CGL

Computational Geometric Learning

Persistence-Based Clustering in Riemannian Manifolds

Frédéric Chazal

Leonidas J. Guibas

Steve Oudot

Primoz Skraba

CGL Technical Report No.: 21

Part of deliverable:	WP-WP1/
Site:	INRIA Saclay
Month:	12

Project co-funded by the European Commission within FP7 (2010–2013)
under contract nr. IST-25582

Persistence-Based Clustering in Riemannian Manifolds

Frédéric Chazal* and Leonidas Guibas† and Steve Oudot ‡ and Primoz Skraba §

October 27, 2011

Abstract

We present a clustering scheme that combines a mode-seeking phase with a cluster merging phase in the corresponding density map. While mode detection is done by a standard graph-based hill-climbing scheme, the novelty of our approach resides in its use of *topological persistence* to guide the merging of clusters. Our algorithm provides additional feedback in the form of a set of points in the plane, called a *persistence diagram* (PD), which provably reflects the prominences of the modes of the density. In practice, this feedback enables the user to choose relevant parameter values, so that under mild sampling conditions the algorithm will output the *correct* number of clusters, a notion that can be made formally sound within persistence theory.

The algorithm only requires rough estimates of the density at the data points, and knowledge of (approximate) pairwise distances between them. It is therefore applicable in any metric space. Meanwhile, its complexity remains practical: although the size of the input distance matrix may be up to quadratic in the number of data points, a careful implementation only uses a linear amount of memory and takes barely more time to run than to read through the input.

In this conference version of the paper we emphasize the experimental aspects of our work, describing the approach, giving an intuitive overview of its theoretical guarantees, discussing the choice of its parameters in practice, and demonstrating its potential in terms of applications through a series of experimental results obtained on synthetic and real-life data sets. Precise statements and proofs of our theoretical claims can be found in the full version of the paper [7].

Categories and Subject Descriptors: F.2.2 [Analysis of Algorithms and Problem Complexity]: Non-numerical Algorithms and Problems — Geometrical problems and computations, Computations on discrete structures.

General Terms: Algorithms, Theory.

Keywords: Unsupervised Learning, Clustering, Mode Seeking, Topological Persistence, Morse Theory.

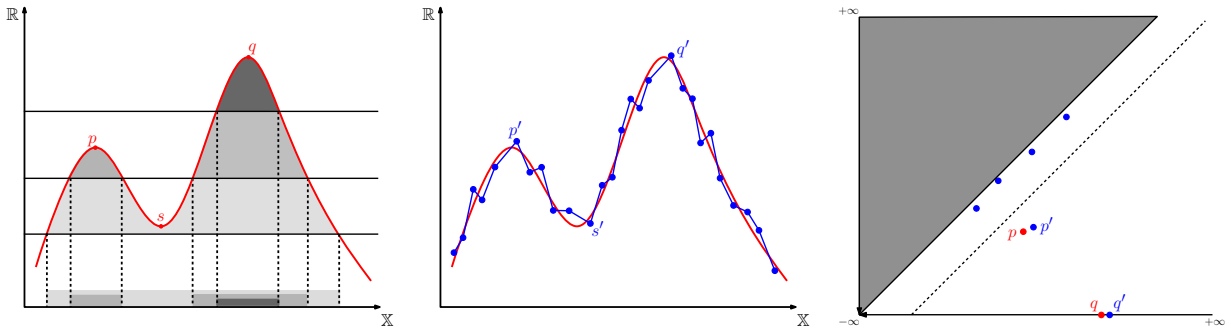
1 Introduction

Unsupervised learning or clustering is an important tool for understanding and interpreting data in a variety of fields. Obtaining the most natural clustering is an ill-posed problem in general, and it is particularly difficult with massive and high-dimensional data sets where visualization techniques

*INRIA Saclay
†Stanford University
‡INRIA Saclay
§INRIA Saclay

fail. The breadth of the existing work on clustering [17] shows the high interest this topic has aroused among the scientific community. Here we recount a few classical methods to show where our approach stands with respect to the literature:

K-means [21] is perhaps the most commonly used approach. Given a fixed number k of clusters, it tries to place cluster centers and define cluster boundaries so as to minimize the sum of the squared distances to the center within each cluster. This minimization problem is known to be NP-hard, so k-means resorts to an iterative expectation-maximization procedure that is guaranteed to converge at least to some local minimum. This minimum is not guaranteed to be global, however. Another issue with k-means and its variants is that they produce bad results on highly non-convex clusters.



(a) (b) (c)

Figure 1: Sketch of topological persistence: (a) a new connected component is born in the superlevel-set F^α when $\alpha = f(p)$, and it dies when $\alpha = f(s)$; its lifespan is represented as a vertical segment on the left of the graph, or equivalently as a point in the PD of f; (b) a piecewise-linear approximation $\tilde{f}$ of f; (c) superimposition of the PDs of f (red) and $\tilde{f}$ (blue), showing the one-to-one correspondence between the prominent peaks of f and $\tilde{f}$.

Spectral clustering [28] was designed specifically to work on non-convex data. It first computes an embedding of the data set endowed with a diffusion distance between the points, given by a Laplacian of some neighborhood graph. Then, it applies the standard k-means method in the new ambient space. Computing the embedding requires an eigendecomposition of the Laplacian, which may have numerical issues as the size of the data grows. The presence of a gap in the spectrum of the Laplacian gives an indication of the correct number k of clusters. However, problems arise when there are more than a small number of outliers in the data, in which case no such gap may exist.

Density-based techniques make the assumption that the data points are drawn from some unknown density function f. Clustering becomes then a problem of understanding the structure of f, as estimated from the samples. A popular approach consists in thresholding the density at some fixed level α, then treating the connected components of the superlevel-set $F^\alpha = f^{-1}([\alpha, +\infty))$ as clusters and the rest of the data as noise. In practice, the density f is unknown so its superlevel F^α needs to be approximated from the data, which algorithms like DBSCAN [15, 23] do by various graph-based heuristics. Unfortunately, due to the use of a fixed density threshold α, these techniques do not respond well to hierarchical data sets, in which subtle multi-scale clustering phenomena may occur.

Another popular approach, called *mode-seeking*, consists in detecting the local peaks of f in order to use them as cluster centers and to partition the data according to their *basins of attraction*.

The precise notion of the basin of attraction B_p of a peak p varies between references, yet the bottom line remains that B_p corresponds to the subset of the data points that eventually reach p by some greedy hill-climbing procedure. This line of work started with the algorithm of Koontz *et al.* [20] and was followed by numerous variants and extensions, including Mean-Shift [13] and its successors [24, 27]. A common issue faced by these techniques is that the gradient and extremal points of a density function are notoriously unstable, so their approximation from a density estimator can lead to unpredictable results. This is why methods such as Mean-Shift adopt a proactive behavior by smoothing the estimator before launching the hill-climbing procedure, which in turn raises the difficult question of how much smoothing is needed to remove the noise without affecting the signal, and to obtain the correct number of clusters.

