated distribution of Mapper complexes, which corresponds to the push forward measure of the distribution of A by the map MapComp.

4. Examples of cover assignment schemes

We now give example strategies to define cover assignment schemes, beginning with the one that corresponds to the standard Mapper construction defined in Section 2.

4.1. Standard cover assignment scheme

Let $f\colon \mathbb{X}_n \to \mathbb{R}$ be a filter function and let $(I_j)_{1\leq j\leq r}$ be a finite cover of the image $f(\mathbb{X}_n)$ of f. The standard Mapper graph is then defined as $\text{MapComp}(e^*)$, where for every $1 \leq i \leq n$ and $1 \leq j \leq r$:

$$e^*_{i,j} = 1 \text{ if and only if} f(x_i) \in I_j.$$

The cover assignment scheme A^*, in this case, is such that every entry $A^*_{i,j}$ follows a Dirac distribution on 1 if $f(x_i) \in I_j$, and a Dirac distribution on 0 otherwise. In other words, the parameters of the Bernoulli distributions satisfy $p_{i,j}(\mathbb{X}_n) = p_{i,j}(x_i) = 1$ if $f(x_i) \in I_j$, and 0 otherwise, that is

$$\mathbb{P}(A^* = e|\mathbb{X}_n) = \begin{cases} 1 & \text{if } e = e^*, \\ 0 & \text{otherwise.} \end{cases}$$

In this degenerated situation, the random variables $A^*_{i,j}$ are all independent conditionally to $\mathbb{X}_n$, and $A^*_{i,j}$ conditionally to $\mathbb{X}_n$ is equal to $A^*_{i,j}$ conditionally to x_i.

Remark 4.1. An alternative and relevant approach for the standard Mapper graph is to define the intervals I_j via the quantiles of the distribution of $f(\mathbb{X}_n)$. In this case, the random variables $A^*_{i,j}$ do not only depend on x_i, but also on the whole point cloud $\mathbb{X}_n$.

4.2. Smooth relaxation of the standard cover assignment scheme

Given some $\delta > 0$, we can now define a cover assignment scheme A_δ that approximates the cover assignment scheme A^* arising from the standard Mapper graph, but that also enjoys useful smoothness properties in the optimization setting that we will consider in the next section. Specifically, using the same notations as before, and denoting each element of the cover with $I_j = [a_j, b_j]$, consider, for each $j \in \{1, ..., r\}$, the function $q_j\colon X \longrightarrow [0, 1]$ defined with:

$$x \mapsto \begin{cases} 1, & \text{if } f(x) \in [a_j, b_j] \\ \exp(1 - 1/(1 - (\frac{a_j - f(x)}{\delta})^2)), & \text{if } f(x) \in (a_j - \delta, a_j] \\ \exp(1 - 1/(1 - (\frac{f(x) - b_j}{\delta})^2)), & \text{if } f(x) \in [b_j, b_j + \delta) \\ 0, & \text{otherwise} \end{cases}$$

Now, define $A_\delta = (A_{\delta,i,j})_{\substack{1\leq i\leq n\\ 1\leq j\leq r}}$ to be the random variable in $\{0,1\}^{n\times r}$ such that for every $(i, j) \in \{1, ..., n\} \times \{1, ..., r\}$:

$$A_{\delta,i,j} \mid \mathbb{X}_n \sim \mathcal{B}(q_j(x_i)),$$

with the $A_{\delta,i,j}$'s being jointly independent conditionally to $\mathbb{X}_n$. As for the standard cover, the Bernoulli parameter $p_{i,j} = q_j(x_i)$ depends on its associated point x_i and also on the chosen filter f.

Moreover, notice that for every $x_i \in \mathbb{X}_n$ and $j \in \{1, ..., r\}$:

$$q_j(x_i) \xrightarrow[\delta\to 0]{} \begin{cases} 1, & \text{if } f(x_i) \in I_j \\ 0, & \text{otherwise} \end{cases}$$

and this shows that $A_\delta \xrightarrow[\delta\to 0]{\mathcal{L}} A^*$. Note that even though we can approximate the standard Mapper graph in this way, we do not always want to do so. For example, there could be cases where δ needs to be large enough so as to account for some uncertainty on the bounds of the cover $(I_j)_{1\leq j\leq r}$. An illustration of how A_δ is built on top of a dataset, a filter function and a cover of its range is shown in Figure 1.

Remark 4.2. Note that the same relaxed construction can be made for a multi-dimensional Mapper, i.e., for filter functions taking values in $\mathbb{R}^d$ (Carrière & Michel, 2022), by making slight adjustments to the definition of q_j using the Euclidean norm.

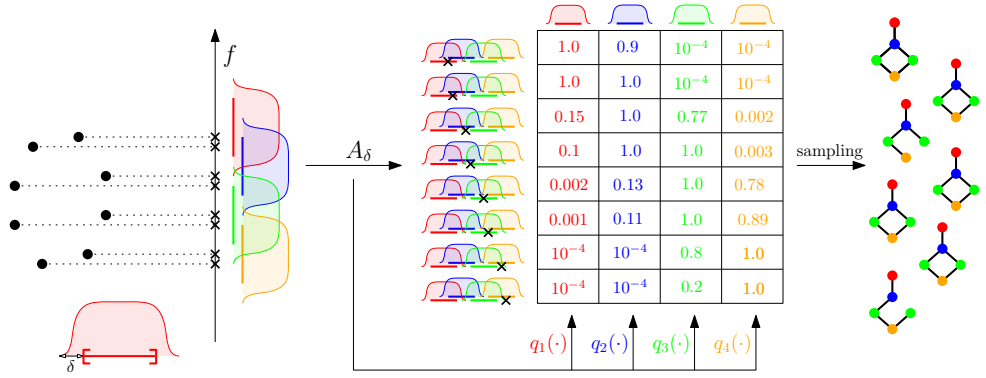

Figure 1. Illustration of the smooth assignment scheme. Intervals that constitute the cover of the range of filter values are represented in different colors. The functions $(q_j)_{1\leq j\leq r}$ are plotted above each interval. The probabilities that give the distribution of the assignment scheme are represented in the middle. On the right, different Mapper graph samples associated to the assignment scheme are shown.

An additional example of a possible cover assignment scheme, which does not imply the existence of a filter function, is given in Appendix A.

5. Topological risk of Soft Mappers

We now switch to the problem of designing filter functions automatically for Mapper graphs using Soft Mapper. To answer this ill-posed problem, we propose to look for filter functions that are optimal with respect to some topological criteria associated to their (Soft)Mapper graphs. In particular, we focus on topological losses based on *persistent homology*.

5.1. Topological signature for Mapper graphs

Persistent homology. Persistent homology is a powerful tool that allows to encode the topological information contained in a nested family of simplicial complexes, also called a *filtered simplicial complex*, see for instance (Edelsbrunner & Harer, 2010) for a general introduction. It traces the evolution of the homology groups of the nested complexes across different scales, producing topological descriptors that are, in particular, useful in machine learning pipelines (Chazal & Michel, 2021). In the context of Mapper graphs, a variation of persistent homology called *extended* persistent homology has been proved useful, as applying it on Mapper graphs produces descriptors called *extended persistence diagrams*. These diagrams only require to define a *filtration function* on the graph, and are made of points in the Euclidean plane, each point encoding the presence and size (w.r.t. the filtration function) of a particular topological structure of the Mapper graph (such as a connected component, a branch or a loop). See Section 2 of (Carrière et al., 2020) for a brief introduction to extended persistence and (Cohen-Steiner et al., 2009) for a detailed presentation.

We now define a filtration function on Mapper graphs in order to compute extended persistence diagrams. Let $\mathcal{F}(\mathbb{X}_n, \mathbb{R})$ be the space of real valued functions defined on the point cloud $\mathbb{X}_n$. For a function $F \in \mathcal{F}(\mathbb{X}_n, \mathbb{R})$, we first associate a filtration ϕ to some $K \in \mathrm{im}(\mathrm{MapComp})$ with:

$$\forall \sigma \in K \ : \ \phi(\sigma) = \max_{c \in \sigma} \frac{\sum_{x \in c} F(x)}{\mathrm{card}(c)},$$

that is, node filtration values are defined as the average filter values of the data points associated to the node, and edge filtration values are computed as the maximum of their node values. Then, we compute the extended persistence diagram (which we consider as a subset of $\mathbb{R}^2$) of the filtered simplicial complex (K, ϕ). We denote by MapPers the function that takes a Mapper graph and a scalar function on $\mathbb{X}_n$, and then outputs the persistence diagram:

$$\mathrm{MapPers}\colon \mathbb{K} \times \mathcal{F}(\mathbb{X}_n, \mathbb{R}) \longrightarrow \mathcal{P}(\mathbb{R}^2).$$

Persistence specific loss. Now, we introduce a generic notation for a loss function—or, more simply, a statistic—that associates a real value to any extended persistence diagram. Denoting PD as the set of subsets of $\mathbb{R}^2$ consisting of a finite number of points outside the diagonal $\Delta = \{(x, x) : x \in \mathbb{R}\}$, such a function can be written as $\ell\colon PD \longrightarrow \mathbb{R}$.

5.2. Statistical risk of the topological signature associated to Soft Mapper

We finally compute the loss associated to a Mapper graph with the function

$$\begin{aligned} \mathcal{L}\colon \{0,1\}^{n\times r} \times \mathcal{F}(\mathbb{X}_n, \mathbb{R}) &\longrightarrow \mathbb{R} \\ (e, F) &\longmapsto \ell\left(\mathrm{MapPers}\left(\mathrm{MapComp}(e), F\right)\right). \end{aligned}$$

Then, we define the risk of a Soft Mapper $\mathrm{MapComp}(A)$ by integrating the loss according to the distribution of the Soft

Mapper, or equivalently according to the distribution of the cover assignment scheme:

$$\mathbb{E}\left(\mathcal{L}(A,F)|\mathbb{X}_n\right) = \sum_{e\in\{0,1\}^{n\times r}} \mathcal{L}(e,F)\cdot\mathbb{P}(A=e|\mathbb{X}_n).$$

Here, both the distribution of A and the risk are conditional to $\mathbb{X}_n$. Note that the risk could also be integrated with respect to the distribution of $\mathbb{X}_n$. However, in this article, we only consider the non-integrated version of the risk.

6. Conditional risk optimization with respect to parameters

Now that we have properly defined risks associated to Soft Mapper distributions, we study in this section the convergence properties of filter optimization schemes minimizing such risks.

6.1. Problem setting

Let us introduce a parameterized family of functions $\{f_\theta : \mathbb{X}_n \to \mathbb{R}, \theta \in \mathbb{R}^s\}$. In order to simplify notations, we assume in the following of the article that the function F used to compute persistence diagrams and the filter function f_θ used to design cover assignments are the same, $F = f_\theta$. Let A be a cover assignment scheme whose joint distribution $\mathbb{P}_\theta$ depends on the filter function f_θ; that is the Bernoulli parameters $p_{i,j}$ may depend on the filter function values and the parameters θ. Note that this dependency is not only true for marginals of the distribution of the cover assignment scheme, but also eventually for its dependency structure.

Our goal is to find the optimal set of parameters $\bar{\theta}$ that minimizes the topological risk associated to MapComp(A), when f_θ is used to define the filtration values on the Mapper graphs. In other words, if we denote:

$$\begin{aligned} \mathrm{L}\colon \mathbb{R}^s &\longrightarrow \mathbb{R} \\ \theta &\longmapsto \mathbb{E}_\theta(\mathcal{L}(A,f_\theta)|\mathbb{X}_n), \end{aligned} \tag{1}$$

our aim is to find a minimizer of L. Note that in the definition of L, the expectation depends on θ because the distribution of A also depends on it.

In order to prove guarantees about minimizing L, we follow (Carriere et al., 2021), which uses the theoretical background introduced in (Davis et al., 2020), in which the authors prove that stochastic gradient descent algorithms converge under certain conditions. To use this framework, it suffices to prove two points (see Corollary 5.9. in (Davis et al., 2020) and Appendix B):

- L is definable in an o-minimal structure,
- L is locally Lipschitz.

Remark 6.1. When the cover assignment scheme is defined as the standard cover assignment scheme corresponding to the standard Mapper graph (see Section 4.1), this problem amounts to finding an optimal f_θ that can be used to compute Mapper graphs. We will see however that convergence of the optimization problem in this case is without guarantees, which constitutes the main motivation for defining our smooth relaxation Soft Mapper (see Section 4.2).

6.2. Theoretical guarantees on the convergence of a gradient descent scheme

Under regularity assumptions on the parameterized family of filter functions $\mathcal{F} = \{f_\theta\colon \mathbb{X}_n \longrightarrow \mathbb{R}, \theta \in \mathbb{R}^s\}$, we now show that the risk L in Equation (1) is definable and smooth.

Theorem 6.2. *Suppose that there exists an o-minimal structure $\mathcal{S}$ such that:*

- *for every $x \in \mathbb{X}_n$, the function $\theta \mapsto f_\theta(x)$ is definable in $\mathcal{S}$ and is locally Lipschitz,*
- *for every $m \in \mathbb{N}$, the restriction of ℓ to the set of (extended) persistence diagrams of size m is definable in $\mathcal{S}$ and is locally Lipschitz,*
- *for every $e \in \{0,1\}^{n\times r}$, the function $\theta \mapsto \mathbb{P}_\theta(A = e|\mathbb{X}_n)$ is definable in $\mathcal{S}$ and is locally Lipschitz.*

Then L *is definable in $\mathcal{S}$ and is locally Lipschitz.*

Remark 6.3. Our proof of Theorem 6.2 is given in Appendix C in the case where regular persistent homology is used, but it can be extended in a straightforward way to extended persistence diagrams, as extended persistent homology on a simplicial complex K can be equivalently seen as regular persistent homology on the cone on K (see chapter VII.3 in (Edelsbrunner & Harer, 2010)). Moreover, defining the filtration on the coned complex also extends naturally by using affine transformations.

Under the assumptions of Theorem 6.2, it is then possible to give guarantees on the convergence of a stochastic gradient descent scheme to some critical points of L. This only requires additional, but mild and not very restrictive technical conditions regarding the stochastic gradient descent algorithm itself (see Appendix D).

6.3. Discussing the assumptions of Theorem 6.2

In this section, we discuss the assumptions of Theorem 6.2, and provide usual cases in which they are satisfied.

Assumption 1. The first assumption concerns the smoothness of the parameterized family of functions $\{f_\theta\colon \mathbb{X}_n \longrightarrow \mathbb{R}, \theta \in \mathbb{R}^s\}$ and its regularity with respect to the set of parameters θ. As mentioned in Appendix B, following the

result of (Wilkie, 1996), semi-algebraic functions (for example polynomial, rational, minimum and maximum functions), the exponential function and functions defined as compositions and usual operations between them are all definable in a same o-minimal structure. Furthermore, choosing continuously differentiable functions is sufficient to also have the local Lipschitz property. As such, the family of linear functions $\{f_\theta \colon x \mapsto \langle x, \theta \rangle,\ \theta \in \mathbb{R}^s\}$ satisfies the assumption, as well as the family of parameterized fully-connected neural networks since they are defined by composition between matrix products (which are polynomial) and activation functions involving exponential, maximum and hyperbolic functions.

Assumption 2. The second assumption concerns the persistence-based loss ℓ, that is used to compute the topological risk. In (Carriere et al., 2021), the authors list a number of possible functions for ℓ that satisfy our second assumption. For example, ℓ can be the opposite of the L_1 *total persistence*, i.e., the sum of the non-essential bars in the persistence diagram which quantifies the information given by it. It is defined as:

$$\{(u_i, v_i)\}_{1 \leq i \leq n} \longmapsto -\sum_{i=1}^{n} |u_i - v_i|.$$

In the numerical experiments below, we focus solely on this loss. Our motivation is that a large total persistence provides a topologically rich Mapper complex, with more persistent topological structures. Alternatively, L_p total persistence can also be used to reinforce the weight of the most persistent structures. Moreover, persistent entropy (Chintakunta et al., 2015; Atienza et al., 2020), which is large for barcodes with bars of equal length and small for barcodes with bars of varying lengths, constitutes an interesting alternative. The loss can also be computed from persistence landscapes (Bubenik, 2015) or from the bottleneck distance (Carriere et al., 2021) to a target persistence diagram, e.g. to a persistence diagram built on the dataset using a Rips filtration. Future work includes running a thorough investigation of the pros and cons of the different choices of a topological loss in our framework.