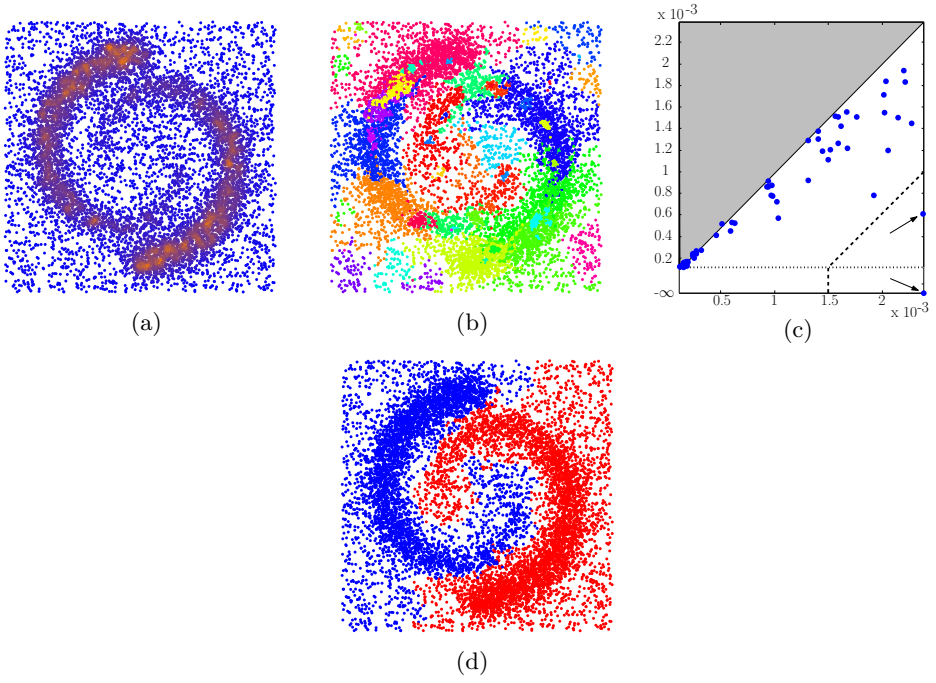


Figure 2: Our approach in a nutshell: (a) estimation of the underlying density function f at the data points; (b) result of the basic graph-based hill-climbing step; (c) approximate PD showing 2 points far off the diagonal corresponding to the 2 prominent peaks of f; (d) final result obtained after merging the clusters of non-prominent peaks.

Enter topological persistence In this paper we adopt a more reactive behavior by using *topological persistence* [14, 29] to detect and merge unstable clusters after their computation, thus regaining some stability. Although our method belongs to the same family as Mean-Shift, the use of persistence makes it possible to link explicitly the input parameter values to the output number of clusters. It also provides a sound theoretical framework for characterizing the correct number of clusters, in the same spirit as spectral clustering.

Topological persistence estimates the *prominence* of the density peaks and builds a hierarchy of the peaks based on it. The prominence of a peak is defined as the difference between its height and the level at which its basin of attraction meets the one of a higher peak (its parent in the hierarchy). More precisely, focusing on the 1-parameter family of superlevel-sets $F^\alpha = f^{-1}([\alpha, +\infty))$ of the

density function f, persistence studies the evolution of the connectivity of F^α as α ranges from $+\infty$ to $-\infty$. A new connected component C is born in F^α when α reaches the height of a peak p of f, and dies when it gets connected in F^α to the component of a higher peak (see Figure 1(a)). As mentioned above, the prominence of p is simply the height difference between birth and death values of C. The lifespan of each connected component C can be represented as a point in the plane, with the x-coordinate giving the birth time of C and the y-coordinate giving its death time. The collection of such points is called the *persistence diagram* (PD) of f, illustrated in Figure 1(c). The key insight of this planar data representation is that the PD reveals part of the topological structure of the density function f. More precisely, each peak of f is uniquely represented by one point in the PD, and its prominence is given by twice the distance of this point to the diagonal $y = x$.

Originally defined in *Morse theory*, prominence is known to be more stable than other measures of significance such as absolute height. For example, a small bump occurring at a high density will have large absolute height but small prominence. The same kind of stability holds for PDs. For instance, f and its noisy approximation $\tilde{f}$ (see Figure 1(b)) have similar PDs, in the sense that there is a one-to-one mapping of small amplitude from the prominent peaks of $\tilde{f}$ to the ones of f, the rest of the peaks being treated as noise and mapped to the diagonal in the PD (see Figure 1(c)). Thanks to this fundamental stability property, with only limited knowledge of the underlying space and a finite estimate of the density f it is possible to *provably* and *efficiently* approximate the PD of f. The combination of such guarantees with computational practicality is at the heart of topological data analysis [1, 2, 3, 16], which includes this work.

It is worth noting that PDs are similar in spirit to the *dendrograms* provided by agglomerative clustering schemes, whose principle is to build the clusters in a bottom-up fashion, starting with each point being its own cluster and merging at each step the most similar clusters together. The output dendrogram describes the sequence of merges that have occurred during the process, thus encoding the hierarchical structure of the obtained family of clusterings. While these techniques bear some connections with ours, they are actually based on a different clustering paradigm that suffers from its own limitations — see e.g. Section 14.3.12 in [18].

Our method Our clustering scheme, called ToMATo (*Topological Mode Analysis Tool*), combines the original graph-based hill-climbing algorithm of Koontz *et al.* [20] with a cluster merging step guided by persistence. As illustrated in Figure 2(b), hill-climbing is very sensitive to perturbations of the density function f that arise from a density estimator $\tilde{f}$. Computing the PD of $\tilde{f}$ enables us to separate the prominent peaks of $\tilde{f}$ from the inconsequential ones. In Figure 2(c) for instance, we can see 2 points (pointed to by arrows) that are clearly further from the diagonal than other points: these correspond to the 2 prominent peaks of $\tilde{f}$ (one of the points is at $y = -\infty$, as the highest peak never dies). To obtain the final clusters, we merge every cluster of prominence less than a given thresholding parameter τ into its parent cluster in the persistence hierarchy. As shown in Figures 2(c) and 2(d), the PD gives us a precise understanding of the relationship between the choice of τ and the number of obtained clusters.

In practice we run ToMATo twice: in the first run we set $\tau = +\infty$ to merge all clusters and thus compute the PD; then, using the PD we choose a value for τ (which amounts to selecting the number of clusters) and re-run the algorithm to obtain the final result. The feedback provided by the PD proves invaluable in interpreting the clustering results in many cases. Indeed, the PD gives a clear indication of whether or not there is a natural number of clusters, and because it is a planar

point cloud we can understand its structure visually, regardless of the dimensionality of the input data set.

ToMATo is highly generic and agnostic to the choice of distance, underlying graph, and density estimator. Our theoretical guarantees make use of graphs that do not require the geographic coordinates of the data points at hand (only pairwise distances are used) nor estimates of the density at extra points. This makes the algorithm applicable in very general settings. ToMATo is also highly efficient: in the worst case it has an almost-linear running time in the distance matrix size, and only a linear memory usage in the number of data points (as in most applications distances to near-neighbors suffice). Most often we use Euclidean distances, however other metrics such as diffusion distances can be used. Indeed, the choice of metric and density estimator define the space we study, while our algorithm gives the structure of this space. Finally, ToMATo comes with a solid mathematical formulation. We show that, given a finite sampling of an *unknown* space with pointwise estimates of an *unknown* density function f, our algorithm computes a faithful approximation of the PD of f. Under conditions of a sufficient *signal-to-noise* ratio in this PD, we can determine the correct number of clusters and show that significant clusters always have stable regions. The precise statements and proofs are included in the full version of the paper [7]; in this conference version we give an intuitive explanation to highlight how interpretation can be aided by our approach. In some applications, the number of clusters is not obvious and we see this in the corresponding PDs. However, in these cases the relationship between the choice of parameters and the number of obtained clusters is transparent.

Obtaining guarantees in such general settings using only simple tools like neighborhood graphs is made possible by recent advances on the stability of persistence diagrams [4, 6]. Previous stability results [12] required the use of piecewise-linear approximations of the density functions, as in Figure 1(b) for instance. The construction of such approximations becomes quickly intractable when the dimensionality of the data grows. This fact might explain why topological persistence was never really exploited in mode analysis before, except in some restricted or low-dimensional settings [22].

2 The algorithm

We first provide an intuitive insight into our approach by considering the continuous setting underlying our input. We then give the details of the algorithm in the discrete setting.

The continuous setting Consider an m-dimensional Riemannian manifold $\mathbb{X}$ and a Morse function $f : \mathbb{X} \to \mathbb{R}$, *i.e.* a C^2-continuous function with non-degenerate critical points. Assume that f has a finite number of critical points. The *ascending region* of a critical point m, noted $A(m)$, is the subset of the points of $\mathbb{X}$ that eventually reach m by moving along the flow induced by the gradient vector field of f. For all $x \in A(m)$, we call m the *root* of x. Ascending regions of the peaks of f are known to form pairwise-disjoint open cells homeomorphic to $\mathbb{R}^m$. Furthermore, assuming $\mathbb{X}$ to have no boundary and f to be bounded from above and proper[1], the ascending regions of the peaks of f cover $\mathbb{X}$ up to a subset of Hausdorff measure zero. It is then natural to use them to partition (almost all) the space $\mathbb{X}$ into regions of influence.

For any $\alpha \in \mathbb{R}$, let $\mathbb{F}^\alpha$ denote the closed superlevel-set $f^{-1}([\alpha, +\infty))$. Consider the nested family of spaces $\{\mathbb{F}^\alpha\}_{\alpha\in\mathbb{R}}$ obtained by letting parameter α decrease from $+\infty$ to $-\infty$. This family

[1]This means that for any bounded closed interval $[a, b] \subset \mathbb{R}$, the pre-image $f^{-1}([a, b])$ is a compact subset of $\mathbb{X}$.

is called the *superlevel-sets filtration* of f. For any $\alpha \in \mathbb{R}$ and $x \in \mathbb{X}$, let $C(x, \alpha) \subseteq \mathbb{F}^\alpha$ denote the path-connected component of $\mathbb{F}^\alpha$ that contains x. Morse theory tells us that when a local maximum m_p of f enters the superlevel-sets filtration, at time $\alpha = b(m_p)$, a new path-connected component $C(m_p, \alpha)$ appears in the superlevel-set $\mathbb{F}^\alpha$. In homological terms, the peak m_p is called the *generator* of the component born at time $b(m_p)$. $C(m_p, \alpha)$ ceases to be an independent connected component in $\mathbb{F}^\alpha$ when it gets connected to another component generated by a higher peak m_q. At that particular time $\alpha = d(m_p)$, persistence theory tells us that the component $C(m_p, \alpha)$ gets *merged* into $C(m_q, \alpha)$. While m_q remains the generator of the component $C(m_q, \alpha)$, m_p ceases to be a generator, and by analogy we call m_q its root, noted $m_q = r(m_p)$. In the 0-dimensional persistence diagram $\mathrm{D}_0 f$, the lifespan of m_p as a generator is encoded by the point p of coordinates $p_x = b(m_p)$ and $p_y = d(m_p) \leq p_x$. The difference $d_p = p_x - p_y \geq 0$ between birth and death times is called the *prominence* of the peak m_p. Equivalently, we say that m_p is *d_p-prominent*.

Given a thresholding parameter $\tau \geq 0$, we restrict our focus to the peaks m_p of f of prominence at least τ. Intuitively, the points of $\mathbb{X}$ that are *attracted* by m_p are the ones belonging to ascending regions that are eventually merged by persistence into the connected component of m_p before being merged into the component of any other peak of prominence at least τ. Formally, for every peak m_q of f (of arbitrary prominence), let us iterate the *root map* $m_q \mapsto r(m_q)$ until some peak of prominence at least τ is reached[2]. We call r_τ^* the thus iterated root map, and we note that every peak of prominence at least τ is a fixed point of r_τ^*. The union of the ascending regions of the peaks mapped to m_p through r_τ^* is referred to as *the basin of attraction of* m_p (of parameter τ) in the paper, noted $B_\tau(m_p)$:

$$\forall m_p \text{ s.t. } p_x - p_y \geq \tau,\ B_\tau(m_p) = \bigcup_{r_\tau^*(m_q) = m_p} A(m_q).$$

Note that $B_\tau(m_p)$ contains $A(m_p)$ since m_p is a fixed point of r_τ^*. More precisely, we have $A(m_p) = B_0(m_p) \subseteq B_\tau(m_p)$. In addition, since the iterated root map $m_q \mapsto r_\tau^*(m_q)$ is uniquely defined, the basins of attraction form a partition of the union of all ascending regions. These basins are our target clusters.

The discrete setting ToMATo takes as input an unweighted simple graph G, whose vertex set represents the data points and whose edges connect the points according to some user-defined proximity rule. Each vertex i of G must be assigned a real value $\tilde{f}(i)$ corresponding to the estimated density at that point. In addition, ToMATo takes in a non-negative *merging* parameter τ, whose choice and use are elaborated below. In this discrete setting, the algorithm mimics the process described above in the continuous setting by running the following procedures in this order (in practice they can be run in a single pass over the vertices of G):

1. (Mode-seeking) To compute the initial clusters, ToMATo iterates over the vertices of G sorted by decreasing $\tilde{f}$-values : at each vertex i, it simulates the effect of the gradient of the underlying density function by connecting i to its neighbor in G with highest $\tilde{f}$-value. If all neighbors of i have lower values, then i is declared a peak of $\tilde{f}$. The resulting collection of pseudo-gradient edges forms a spanning forest of the graph, and each tree in this forest can be viewed as the analog within G of the ascending region of a peak of the true density function in the underlying continuous domain.