Assumption 3. Finally, the third assumption concerns the cover assignment scheme A. More specifically, it requires the regularity and smoothness of the success probabilities that give the distribution of A.
Interestingly, this assumption does *not* hold for the standard cover assignment scheme. For example, consider the elementary example where $\mathbb{X}_n \subseteq \mathbb{R}$ and A is the standard cover assignment scheme, which is degenerate at e_θ, and which corresponds to the linear filter function $f_\theta \colon x \mapsto \langle x, \theta \rangle$ and a cover (I_j) of its image. Fix a non-zero positive point $x \in \mathbb{X}_n$ (a similar argument can be made if it is negative) and a left hand bound a_j of one of the intervals. Denoting $\theta_0 = \frac{a_j}{x}$, we have that $\theta \mapsto \mathbb{P}_\theta(A = e_{\theta_0} | \mathbb{X}_n)$ is discontinuous at θ_0, since $\forall \epsilon > 0 \ : \ \langle x, \theta_0 - \epsilon \rangle = x \cdot (\theta_0 - \epsilon) < a_j$, and therefore, $\mathbb{P}_{\theta_0 - \epsilon}(A = e_{\theta_0} | \mathbb{X}_n) = 0$.

This constitutes the main motivation for introducing our smooth cover assignment scheme because the functions $\theta \mapsto \mathbb{P}_\theta(A = e | \mathbb{X}_n)$ are in this case products of the functions q_j, which are smooth with respect to the parameters (if our first assumption holds, for a detailed proof see Appendix E).

Algorithm 1 Soft Mapper Optimization Algorithm

Require: Initial parameter set θ_0, Number of Monte Carlo random samples M, Learning rate sequence $(\alpha_i)_i$, Random noise sequence $(\xi_i)_i$, Number of epochs N.
  for $0 \leq i \leq N-1$ **do**
    for $1 \leq m \leq M$ **do**
      $e \leftarrow$ sample from $\mathbb{P}_{\theta_i}$
      $y_{i,m} \leftarrow$ an element of the sub-differential in θ_i of $\mathcal{L}_e \colon \theta \mapsto \mathcal{L}(e, f_\theta)$
    end for
    $y_i \leftarrow \frac{1}{M} \sum_{m=1}^{M} y_{i,m}$
    $\theta_{i+1} \leftarrow \theta_i - \alpha_i (y_i + \xi_i)$
  end for
  return θ_N

6.4. Computing the conditional risk in practice

Computing the conditional risk $\mathrm{L}(\theta)$, for a fixed $\theta \in \mathbb{R}^s$, can be costly in practice since it requires computing the loss $\mathcal{L}(e, f_\theta)$ for every possible cover assignment $e \in \{0,1\}^{n \times r}$. As such, we estimate $\mathrm{L}(\theta)$ with Monte Carlo methods. Note that this is possible here because the distribution $\mathbb{P}_\theta$ of the cover assignment scheme is indeed explicitly defined and known at each step of the gradient descent. If M is a non-zero integer and $(e^{(m)})_{1 \leq m \leq M}$ is a sequence of independent realizations of the cover assignment scheme A, then the Monte Carlo approximation of the conditional risk is:

$$\tilde{\mathrm{L}}(\theta) = \frac{1}{M} \sum_{m=1}^{M} \mathcal{L}(e^{(m)}, f_\theta).$$

The law of large numbers gives:

$$\tilde{\mathrm{L}}(\theta) \xrightarrow[M \to \infty]{a.s} \mathrm{L}(\theta).$$

Moreover, the coordinates of A follow a Bernoulli conditional distribution, making repeated random sampling straightforward, at least when the marginal distributions of $\mathbb{P}_\theta$ are assumed to be independent.

For a fixed point cloud $\mathbb{X}_n$, a chosen family of parameterized conditional probabilities $\theta \mapsto \mathbb{P}_\theta(\cdot | \mathbb{X}_n)$ and a family of parameterized filters $\theta \mapsto f_\theta$, our corresponding optimization algorithm is detailed in Algorithm 1. Its complexity analysis is given in Appendix F.

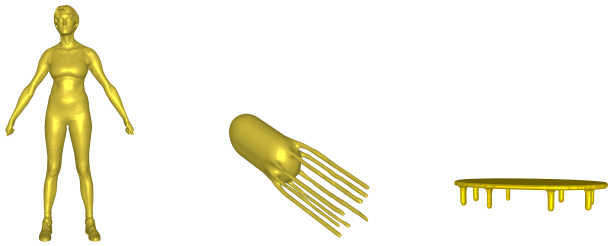

Figure 2. Meshes of 3-dimensional point clouds representing from left to right: a human, an octopus and a table.

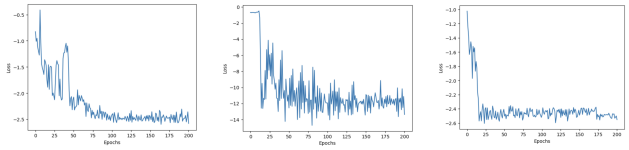

Figure 3. Learning curves for the 3-dimensional shapes corresponding, from left to right, to: the human, the octopus and the table.

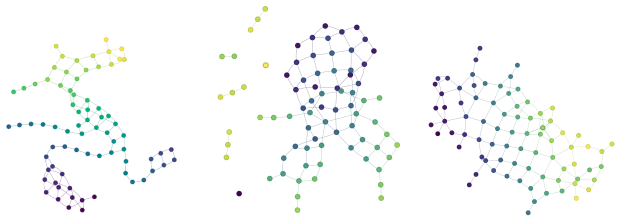

Figure 4. Regular Mapper graphs computed with the initial filter function, corresponding, from left to right, to: the human, the octopus and the table. Vertices are colored using the mean value of the filter function in the corresponding clusters.

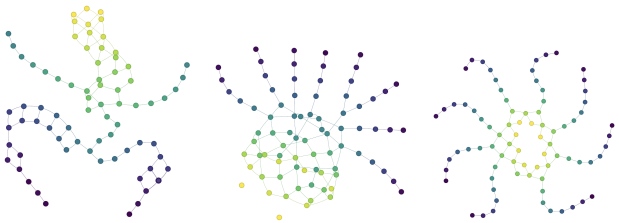

Figure 5. Regular Mapper graphs computed with the optimized filter function, corresponding, from left to right, to: the human, the octopus and the table. Vertices are colored using the mean value of the filter function in the corresponding clusters.

7. Numerical Experiments

In this section, we illustrate the efficiency of optimizing filter functions with Soft Mapper. In particular, we show that Mapper graphs computed from an optimized filter function (computed with gradient descent on Soft Mapper) are generally much better structured than Mapper graphs obtained from arbitrary filters (as is usually done in the Mapper applications). We present applications on 3D shape data in Section 7.1 and on single-cell RNA sequencing data in Section 7.2. Our code is available at (Oulhaj, 2024).

7.1. Mapper optimization on 3D shapes

A first application where we can use the Soft Mapper optimization setting is finding linear filters in order to skeletonize 3-dimensional shapes with Mapper graphs. Here, our dataset $\mathbb{X}_n$ consists each time of a point cloud embedded in $\mathbb{R}^3$. The different point clouds we study are displayed (as meshes) in Figure 2. The parametric family of functions is linear, i.e., equal to $\{f_\theta \colon x \mapsto \langle x, \theta \rangle,\ \theta \in \mathbb{R}^3\}$, and the cover assignment scheme A_δ is the smooth relaxation of the standard case, with $\delta = 10^{-2} \cdot (\max_{x \in \mathbb{X}_n} f_\theta(x) - \min_{x \in \mathbb{X}_n} f_\theta(x))$. The cover of the image space is given by r intervals of the same length, such that consecutive intervals have a percentage g of their length in common. The clustering algorithm for the three shapes is KMeans. The values of r (also called resolution), g (also called gain) and the number of clusters in the KMeans algorithm, for each 3-dimensional shape, are summarized in Appendix G.