[2]Such a prominent peak is always reached eventually, since the function f has finitely many peaks and since the root map satisfies $f(m_q) < f(r(m_q))$, meaning that $r(m_q)$ is more prominent than m_q.

2. (Merging) To handle merges between trees, ToMATo iterates over the vertices of G again, in the same order, while maintaining a union-find data structure, where each entry corresponds to a union of trees of the spanning forest. We call *root* of an entry e, or $r(e)$ for short, the vertex contained in e whose $\tilde{f}$-value is highest. By definition, this vertex is the root of one of the trees contained in e. During the iteration process, two different scenarios may occur when a vertex i is considered: (a.) If i is a peak of $\tilde{f}$ within G, *i.e.* the root of some tree T_i, then i creates a new entry e_i in the union-find data structure, in which T_i is stored. (b.) If i is not a peak and therefore belongs to some tree T_j stored in an existing entry e_j, then we look for potential merges of e_j with other entries. Specifically, iterating (in an arbitrary order) over the neighbors k of i in G satisfying $\tilde{f}(k) \geq \tilde{f}(i)$, we merge e_k into e_j if $\tilde{f}(r(e_k)) < \min\{\tilde{f}(r(e_j)),\ \tilde{f}(i)+\tau\}$, in other words if the peak $r(e_k)$ is lower than $r(e_j)$ and has prominence less than τ. Reciprocally, we merge e_j into e_k if $\tilde{f}(r(e_j)) < \min\{\tilde{f}(r(e_k)),\ \tilde{f}(i)+\tau\}$.

The output of ToMATo is the collection of (merged) clusters stored in the entries of the union-find data structure at the end of the merging phase. In addition, ToMATo outputs the lifespans of all the entries created during the process, each lifespan being represented as a point in the plane. When τ is set to $+\infty$, this planar set of points coincides with the PD of the scalar field $\tilde{f}$ over the graph G.

3 Parameter selection

ToMATo takes in three inputs: the neighborhood graph G, the density estimator $\tilde{f}$, and the merging parameter τ. Although the freedom left to the user in the choice of these inputs gives our approach a lot of flexibility, the latter must not come at the expense of a significant increase in the amount of effort needed to run the program. This is why this section provides some insights into the choice of our parameters.

Neighborhood graph G ToMATo relies heavily on the neighborhood information encoded in the input graph G. Choosing a relevant neighborhood graph (and thereby a relevant metric) is a problem faced by many clustering techniques. In our experiments we primarily used the *δ-Rips graph*, which connects two data points whenever they lie within distance δ of each other. This purely metric definition makes it possible to use these graphs in arbitrary metric spaces, and to interpret the structure of the obtained PDs thanks to a sound theoretical framework (see Section 4). The choice of a particular value for δ corresponds more or less to the choice of a scale at which to inspect the data. It can be tricky on some instances, where different choices of scale may reveal different structures. This is why we recommend running ToMATo at several scales, either sequentially or in parallel. This can be done even for large data sets thanks to the efficiency of the algorithm. For too large values of δ there will be no real structure in the PD, while too small values of δ will produce many infinitely prominent peaks in the PD, corresponding to the connected components of the graph. By examining the PDs obtained at different scales, one can find an appropriate trade-off.

Another popular choice of neighborhood graph is the k-nearest neighbor (k-nn) graph. Its main advantage is that it remains sparse whatever the layout of the data. We tested the algorithm with this graph and generally found that it performed well, recovering the correct clusters under a suitable choice of parameter k. However, to the best of our knowledge there currently exists no theory that validates these empirical observations, and in practice we were left with the task of choosing k, which we accomplished by trial-and-error.

We also ran ToMATo using Delaunay graphs and some of their variants [26]. These have the great advantage of being parameter-free, and the disadvantage of creating long edges connecting high-density areas that are far apart, thus leading to artificial merges between clusters. One way around this issue is to discretize the long edges and to estimate the density at the newly created nodes, in order to reveal additional valleys that separate the prominent peaks. This requires the ability to estimate the density outside the input point cloud, which is generally the case when a Delaunay graph is built.

Density estimator $\tilde{f}$ While the algorithm is agnostic to the choice of density estimator, we experimented with two of them: a truncated Gaussian kernel estimator, and the *distance to a measure* proposed in [5]. Each of these estimators uses one parameter, and we refer the reader to the appropriate references for some insights into the choice of these parameters.

Merging parameter τ During the merging phase, ToMATo eventually merges all clusters of prominence less than τ into clusters of prominence at least τ. In other words, the choice of τ determines which peaks of $\tilde{f}$ are considered real and which peaks are treated as noise. To choose τ, we run ToMATo twice. In the first run, τ is set to $+\infty$, which makes ToMATo output the PD of the scalar field $\tilde{f}$ over the graph G, just as the 0-dimensional version of the standard persistence algorithm [14] would do. This PD reveals the topological structure of $\tilde{f}$, providing the height and prominence of each peak of $\tilde{f}$. Hence it can be used to determine a suitable value for τ, to be assigned in a second run of ToMATo that computes the final clustering.

In cases where the PD of $\tilde{f}$ shows a large gap separating a small set of k highly prominent peaks from the rest of the structure, we infer that the number of clusters is likely to be k, and so we set τ to be any value between the prominences of the k distinguished peaks and the prominences of the rest of the PD. Then the output of the second run of ToMATo contains exactly k clusters. Detecting a large gap automatically can be done by means of the following simple heuristic: we sort the points in the PD by decreasing prominence (possibly weighted by the corresponding peak heights, to avoid a squeezing effect due to the presence of extremely or even infinitely prominent peaks), and then we look for the largest drop in the sequence of (weighted) prominences. This is reminiscent of what is commonly done in spectral clustering for finding a gap in a Laplacian spectrum, and in fact our prominence gap and the spectral gap play very similar roles, even if in completely different settings.

In cases where the PD of $\tilde{f}$ does not show any well-separated structure, it still provides a clear relationship between the choice of parameter τ and the number of clusters obtained after re-running ToMATo. The choice of a particular value (or of a collection of values) for τ depends on the context, and in practice it requires to use additional application-specific information on the data. This is what we did for instance on the biological data set to distinguish between several possible choices of τ (see Section 5).

4 Theoretical guarantees.

In this section we give an intuitive overview of the theoretical guarantees that come along with ToMATo and validate the above heuristics. Formal statements and proofs can be found in the full version of the paper [7].

Let $\mathbb{X}$ be an m-Riemannian manifold with positive convexity radius, and $f : \mathbb{X} \to \mathbb{R}$ a Lipschitz-continuous probability density function with respect to the m-dimensional Hausdorff measure. We assume that the input data set P has been sampled over $\mathbb{X}$ according to f in i.i.d. fashion, and that the values of f at the data points and the geodesic distances between the data points are known either exactly or within a small additive error. Finally, we assume the input graph G to be the δ-Rips graph built over P using the estimated geodesic distances, for some user-defined parameter δ.