Objective. Intuitively, the optimal directions to filter the 3-dimensional shapes (in a topological sense) are: the vertical direction for the human, the parallel direction to the tentacles for the octopus and the perpendicular direction to the upper surface for the table. This can be justified by the fact that these directions induce Mapper graphs with more topological structures. We will therefore measure the quality of our results by comparing our optimized directions $\bar{\theta}$ to the ones cited above. To find $\bar{\theta}$, we use the opposite of the L_1 total (regular) persistence as a persistence specific loss ℓ and we run Algorithm 1 with $N = 200$ and $M = 10$, each time taking the diagonal as the initial direction, i.e. $\theta_0 = (\frac{1}{\sqrt{3}}, \frac{1}{\sqrt{3}}, \frac{1}{\sqrt{3}})^T$. The learning curves for each 3-dimensional shape are displayed in Figure 3, and the cosine distances between the optimized directions and those we identified as intuitively optimal are summarized in the following table, in which one can see that we are able to recover these intuitive directions with gradient descent.

Human	Octopus	Table
0.9999	-0.9984	0.9993

Qualitative assessment. One can see, in Figures 4 and 5, that the regular Mapper graphs built with the initial and final (optimized) filter functions show clear improvement in the ability of the graphs to act as skeletons of the original point clouds. As such, we see that optimizing the Soft Mapper corresponding to the smooth relaxation of the standard cover assignment scheme succeeds in identifying optimal filter functions. The third shape, representing a table,

		PCA	t-SNE	UMAP	Mapper (initial)	Mapper (optim)
human	Rand	-0.00151	0.19 ± 5.39e-05	-0.000387 ± 2.58e-08	0.238	**0.379** ± **0.0082**
	MI	0.00205	0.429 ± 3.08e-06	0.000137 ± 8.57e-08	0.425	**0.563** ± **0.00263**
	Comp	0.164	0.463 ± 4.56e-05	0.165 ± 0.000206	0.492	**0.613** ± **0.00104**
	FM	0.39	0.341 ± 1.95e-05	0.392 ± 6.67e-08	0.412	**0.518** ± **0.00283**
octopus	Rand	-0.00421	0.173 ± 6.73e-05	0.0426 ± 0.00377	0.0777	**0.568** ± **0.0112**
	MI	-0.00216	0.5 ± 3.32e-06	0.0972 ± 0.0173	0.347	**0.512** ± **0.000247**
	Comp	0.0701	**0.641** ± **8.43e-05**	0.334 ± 0.105	0.315	0.519 ± 0.00167
	FM	0.545	0.492 ± 2.31e-05	0.555 ± 0.000134	0.298	**0.708** ± **0.00856**
table	Rand	-0.000351	0.0136 ± 1.84e-05	-0.0007 ± 4.85e-07	-0.0134	**0.161** ± **2.99e-05**
	MI	-0.000102	0.000309 ± 5.04e-05	2.23e-05 ± 6.17e-08	0.00905	**0.135** ± **1.89e-05**
	Comp	0.0136	0.000484 ± 4.05e-05	0.0142 ± 1.6e-06	0.0194	**0.41** ± **1.02e-05**
	FM	0.887	0.806 ± 9.63e-05	0.887 ± 1.99e-07	0.818	**0.896** ± **1.15e-07**
sctda	Rand	0.0716	0.259 ± 8.06e-05	0.266 ± 1.59e-07	0.00979	**0.381** ± **1.21e-06**
	MI	0.246	**0.506** ± **2.51e-05**	0.503 ± 5.38e-07	0.0539	0.487 ± 2.03e-06
	Comp	**0.919**	0.672 ± 4.84e-05	0.657 ± 2.73e-07	0.124	0.567 ± 4.02e-06
	FM	0.522	0.553 ± 6.93e-06	0.552 ± 3.59e-07	0.446	**0.581** ± **3.41e-06**

Table 1. Clustering scores (Rand, Mututal Information, Completeness and Fowlkes-Mallow) that compare ground truth clusterings to hierarchical clusterings induced by different latent representations (PCA, t-SNE, UMAP and Mapper), for the 3D shapes dataset and the human preimplantation dataset (sctda).

is particularly interesting. Indeed, the optimal direction that we captured is different from the first and the second principal components computed by PCA, since the principal plane of the point cloud is given by the table surface. Hence, there is a contrast between our total persistence criterion and the maximum variance criterion of PCA.

Quantitative assessment. We design a quantitative score (in order to compare to baselines) by using ground-truth information: we compute four clustering scores (Rand (Rand, 1971), Mutual Information, Completeness and Fowlkes-Mallow (Fowlkes & Mallows, 1983)) between: the clusterings induced by the 3D shape segmentations (which assign labels to the 3D shape vertices, such as arm, leg, torso, etc.) and the clusterings obtained by running hierarchical clusterings on the latent representations (using either the Euclidean distances in the latent PCA/t-SNE/UMAP spaces, or the geodesic distances induced by the Mapper complexes).

Table 1 summarizes the results of this analysis. We see that the clusterings induced by Mapper complexes are particularly efficient, as Mapper is known to be good for extracting non-linear, complex structures: for example, the arms of the human shape, and the legs of the octopus and of the table can all be detected as branches (0-dimensional topological features) of Mapper complexes, while they can be squeezed in the other methods' latent spaces.

7.2. Mapper optimization on RNA sequencing data

We now apply Mapper optimization on the human preimplantation dataset of (Petropoulos et al., 2016), which can also be found in the tutorial of the `scTDA` Python library. The dataset consists of $n = 1,529$ cells form 88 human preimplantation embryos, sampled at 5 different timepoints. The dataset can be accessed in the following link (sct), and it contains the expression levels for $p = 26,270$ genes for each individual cell. The information of the sampling timepoint for each cell is also given, but we do not include it during optimization. The dataset is first preprocessed using the `Seurat` package in `R` (gene counts for each cell are divided by the total counts for that cell and multiplied by 10^4, and then they are natural-log transformed using $\log(1 + \cdot)$), which produces a normalized dataset $\mathbb{X}_n \subseteq \mathbb{R}^p$. The parametric family of filter functions we wish to optimize is also linear here, i.e. equal to $\{f_\theta \colon x \mapsto \langle x, \theta \rangle, \theta \in \mathbb{R}^p\}$, and the cover assignment scheme A_δ is the smooth relaxation of the standard case with $\delta = 10^{-5} \cdot (\max_{x \in \mathbb{X}_n} f_\theta(x) - \min_{x \in \mathbb{X}_n} f_\theta(x))$. The cover of the image space is given by 25 intervals of the same length, such that consecutive intervals have a percentage of 30% of their length in common. The clustering algorithm used is agglomerative clustering and its threshold is fixed using a Hausdorff distance heuristic: we first compute the Hausdorff distance between $\mathbb{X}_n$ and a randomly sampled subset of $\mathbb{X}_n$ of size $n/3 \approx 500$, then we manually tune the threshold using factors of this distance until we get Mapper graphs of reasonable size.

For this dataset, additional experiments with filter functions of the form $f(x) = \sum_{i=1}^n \alpha_i K(x_i, x)$, where K is a Gaussian kernel; and neural network filter functions with two dense layers (of 32 and 16 units respectively) and ReLU activations are given in Appendix G, as well as an ablation

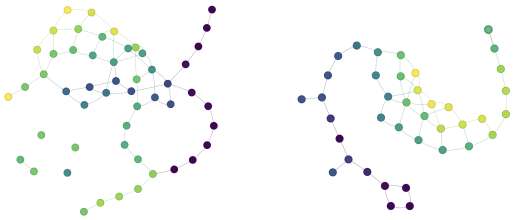

Figure 6. Regular Mapper graphs for the human preimplantation dataset computed using: in the left the initial filter function and in the right the optimized filter function. Vertices are colored using the mean value of the sampling timepoint in the clusters.