Definition 4.1 *Given two values $d_2 > d_1 \geq 0$, the persistence diagram $\mathrm{D}_0 f$ is called (d_1, d_2)-*separated *if every point of $\mathrm{D}_0 f$ lies either in the region D_1 above the diagonal line $y = x - d_1$, or in the region D_2 below the diagonal line $y = x - d_2$ and to the right of the vertical line $x = d_2$.*

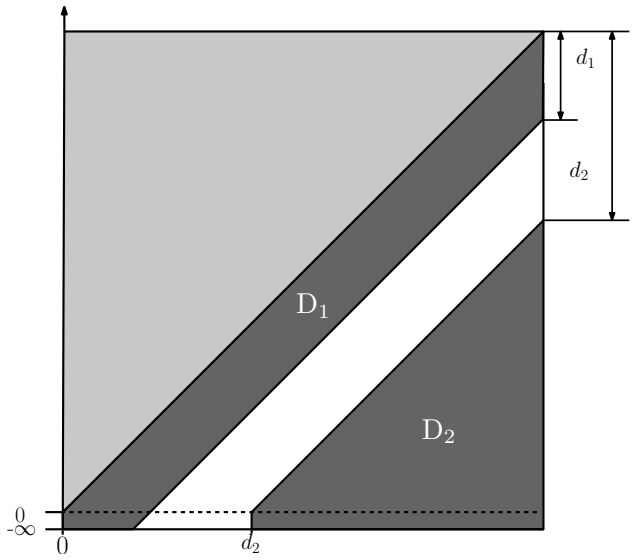


Figure 3: Separation of the persistence diagram $\mathrm{D}_0 f$ between prominent peaks (D_2) and topological noise (D_1).

This condition, illustrated in Figure 3, formalizes the intuitive notion that the points of $\mathrm{D}_0 f$ can be separated between prominent peaks (region D_2) and topological noise (region D_1), as illustrated opposite. In this respect, it acts very similarly to a signal-to-noise ratio condition: the larger the prominence gap $d_2 - d_1$, the more clearly one can separate the prominent peaks from the noise. In the limit scenario where $d_1 = 0$, all peaks of f are at least d_2-prominent and none of them is viewed as noise. The additional condition that the points of D_2 must lie to the right of the vertical line $x = d_2$ is purely technical and stems from the fact that only some superlevel-set of the density f can be densely sampled by the data points, not the whole manifold $\mathbb{X}$. Our first result relates the number of clusters computed by the algorithm to the number of prominent peaks of f. Using the stability of persistence diagrams [4] to relate the diagram of f to the diagram output by step 2 of the algorithm, we can prove that the regions D_1 and D_2 remain disjoint under perturbations caused by our approximation, and can therefore be separated using any value within a certain range for the thresholding parameter τ. With such values of parameter τ as input, the algorithm computes the correct number of clusters with high probability:

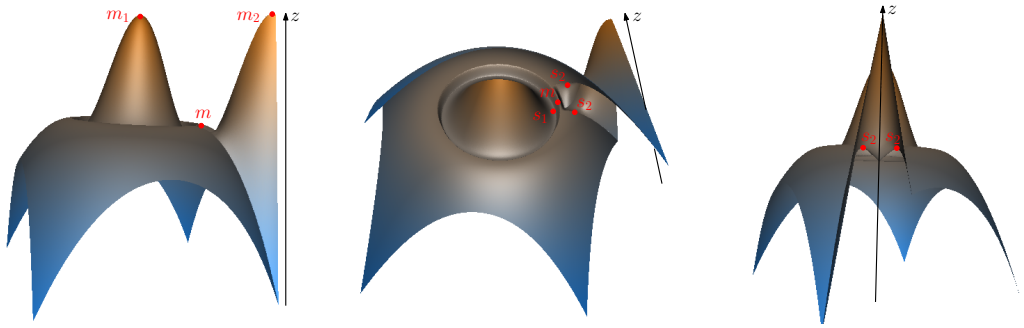


Figure 4: A function $f : [0,1]^2 \to \mathbb{R}$ with unstable basins of attraction. The three peaks m, m_1, m_2 have respective prominences $f(m)-f(s_2)$, $f(m_1)-f(s_1)$, and $+\infty$. When $\tau > f(m)-f(s_2)$, the ascending region $A(m)$ is merged into the basin of attraction $B_\tau(m_2)$ at the value $\alpha = f(s_2)$. However, since $f(s_2) - f(s_1)$ can be made arbitrarily small compared to $f(m_1) - f(m)$, arbitrarily small perturbations of f compared to the prominence gap $f(m_1) - f(m) + f(s_2) - f(s_1)$ merge $A(m)$ into $B_\tau(m_1)$ instead, thus making $A(m)$ an unstable part of $B_\tau(m_2)$. In the discrete setting, where the square $[0,1]^2$ is replaced by a point cloud, different samplings of the square or different values of parameter δ lead to different merges of the cluster associated with m. This erratic behavior of the algorithm only stops when δ become small enough compared to the (arbitrarily small) quantity $f(s_2) - f(s_1)$.

Result 1 (Theorem 4.8 in the full version [7])
If $\mathrm{D}_0 f$ *is* (d_1, d_2)*-separated and if the Rips parameter* $\delta > 0$ *is smaller than a fraction of* $d_2 - d_1$ *and of the convexity radius of* $\mathbb{X}$*, then there is a range* $[d_1 + O(\delta),\ d_2 - O(\delta)]$ *of values of the thresholding parameter* τ *such that the number of clusters computed by the algorithm is equal to the number of peaks of* f *of prominence at least* τ *with probability at least* $1 - e^{-\Omega(n)}$*, where* n *is the number of data points.*

Explicit bounds are given in the full statement of the theorem. The big-O notation hides some factors depending on the Lipschitz constant c of f, while the big-Ω notation hides some factors depending on c, δ, and certain geometric quantities of the manifold $\mathbb{X}$. As can be seen from the statement, the larger the prominence gap $d_2 - d_1$, the larger the interval of admissible values for τ, and of course the more easily this interval can be detected. In the meantime, the smaller δ, the larger the interval, but also the more points need to be sampled, simply because a minimum point density is required for the connectivity of the δ-Rips graph to reflect the one of some superlevel-set of the density f.