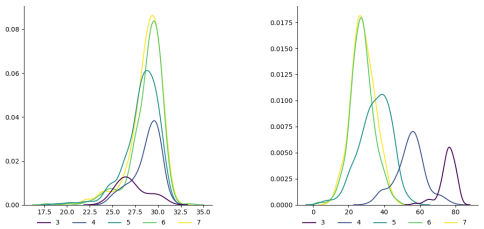

Figure 7. Estimated density of each subset of cells having the same sampling timepoint, with respect to: in the left the initial filter function values and in the right the optimized filter function values. Colors indicate the sampling timepoint in days.

study w.r.t. the δ parameter in Appendix H. An additional RNA sequencing dataset experiment is in Appendix I.

Objective. To find $\bar{\theta}$, we use the opposite of the L_1 total extended persistence as a persistence specific loss ℓ and we run Algorithm 1 with $N = 200$ and $M = 10$, taking the diagonal as the initial direction, i.e. $\theta_0 = (\frac{1}{\sqrt{p}}, ..., \frac{1}{\sqrt{p}})^T$. The learning curve in displayed in Figure 10 of Appendix G. The regular Mapper graphs computed using the initial and the final filter functions are displayed in Figure 6, and are colored with respect to the time component (which was not included in the training dataset).

Qualitative assessment. One can see that the data representation in the Mapper graph produced by the optimized filter function fits the time structure better than with the initial function. In order to confirm this, we isolate each subset of cells having the same sampling timepoint and we plot their respective estimated densities with respect to the initial and the optimized filter function values, in Figure 7. One can see that the optimized filter that we captured is capable of sorting the cells with respect to time, especially at the early timepoints. The reduced performance in this aspect for the later timepoints is, in our guess, due to slowing down of the cell differentiation process. Furthermore, the comparison, in Table 3 of Appendix G, between Pearson's correlation coefficients also show that the optimized filter is more correlated to time. We also verify that the branches

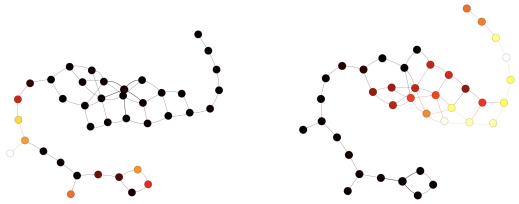

Figure 8. Regular Mapper graph computed using the optimized filter function, colored using the mean normalized expression of: in the left gene HTR3E and in the right gene CDX1.

in our optimized Mapper graph correspond to the same two genes, HTR3E for the early timepoints and CDX1 for the later ones, that were identified by (Rizvi et al., 2017b), see Figure 8. We also identified a few nodes in the branch containing the cells which were sampled in the early stages, that do not contain a high expression level for the HTR3E gene, potentially pointing out another subpopulation of cells with distinct genomic profiles.

Quantitative assessment We compute the same scores as for the previous experiment, with the clustering induced by time as ground truth. Table 1 shows a less striking difference, compared to the 3D shapes experiment, between the clusterings induced by the Mapper and those induced by the other methods. However, the scores (after optimization) are still comparable to the baselines. In all cases nonetheless, optimizing the Mapper filter with the total persistence loss is beneficial and results in an increase in the scores.

8. Discussion and future work

In this article, we have introduced Soft Mapper, a distributional and smoother version of the standard Mapper graph, with provable convergence guarantees using persistence-based losses and risks. Our case study in this article was finding an optimal filter function, among a parameterized family of functions, in order to construct regular Mapper graphs incorporating an optimized and maximal amount of topological information. We then produced examples of such optimization processes on real 3D shape and single-cell RNA sequencing data, for which we were able to obtain structured Mapper representations in an unsupervised way. These representations, especially for the single cell RNA sequencing data, are not meant to represent novel or state of the art data representations in their respective research domains, but as a proof of concept of the practical benefit of our method. Moreover, our construction is not limited to the filter optimization setting as a whole. Possible future work includes inspecting different choices of filter function families and topological losses, and studying Soft Mappers based on different cover assignment schemes, like the Gaussian cover assignment scheme defined in Appendix A.

Acknowledgment. The research was supported by three grants from Agence Nationale de la Recherche: GeoDSIC ANR-22-CE40-0007, ANR JCJC TopModel ANR-23-CE23-0014 and AI4scMed, France 2030 ANR-22-PESN-000.

Impact Statement

This article presents work whose goal is to advance the field of Machine Learning. There are many potential societal consequences of our work, none which we feel must be specifically highlighted here.

References

sctda. https://github.com/CamaraLab/scTDA. Accessed: 2024-01-23.

Atienza, N., González-Díaz, R., and Soriano-Trigueros, M. On the stability of persistent entropy and new summary functions for topological data analysis. *Pattern Recognition*, 107:107509, 2020.

Belchí, F., Brodzki, J., Burfitt, M., and Niranjan, M. A numerical measure of the instability of mapper-type algorithms. *The Journal of Machine Learning Research*, 21 (1):8347–8391, 2020.

Brown, A., Bobrowski, O., Munch, E., and Wang, B. Probabilistic convergence and stability of random Mapper graphs. *Journal of Applied and Computational Topology*, 5:99–140, 2021.

Bubenik, P. Statistical topological data analysis using persistence landscapes. *Journal of Machine Learning Research*, 16(3):77–102, 2015.

Bui, Q.-T., Vo, B., Do, H.-A. N., Hung, N. Q. V., and Snasel, V. F-mapper: A fuzzy mapper clustering algorithm. *Knowledge-Based Systems*, 189:105107, 2020.

Carrière, M. and Michel, B. Statistical analysis of mapper for stochastic and multivariate filters. *Journal of Applied and Computational Topology*, 6(3):331–369, 2022.

Carriere, M., Michel, B., and Oudot, S. Statistical analysis and parameter selection for mapper. *The Journal of Machine Learning Research*, 19(1):478–516, 2018.

Carrière, M., Chazal, F., Ike, Y., Lacombe, T., Royer, M., and Umeda, Y. Perslay: A neural network layer for persistence diagrams and new graph topological signatures. In *International Conference on Artificial Intelligence and Statistics*, pp. 2786–2796. PMLR, 2020.

Carriere, M., Chazal, F., Glisse, M., Ike, Y., Kannan, H., and Umeda, Y. Optimizing persistent homology based functions. In *International conference on machine learning*, pp. 1294–1303. PMLR, 2021.

Chazal, F. and Michel, B. An introduction to topological data analysis: fundamental and practical aspects for data scientists. *Frontiers in artificial intelligence*, 4:108, 2021.

Chintakunta, H., Gentimis, T., Gonzalez-Diaz, R., Jimenez, M.-J., and Krim, H. An entropy-based persistence barcode. *Pattern Recognition*, 48(2):391–401, 2015.

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

Coste, M. *An introduction to o-minimal geometry*. Istituti editoriali e poligrafici internazionali Pisa, 2000.

Davis, D., Drusvyatskiy, D., Kakade, S., and Lee, J. D. Stochastic subgradient method converges on tame functions. *Foundations of computational mathematics*, 20(1): 119–154, 2020.

Edelsbrunner, H. and Harer, J. *Computational topology: an introduction*. American Mathematical Society, 2010.

Fitzpatrick, P., Jurek-Loughrey, A., Dłotko, P., and Del Rincon, J. M. Ensemble learning for mapper parameter optimization. In *2023 IEEE 35th International Conference on Tools with Artificial Intelligence (ICTAI)*, pp. 129–134. IEEE, 2023.

Fowlkes, E. B. and Mallows, C. L. A method for comparing two hierarchical clusterings. *Journal of the American statistical association*, 78(383):553–569, 1983.

Fry, R. and McManus, S. Smooth bump functions and the geometry of banach spaces: a brief survey. *Expositiones Mathematicae*, 20(2):143–183, 2002.

Joseph, B. B., Pham, T., and Hastings, C. Topological data analysis in conjunction with traditional machine learning techniques to predict future mdap pm ratings. Acquisition Research Program, 2021.

Kang, S. J. and Lim, Y. Ensemble mapper. *Stat*, 10(1):e405, 2021.

Mitra, S. and Rao JV, K. Experiments on fraud detection use case with qml and tda mapper. In *2021 IEEE International Conference on Quantum Computing and Engineering (QCE)*, pp. 471–472, 2021. doi: 10.1109/QCE52317.2021.00083.

Oulhaj, Z. Mapper filter optimization. https://github.com/ZiyadOulhaj/Mapper-Optimization, 2024.

Petropoulos, S., Edsgärd, D., Reinius, B., Deng, Q., Panula, S. P., Codeluppi, S., Reyes, A. P., Linnarsson, S., Sandberg, R., and Lanner, F. Single-cell rna-seq reveals lineage and x chromosome dynamics in human preimplantation embryos. *Cell*, 165(4):1012–1026, 2016.

Rand, W. M. Objective criteria for the evaluation of clustering methods. *Journal of the American Statistical association*, 66(336):846–850, 1971.

Rizvi, A., Cámara, P., Kandror, E., Roberts, T., Schieren, I., Maniatis, T., and Rabadán, R. Single-cell topological RNA-seq analysis reveals insights into cellular differentiation and development. *Nature Biotechnology*, 35: 551–560, 2017a.

Rizvi, A. H., Camara, P. G., Kandror, E. K., Roberts, T. J., Schieren, I., Maniatis, T., and Rabadan, R. Single-cell topological rna-seq analysis reveals insights into cellular differentiation and development. *Nature biotechnology*, 35(6):551–560, 2017b.

Rosen, P., Hajij, M., Tu, J., Arafin, T., and Piegl, L. Inferring quality in point cloud-based 3d printed objects using topological data analysis. *arXiv preprint arXiv:1807.02921*, 2018.

Schiebinger, G., Shu, J., Tabaka, M., Cleary, B., Subramanian, V., Solomon, A., Gould, J., Liu, S., Lin, S., Berube, P., et al. Optimal-transport analysis of single-cell gene expression identifies developmental trajectories in reprogramming. *Cell*, 176(4):928–943, 2019.

Singh, G., Mémoli, F., Carlsson, G. E., et al. Topological methods for the analysis of high dimensional data sets and 3d object recognition. *PBG@ Eurographics*, 2:091–100, 2007.

Wang, T., Johnson, T., Zhang, J., and Huang, K. Topological methods for visualization and analysis of high dimensional single-cell rna sequencing data. In *BIOCOMPUTING 2019: Proceedings of the Pacific Symposium*, pp. 350–361. World Scientific, 2018.

Wang, Z. Exploration of topological data analysis in 3d printing. In *2020 International Conference on Information Science, Parallel and Distributed Systems (ISPDS)*, pp. 150–153. IEEE, 2020.

Wilkie, A. Model completeness results for expansions of the ordered field of real numbers by restricted pfaffian functions and the exponential function. *Journal of the American Mathematical Society*, 9(4):1051–1094, 1996.

Zechel, S., Zajac, P., Lönnerberg, P., Ibáñez, C. F., and Linnarsson, S. Topographical transcriptome mapping of the mouse medial ganglionic eminence by spatially resolved rna-seq. *Genome biology*, 15:1–12, 2014.

A. Gaussian cover assignment scheme

In this section, it is assumed that $\mathbb{X}_n$ is a point cloud in $\mathbb{R}^p$. Additionally, we consider r centers $\{c_1, ..., c_r\} \subseteq \mathbb{R}^p$ and r symmetric, semi-definite and positive matrices $\{\Sigma_1, ..., \Sigma_r\} \subseteq \mathbb{R}^{p\times p}$. For each $j \in \{1, ..., r\}$, consider the function:

$$\begin{aligned} q_j : \mathbb{R}^p &\longrightarrow [0, 1] \\ x &\longmapsto \exp\left(-(x - c_j)^T \Sigma_j^{-1}(x - c_j)\right). \end{aligned}$$

Define $A = (A_{i,j})_{\substack{1\leq i\leq n \\ 1\leq j\leq r}}$ to be a random variable in $\{0, 1\}^{n\times r}$ such that for every $(i, j) \in \{1, ..., n\} \times \{1, ..., r\}$:

$$A_{i,j} \mid \mathbb{X}_n \sim \mathcal{B}(q_j(x_i)),$$

and as before we take the $A_{i,j}$'s to be jointly conditionally independent.

This cover assignment scheme is similar to Gaussian mixture models, in that its realizations can be seen as a "one-hot encoding" of the latent variables in a mixture model. However, we can see that a realization of A can have more than one non-zero entry per line as opposed to a mixture model. Furthermore, mean and variance parameters can be inferred with an EM algorithm, and estimated proportions can be also involved in the definition of the q_j's.

Note that this strategy of defining a cover assignment scheme does not use a filter function or an overlapping cover entirely.

B. Elements of o-minimal geometry

Definition B.1. An o-minimal structure on the field of real numbers $\mathbb{R}$ is a collection $(S_n)_{n\in\mathbb{N}}$ where each S_n is a set of subsets of $\mathbb{R}^n$ that satisfies:

1. All algebraic subsets of $\mathbb{R}^n$ are in S_n;
2. S_n is a Boolean subalgebra of the powerset of $\mathbb{R}^n$ (i.e. stable by finite union, finite intersection and complementary);
3. if $A \in S_n$ and $B \in S_m$, then $A \times B \in S_{n+m}$;
4. if $\pi : \mathbb{R}^{n+1} \to \mathbb{R}^n$ is the linear projection onto the first n coordinates and $A \in S_{n+1}$ then $\pi(A) \in S_n$;
5. S_1 is exactly the family of finite unions of points and intervals.

The elementary example of an o-minimal structure is the collection of semi-algebraic sets. An element $A \in S_n$ for some $n \in \mathbb{N}$ is called a definable set. Furthermore, a map $f : A \to \mathbb{R}^m$ is called a definable map if its graph (i.e. $\{(x, f(x)) : x \in A\}$) is in S_{n+m}.

Definable maps are stable under addition, product and composition. A function that is coordinate-wise definable is also definable. Moreover, the result of (Wilkie, 1996) shows that there exists an o-minimal structure that contains the graph of the exponential function.

An important property of definable maps is that they admit a finite Whitney stratification. This means that if $f : A \to \mathbb{R}^m$ is definable with $A \in S_n$, then A can be decomposed into a finite union of smooth manifolds such that the restriction of f to each of these manifolds is a smooth function.

For more details on o-minimal geometry, see (Coste, 2000).

C. Proof of Theorem 6.2

Lemma C.1. *Let $\mathcal{S}$ be an o-minimal structure on $\mathbb{R}$. Assume that the two following conditions are satisfied.*

- *For every $x \in \mathbb{X}_n$, the function $\theta \in \mathbb{R}^s \mapsto f_\theta(x)$ is definable in $\mathcal{S}$ and is locally Lipschitz.*
- *For every $m \in \mathbb{N}$, the restriction of the persistence specific loss ℓ to the set of persistent diagrams of size m is definable in $\mathcal{S}$ and is locally Lipschitz.*

Then for every $e \in \{0,1\}^{n\times r}$, the function

$$\mathcal{L}_e \colon \theta \in \mathbb{R}^s \mapsto \mathcal{L}(e, f_\theta)$$

is definable in $\mathcal{S}$ and is locally Lipschitz.

Proof. Let $e \in \{0,1\}^{n\times r}$. Let $K = \text{MapComp}(e)$ with vertex set V. Remember that each vertex $c \in V$ is actually a subset of $\mathbb{X}_n$. We now define three maps to decompose the function $\mathcal{L}_e$. First, let us introduce the function

$$\begin{aligned} \text{VertexFilt}\colon \mathbb{R}^s &\longrightarrow \mathbb{R}^{|V|} \\ \theta &\longmapsto \left(\frac{\sum_{x\in c} f_\theta(x)}{\text{card}(c)}\right)_{c\in V}. \end{aligned}$$

For each coordinate of the function VertexFilt, that is for each $c \in V$, the function $\theta \longmapsto [\text{VertexFilt}(\theta)]_c$ is a linear combination of the functions $\theta \mapsto f_\theta(x)$. We can therefore see that it is definable in $\mathcal{S}$ and locally Lipschitz, by our first assumption.

Then we introduce

$$\begin{aligned} \text{SubFilt}\colon \mathbb{R}^{|V|} &\longrightarrow \mathbb{R}^{|K|} \\ \Phi &\longmapsto (\max_{c\in\sigma} \Phi_c)_{\sigma\in K}, \end{aligned}$$

and finally Persistence$\colon \mathbb{R}^{|K|} \longrightarrow \mathbb{R}^{|K|}$ that computes persistence for a filtration that acts on a fixed simplicial complex. The two functions SubFilt and Persistence are taken from (Carriere et al., 2021), where they are both proven to be definable in every o-minimal structure and Lipschitz.