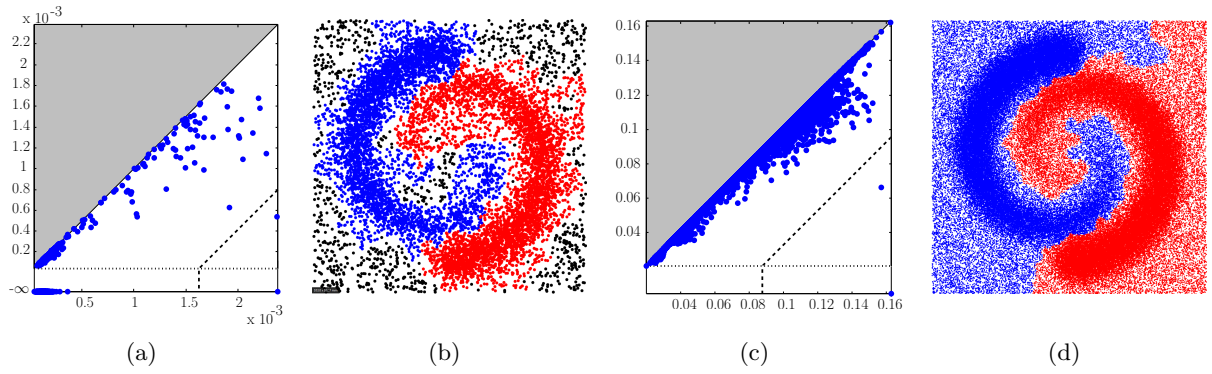
Another question is how well the output of the algorithm approximates the basins of attraction of the prominent peaks over the point cloud, assuming that f is of Morse type. In full generality, this is a hopeless question since the basins of attaction are not stable even in the smooth case. There are indeed many examples of very close functions having very different basins of attraction, and clearly the algorithm cannot provably-well approximate the unstable parts of the basins. An illustrative example is given in Figure 4. Yet, we can ensure that the output of the algorithm approximates some stable parts of the basins:

Result 2 (Theorem 4.9 in the full version [7])
Under the same hypotheses as in Result 1, it holds with probability at least $1 - e^{-\Omega(n)}$ *that for every point* $p \in \mathrm{D}_2$ *the algorithm outputs a cluster* C *such that* $C \cap \mathbb{F}^\alpha = B_\tau(m_p) \cap P \cap \mathbb{F}^\alpha$ *for all values* $\alpha \in [\alpha_\tau(m_p) + d_1 + O(\delta),\ f(m_p))$*, where* m_p *is the peak of* f *corresponding to point* p*, where*

$B_\tau(m_p)$ denotes the basin of attraction of m_p in the underlying manifold $\mathbb{X}$, and where $\alpha_\tau(m_p)$ is the first value of α at which $B_\tau(m_p)$ gets connected to the basin of attraction of another peak of f of prominence at least τ in the superlevel-set $\mathbb{F}^\alpha$.

In plain words, cluster C is the *trace* of the basin of attraction $B_\tau(m_p)$ over the point cloud P, until (approximately) the value $\alpha_\tau(m_p)$ at which $B_\tau(m_p)$ meets the basin of another τ-prominent peak of f. Beyond that value, the cluster may start diverging from the basin, which itself may start being unstable, as illustrated in Figure 4. We show in the full version of the paper (Eq. 7 in [7]) that $\alpha_\tau(m_p) \leq f(m_p) - d_2$, so the length of the interval of values of α for which C is the trace of $B_\tau(m_p)$ over P is at least $d_2 - d_1 - O(\delta)$.

Our proof of Result 2 also shows an important fact, namely: that each basin of attraction $B_\tau(m_p)$ is stable under small perturbations of the function f, at least between values $f(m_p)$ and $\alpha_\tau(m_p) + d_1 + O(\delta)$. This fact opens the door to a more statistical approach to clustering: since we know the top parts of the basins (and therefore of the clusters computed by the algorithm) are stable under small perturbations of the function, we can conduct multiple runs of the algorithm with random perturbations of the function, and then find correspondences between the outputs of different runs. Each point can then be assigned a quantitative measure of its classification stability over the runs.



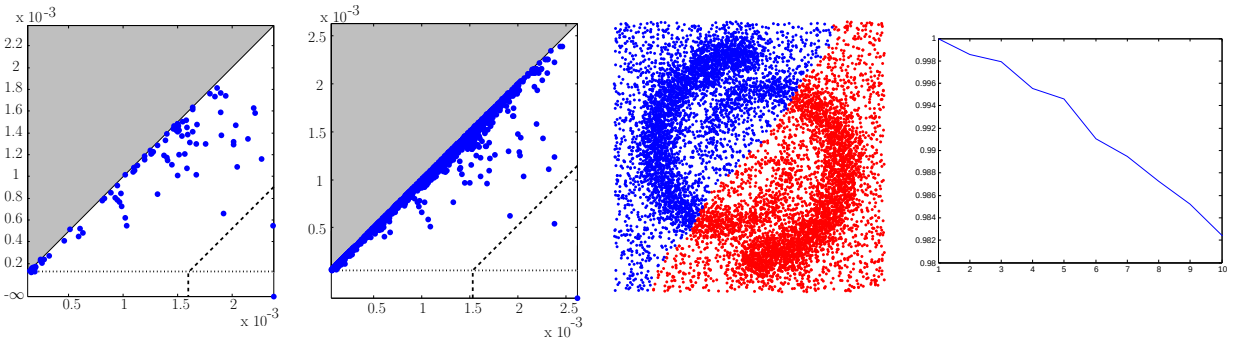
(a) (b) (c) (d)

Figure 5: Left: the spirals data set from Figure 2, processed using a smaller Rips parameter: (a) the persistence diagram; (b) the final clustering with late appearing connected components filtered out. Right: the same setting with 100k input points instead of 10k, processed using the Delaunay graph: (c) persistence diagram; (d) final clustering.

Note finally that the probabilistic nature of our theoretical results does not stem from the algorithm itself, which is deterministic, but from the fact that the input data set must form a dense sampling of some superlevel-set of f for the algorithm to produce a faithful approximation of $\mathrm{D}_0 f$. This event can only occur with some probability since the data points are sampled at random from f.

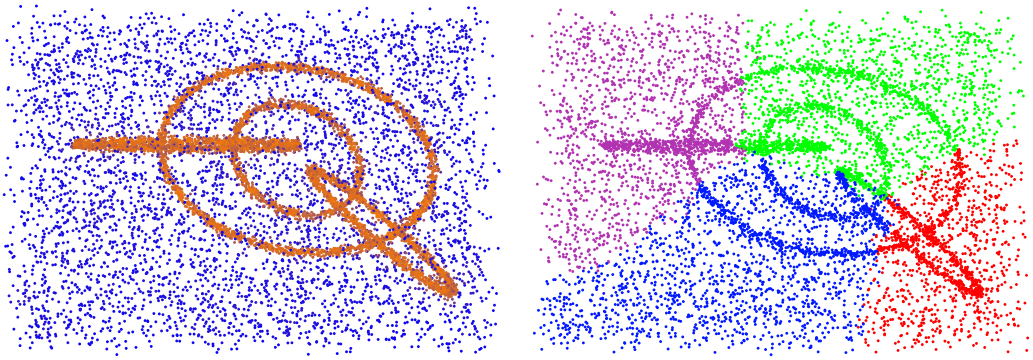
5 Applications

We focused on two types of inputs: (1.) structured synthetic data sets in $\mathbb{R}^2$ and $\mathbb{R}^3$, where direct data inspection allowed us to check our results visually; (2.) simulated alanine-dipeptide protein conformations in $\mathbb{R}^{21}$, where the knowledge of the intrinsic parameters of the simulation allowed



(a) (b) (c) (d)

Figure 6: Left: PDs obtained on the spirals data set from Figure 2 with 10k input points, processed using (a) the k-nn graph with $k = 35$ and (b) the Delaunay graph — the resulting clusterings are visually indistinguishable from 2(d). Right: Spectral clustering on the same data set: (c) the obtained clustering, and (d) a plot of the first 10 eigenvalues.