Notice that:

$$\mathcal{L}_e = \ell \circ \text{Persistence} \circ \text{SubFilt} \circ \text{VertexFilt}.$$

Since e, and thus K, are fixed, ℓ can be replaced by its restriction to persistence diagrams of size $|K|$. Hence, following our second assumption, $\mathcal{L}_e$ is definable in $\mathcal{S}$ and locally Lipschitz. □

Recall the assumptions in Theorem 6.2 :

Suppose that there exists an o-minimal structure $\mathcal{S}$ and we have that:

- for every $x \in \mathbb{X}_n$, the function $\theta \mapsto f_\theta(x)$ is definable in $\mathcal{S}$ and is locally Lipschitz.
- for every $m \in \mathbb{N}$, the restriction of ℓ to the set of persistent diagrams of size m is definable in $\mathcal{S}$ and is locally Lipschitz.
- for every $e \in \{0,1\}^{n\times r}$, the function $\theta \mapsto \mathbb{P}_\theta(A = e|\mathbb{X}_n)$ is definable in $\mathcal{S}$ and is locally Lipschitz.

By Lemma C.1 and following the first two assumptions, we know that for every $e \in \{0,1\}^{n\times r}$, the function $\mathcal{L}_e \colon \theta \mapsto \mathcal{L}(e, f_\theta)$ is definable in $\mathcal{S}$ and is locally Lipschitz. Now, for every $\theta \in \mathbb{R}^s$:

$$\text{L}(\theta) = \sum_{e\in\{0,1\}^{e\times r}} \mathcal{L}(e, f_\theta) \cdot \mathbb{P}_\theta(A = e|\mathbb{X}_n).$$

As such, L is a sum of products between functions that are definable in $\mathcal{S}$ and locally Lipschitz. We conclude that L is itself definable in $\mathcal{S}$ and locally Lipschitz.

Note that the local Lipschitz property is stable by product (as opposed to the global Lipschitz property). This is due to the fact that the product of two Lipschitz and bounded functions is Lipschitz, and the fact that we can always limit the neighborhoods of points in $\mathbb{R}^s$ to bounded ones.

D. Technical conditions for Stochastic Gradient Descent

We are in the setting where we use stochastic gradient descent to minimize a function L. If we write the iterates of the algorithm as:

$$x_{k+1} = x_k - \alpha_k(y_k + \xi_k),$$

where

$$y_k \in \text{Conv}\left\{\lim_{z \to x_k} \nabla \text{L}(z) \ : \ \text{L is differentiable at } z\right\},$$

consider the following three conditions:

1. for any k, $\alpha_k \geq 0$, $\sum_{k=1}^{\infty} \alpha_k = +\infty$ and $\sum_{k=1}^{\infty} \alpha_k^2 < +\infty$;
2. $\sup_k \|x_k\| < +\infty$, almost surely;
3. Conditionally on the past, ξ_k must have zero mean and have a second moment that is bounded by a function $p\colon \mathbb{R}^s \longrightarrow \mathbb{R}$ which is bounded on bounded sets.

Note that the last condition is satisfied by taking a sequence of independent and centered variables with bounded variance, which are also independent of the past iterates $(x_k)_k$ and $(y_k)_k$.

According to (Davis et al., 2020), under these three conditions together with the condition that L is definable in an o-minimal structure and locally Lipschitz, then $(\text{L}(x_k))_k$ converges almost surely to a critical values and the limit points of $(x_k)_k$ are critical points of L.

E. Additional proof

Let $\mathcal{S}$ be the o-minimal structure presented in (Wilkie, 1996) containing the graph of the exponential function. Let $\delta > 0$, and A_δ be the smooth cover assignment scheme associated to the filter function f_θ and the cover $([a_j, b_j])_{1 \leq j \leq r}$. We prove that if the first assumption of Theorem 6.2 holds ($\mathcal{S}$ is the o-minimal structure in question), then the third assumption holds for A_δ. Consider, for each $j \in \{1, ..., r\}$, the function $u_j\colon \mathbb{R} \longrightarrow [0, 1]$ defined with:

$$x \mapsto \begin{cases} 1, & \text{if } x \in [a_j, b_j] \\ \exp(1 - 1/(1 - (\frac{a_j - x}{\delta})^2)), & \text{if } x \in (a_j - \delta, a_j] \\ \exp(1 - 1/(1 - (\frac{x - b_j}{\delta})^2)), & \text{if } x \in [b_j, b_j + \delta) \\ 0, & \text{otherwise} \end{cases}$$

u_j is definable in $\mathcal{S}$, this is because u_j is defined in a piecewise fashion from constant functions, and functions that are the composition of the exponential and rational functions.
Furthermore, u_j is infinitely differentiable and therefore locally Lipschitz. This is an example of a smooth bump function, see (Fry & McManus, 2002).
Now, notice that :

$$\mathbb{P}_\theta(A = e|\mathbb{X}_n) = \Pi_{\substack{1 \leq i \leq n \\ 1 \leq j \leq r}} [q_j(x_i) \cdot e_{i,j} + (1 - q_j(x_i)) \cdot (1 - e_{i,j})],$$

and

$$q_j(x_i) = u_j \circ f_\theta(x_i).$$

Definability and the local Lipshitz property are stable by composition and product. As such, the functions $\theta \mapsto \mathbb{P}_\theta(A = e|\mathbb{X}_n)$ are definable in $\mathcal{S}$ and locally Lipshitz.

F. Complexity analysis of Algorithm 1

The running time of each epoch in Algorithm 1 has three steps in practice:

1. computing the distribution of the assignment scheme $\mathbb{P}_\theta$,

2. computing several Mappers by sampling from $\mathbb{P}_\theta$,

3. evaluating the corresponding total persistence-based loss.

Let n be the number of points in the dataset, and r be the number of patches (i.e. elements in the latent cover of the dataset) in the Mapper cover (it is fixed and user-defined). Step 1 can be achieved for example in $O(n \times r)$ with Section 4.2's equations. Step 2 involves computing N sampled Mapper complexes. Every sampling can be done in $O(n \times r)$ time. Moreover, a single computation of a Mapper complex depends on the clustering method, and involves running it on every patch (for getting the Mapper nodes) and scanning through the points to detect clusters with non-empty intersection (for getting the Mapper simplices). Thus the complexity of this step is $O(N \times ((n \times r) + (\mathrm{Clus}(n) \times r + n)))$ (where $\mathrm{Clus}(n)$ is the complexity of the clustering method). Note that in practice, computing both the N Mappers and applying the clustering method to every patch (within a single Mapper computation) can be run in parallel. Step 3 involves computing N total persistence losses from the N Mapper complexes, which requires $O(N \times m^3)$ running time, where m is an upper bound on the number of simplices. This number of simplices depends linearly on the number of Mapper nodes (as we use persistence in degrees 0 and 1), which itself depends on the clustering method. If n_{clus} is an upper bound on the number of clusters, then m is typically of the order of $r \times n_{\mathrm{clus}}$.

In practice, the main bottleneck is running the clustering method on all patches, which has to be done N times. Note that when the clustering method depends on the pairwise distances, such as hierarchical clustering, these n^2 distances only need to be evaluated once. Our implementation can run these computations in parallel, which makes our code highly scalable. Boxplots of timings per epochs for our experiments can be found in Figure 9.

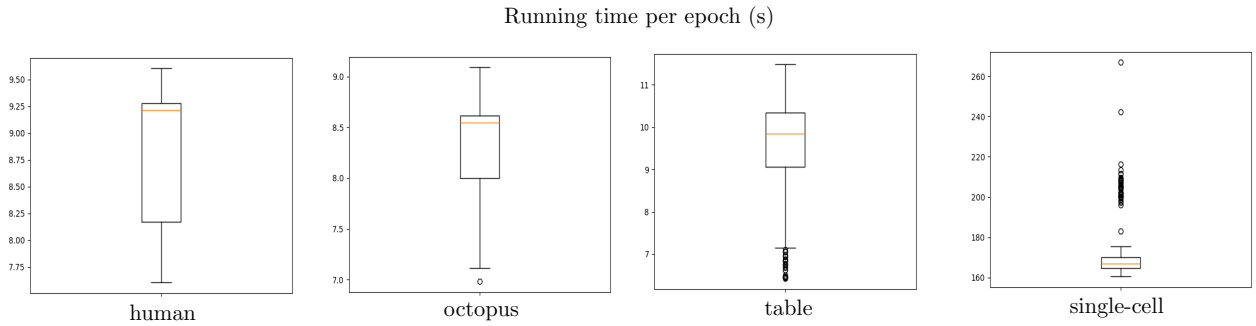

Figure 9. Boxplots of timings per epoch for the 3D shapes experiment and the human preimplantation (single cell) experiment.

G. Additional Figures and Tables for the experiments

Parameter	Human	Octopus	Table
Resolution	25	10	10
Gain	0.3	0.3	0.35
Number of clusters	3	8	8

Table 2. Resolution, gain and number of clusters parameters that are used to compute the Mapper for each 3-dimensional shape.

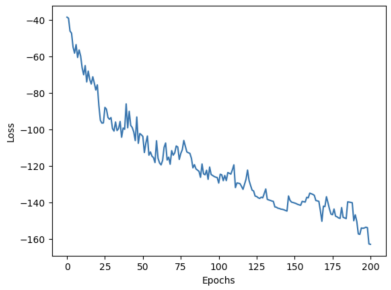

Figure 10. Learning curve for the human preimplantation dataset.

Filter	Correlation with time	P-value
Initial	0.1330	1.7596e-07
Optimized	-0.7549	4.0503e-282

Table 3. Pearson's correlation between the initial filter and time, and the optimized filter and time for the human preimplantation dataset. The associated p-values, obtained from testing the null hypothesis that the true correlation coefficient is zero, are also presented.

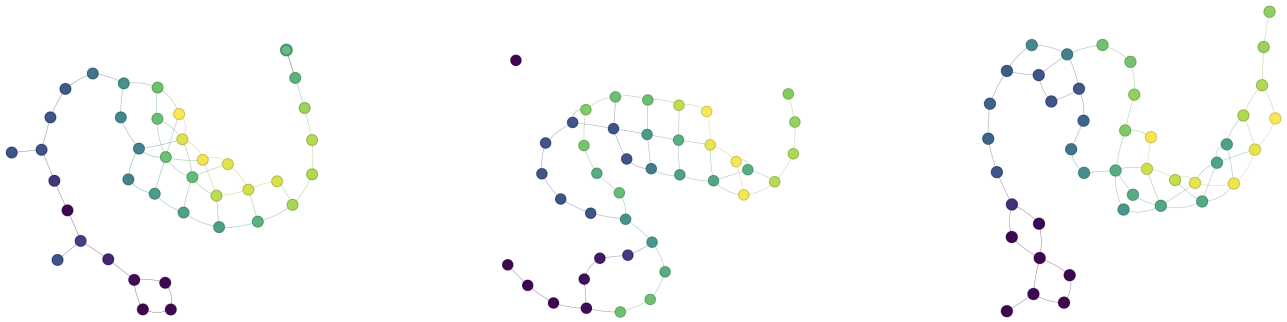

Figure 11. Regular Mapper graphs for the human preimplantation dataset, computed using optimized filter functions from different parameterized families: the linear family, the Gaussian RKHS linear family and the family of fully-connected neural networks with an architecture of two dense layers and ReLU activations, from left to right respectively.

H. Ablation study w.r.t the δ parameter

We re-run the single-cell experiment in Subsection 7.2 several times with different δ parameters and for each one we record the same clustering scores we used before. The study is summarized in Table 4. As expected, the smaller δ, the better the cosine distance with time of the filter is, as well as the clustering scores (even though the increase is not strictly monotonic). Indeed, when using larger δ values, points are more likely to belong to intervals that are far from their corresponding filter values, leading to Mapper complexes that tend to make less sense (while still being fit for gradient descent, i.e., definable and locally Lipschitz).

δ	10^{-1}	10^{-2}	10^{-2}	10^{-4}	10^{-5}
Corr.	0.428	-0.539	-0.724	-0.744	-0.751
Rand	0.0706	0.125	0.352	0.278	0.229
MI	0.141	0.178	0.422	0.368	0.391
Comp	0.309	0.228	0.469	0.407	0.57
FM	0.496	0.421	0.556	0.49	0.544

Table 4. Cosine distance w.r.t. time and clustering scores (Rand, Mututal Information, Completeness and Fowlkes-Mallow), comparing the ground truth clustering to the clusterings induced by the optimized Mapper graphs, for different values of δ.

I. Mouse embryonic fibroblasts reprogramming dataset

We consider the mouse embryonic fibroblasts (MEF) reprogramming dataset of (Schiebinger et al., 2019). It consists of $p = 19,089$ gene expressions for $251,203$ MEF cells, densely sampled across 18 days, with 39 individual timepoints. The experiment involves adding Doxorubicine (Dox) to the cells on day 0, withdrawing it at day 8, and then transferring them to either a serum-free N2B27 2i medium or maintaining them in serum.

Objective. We would, therefore, want to produce a representation, using our Soft Mapper optimization, that accounts for the time component (like in Section 7.2) and for the divergence in the treatment that the cells received at day 8. In order to achieve this, we first take a uniformly sampled subsample of the dataset of size $n = 1,500$ and we use the same preprocessing procedure as with the human preimplantation dataset. Similarly, we consider the same settings (linear family of filter functions, smooth cover assignment scheme, agglomerative clustering, diagonal initial parameter set and extended total persistence), and we run Algorithm 1 with $N = 300$ and $M = 10$. The learning curve is displayed in Figure 12.

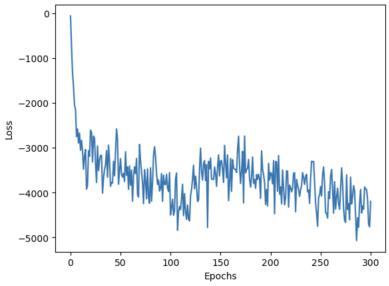

Figure 12. Learning curve for the MEF reprogramming dataset.

Qualitative assessment. By looking at the standard Mapper graphs corresponding to the initial and the optimized filter functions in Figure 13, one can see that the optimized Mapper graph represents the time component better and that it shows two major branches, which point to the two phases that appear in day 8 of the experiment. These observations are confirmed by the improvement in the Pearson's correlation coefficients with respect to time between the initial and the optimized filter function values in Table 5. We also color the optimized Mapper graph in Figure 14 using the three phases in the experiment (Dox, 2i and Serum), each mapped to a different color channel.

Figure 13. Classical Mapper graphs for the MEF reprogramming dataset computed using: in the left the initial filter function and in the right the optimized filter function. Vertices are colored using the mean value of the sampling timepoint in the corresponding clusters.

Filter	Correlation with time	P-value
Initial	-0.0560	2.9882e-02
Optimized	-0.4015	3.2090e-59

Table 5. Pearson's correlation between the initial filter and time, and the optimized filter and time. The associated p-values, obtained from testing the null hypothesis that the true correlation coefficient is zero, are also presented.

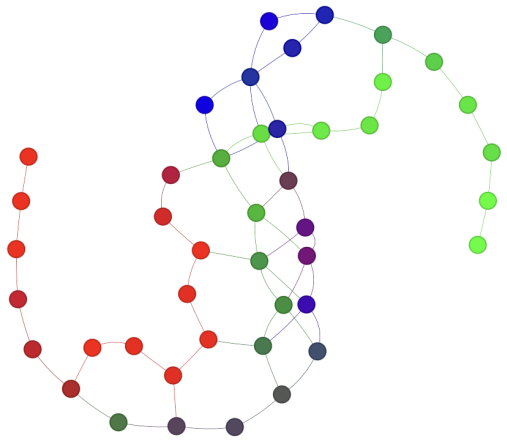

Figure 14. Standard Mapper graph computed using the optimized filter function, colored by mapping each phase to a color channel: Dox in green, Serum in blue and 2i in red.