(a) (b)

Figure 7: (a) The rings data set with the estimated density function. (b) The result obtained using spectral clustering.

us to check our results *a posteriori*. In our experiments we used the two estimators mentioned in Section 3: truncated Gaussian kernel and distance to a measure. Our implementation was done in C++, and it was run on a PC with 8 CPU cores running at 2.4 GHz and 8 GB of RAM[3].

Synthetic Data Our first data set consists of 10k points sampled from two spirals in the unit square, shown in Figure 2 (a). Using a δ-Rips graph, with $\delta = 0.04$, and the inverse of a distance to a measure as our density estimator, we obtain the PD in Figure 2(c). Choosing τ by the gap heuristic we obtain the clustering shown in Figure 2(d). A smaller Rips parameter, $\delta = 0.02$, gives many infinitely persistent components (Figure 5(a)), with all except one appearing late in the PD (near the lower left-hand corner). Components in this part of the PD correspond to low peaks which cannot be distinguished from noise. Ignoring these clusters removes much of the background noise (Figure 5(b)).

We also experimented with the k-nn graph (taking $k = 35$) and the Delaunay graph. The

[3]Each run used only one core and a fraction of the available memory.

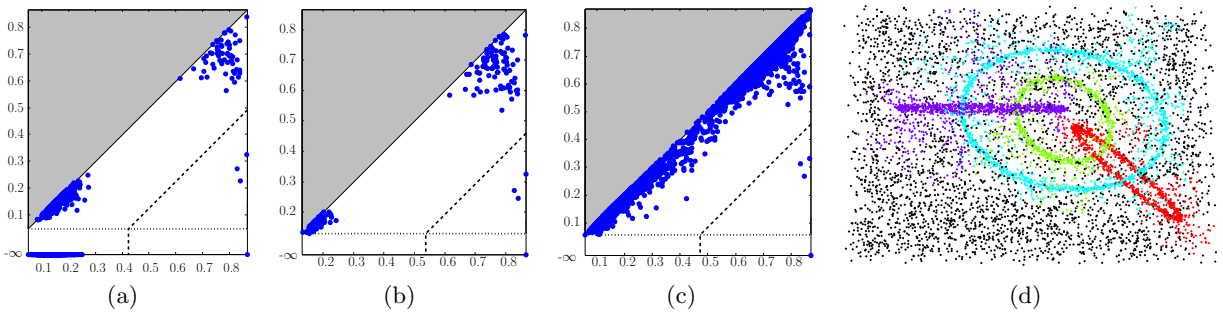


Figure 8: Outputs of ToMATo on the rings data set: the obtained PD with (a) δ-Rips graph, (b) k-nn graph, and (c) Delaunay graph. (d) Clustering obtained with the δ-Rips graph.

resulting PDs are shown in Figures 6(a) and 6(b) respectively. Although not identical, they share the same overall structure with 2 prominent clusters. The resulting clusterings are virtually identical to Figure 2(d), the most notable difference being that there are no late-appearing independent connected components because the Delaunay graph is always connected and the k-nn graph turned out to be connected in this case.

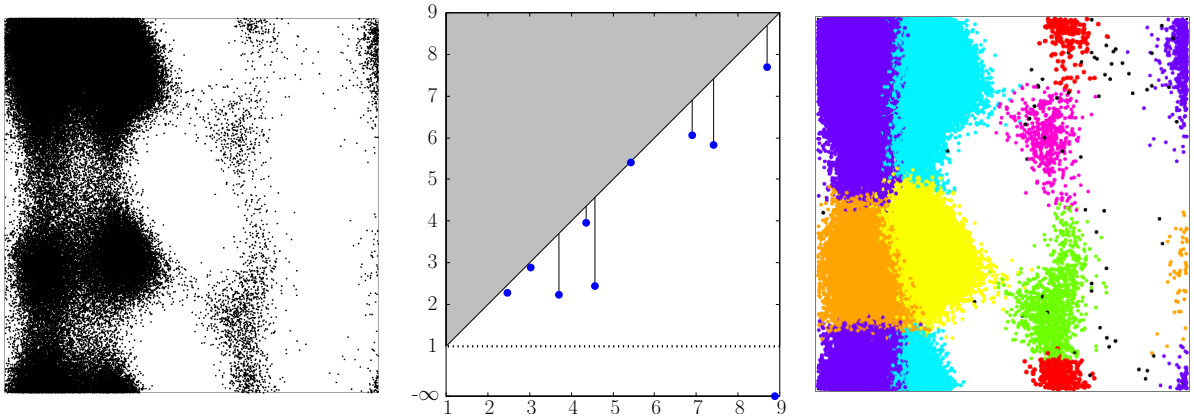
Figures 5(c) and 5(d) illustrate the scalability of our approach by showing the result obtained on the spirals data set with 100k samples. Computation using the Delaunay graph took less than a minute. The PD is much better separated than in the previous cases because the approximation of the PD of the underlying density provably improves with the number of samples, as stated in our theoretical results.

For comparison, we ran spectral clustering [8] on the spirals data set with 10k input points, using the k-nn graph. The result, shown in Figures 6(c) and 6(d), was consistent across choices of input parameters. We were unable to run the code on δ-Rips graphs or on the data set with 100k points because of numerical issues in the eigenvalues computation.

We considered a second synthetic example, made of four noisy interlocked rings in $\mathbb{R}^3$ with uniform background noise added (Figure 7(a)). Spectral clustering again failed on this data set (Figure 7(b)), mainly due to the effect of the background noise on the k-means procedure in eigenspace. It did obtain correct clusters with much of the background noise removed, but this required significant tweaking of the number of neighbors: too many resulted in bad clustering and too few resulted in numerical instability in the computation. For comparison, Figure 8 shows the outputs of ToMATo.

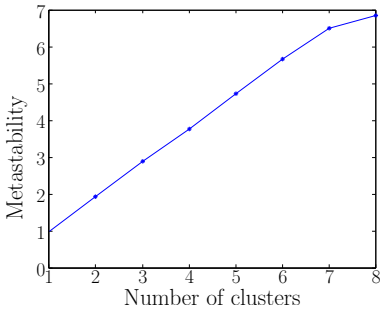
Alanine-dipeptide conformations Next we cluster conformations of the alanine-dipeptide molecule. Our data consists of short trajectories of conformations generated by atomistic simulations of this small protein [10]. Accurate simulation by molecular dynamics must be done at the atomic scale, generally limiting the length of simulations to picoseconds because of the small time steps needed to integrate stiff bond length and angle potentials. Biologically interesting dynamics, however, often occur on the scale of milliseconds. One solution to this issue is to generate a coarser model using *metastable* states [19]. These are conformational clusters between which transitions are infrequent and independent. Such coarser representations are tractable using Markovian models [9, 10, 11] while still allowing for useful simulations. A key problem is the discovery of these metastable states.

The alanine-dipeptide was chosen as example because its dynamics are relatively well-understood:



(a) (b) (c)

Figure 9: Biological data set: (a) input point cloud, projected down to the (ϕ, ψ) domain for visualization purposes; (b) output PD represented on a log-log scale; (c) output clustering with 7 clusters.



(a)

Rank	(Unnormalized) Prominence	Metastability
1	∞	0.99982
2	3827	1.91865
3	1334	2.8813
4	557	3.76217
5	85	4.73838
6	32	5.65553
7	26	6.50757
8	7.2	6.8193
9	3.0	–
10	2.2	–

(b)

Figure 10: Quantitative evaluation of the quality of the output of ToMATo on the biological data set: (a) metastability of the obtained clustering versus the number of clusters; (b) corresponding intervals sorted by decreasing prominence.

it is known that there are only two relevant degrees of freedom, and these are known *a priori*. This makes it possible to visualize the clustering results by projecting the points onto these coordinates which are referred to as ϕ and ψ (Ramachandran plots). In previous work [10], clustering was done manually into 6 clusters. Subsequent work [9] tried to automatically recover these 6 clusters, as we did using our method.

Our input consisted of 960 trajectories, each one made of 200 protein conformations, each conformation being represented as a 21-dimensional vector with 3 coordinates per atom of the protein. For our experiments we took the trajectories and treated the conformations as 192,000 independent samples in $\mathbb{R}^{21}$. The metric used on this point cloud was root-mean-squared deviation (RMSD) after the best possible rigid matching computed using the Theobald method [25]. The RMSD distance matrix was the only input to our clustering scheme. The output is shown in Figure 9.

It appears from the persistence diagram that there could be anywhere from 4 to 7 clusters. The first 4 clusters are much more prominent than the following 3 clusters. Since there is clearly a multiscale behavior, we plot the PD on a log-log scale. From this perspective, the first 4 clusters

are still prominent but relative to their height the 5th and 6th clusters are prominent as well. While the 7th cluster is not as prominent, it is still more prominent than the following clusters, suggesting that 7 is also a reasonable number of clusters. To confirm this insight we came back to the original problem of finding clusters that maximize the *metastability* (as defined in [19]): we computed the metastabilities of all our candidate clusterings, and we reported them in the table and plot of Figure 10. These results show that the metastability increases linearly with the number of clusters, up to 7 clusters, after which it starts leveling off. So, choosing 4, 5, 6 or 7 clusters should not affect the metastablity significantly, thus confirming the observations made from the PD. This is an example of a scenario where the insights into the number of clusters provided by the PD can be validated by exploiting further application-specific information on the data.

Computing the input RMSD distance matrix took the most time: all pairwise distances between conformations were estimated, which took about a day of computation. In order to save space, for each conformation we only recorded the distances to its 15,000 closest conformations in the matrix. On this input, ToMATo only took 10 minutes to run. Meanwhile, the amount of memory used remained approximately constant, which enabled us to make several runs in parallel to find a suitable Rips parameter δ.

Other types of data In the full version of the paper [7] we experiment with a third type of data: bitmap images, which we segment using our clustering scheme in color space. This kind of data typically produces PDs without any significant prominence gaps. The correct number of clusters is then ill-defined, which is consistent with the widely accepted idea that image segmentation is an ill-posed problem. Yet, the PDs still provide a precise understanding of the relationship between the choice of parameter τ and the number of obtained clusters on each image.

6 Conclusion

Our algorithm combines a classical mode-seeking stage and a novel persistence-based cluster merging stage. It is straightforward to implement and provably robust to noise. Rather than rely on heuristics, it returns structural information about the modes of the density function in the form of a persistence diagram, which allows the user to see the relationship between the choice of parameter values and the number of obtained clusters. In many cases this diagram provides insights into the correct number of clusters, which can be automatically inferred by further processing. Our method can work with any density estimator and any metric, including Euclidean, geodesic, and diffusion distances. The point is that the persistence diagram only displays the information that is present in the density function and underlying space (known through the input distance matrix).

Our theoretical developments provide an understanding of when the data has a clear number of clusters, and which parts of the clusters are stable under small perturbations of the input. This opens up the possibility of doing soft-clustering, where each point is assigned to a cluster with some probability. Finally we note that, because we use a topological framework, additional features can be extracted from the data through higher-dimensional persistence diagrams [6], such as the circular structure of the rings in the synthetic data set of Figure 7(a) — although it is not yet clear how this type of information can be exploited.

Acknowledgements

This work was carried out within the associated teams TGDA and CoMeT. It was suported by ANR grant GIGA, European project CG-Learning No. 255827, NSF grants FODAVA 0808515, DMS 0900700, and CCF 1011228, by a Stanford-France grant, and by a gift from Google, Inc.

References

[1] G. Carlsson. Topology and data. *Bulletin of the American Mathematical Society*, 46(2):255–308, 2009.

[2] G. Carlsson, T. Ishkhanov, V. Silva, and A. Zomorodian. On the local behavior of spaces of natural images. *Int. J. Computer Vision*, 76(1):1–12, 2008.

[3] G. Carlsson, A. Zomorodian, A. Collins, and L. Guibas. Persistence barcodes for shapes. In *Proc. Symp. Geom. Process.*, pages 127–138, 2004.

[4] F. Chazal, D. Cohen-Steiner, L. J. Guibas, M. Glisse, and S. Y. Oudot. Proximity of persistence modules and their diagrams. In *Proc. 25th ACM Sympos. Comput. Geom.*, pages 237–246, 2009.

[5] F. Chazal, D. Cohen-Steiner, and Q. Merigot. Geometric inference for measures based on distance functions. Technical report, INRIA, May 2009.

[6] F. Chazal, L. J. Guibas, S. Y. Oudot, and P. Skraba. Analysis of scalar fields over point cloud data. In *Proc. 20th ACM-SIAM Sympos. Discrete Algorithms*, pages 1021–1030, 2009.

[7] F. Chazal, L. J. Guibas, S. Y. Oudot, and P. Skraba. Persistence-based clustering in Riemannian manifolds. Research Report 6968, INRIA, June 2009.

[8] W.-Y. Chen, Y. Song, H. Bai, C.-J. Lin, and E. Y. Chang. *PSC: Parallel Spectral Clustering*, 2008. `http://www.cs.ucsb.edu/~wychen/sc`.

[9] J. D. Chodera, N. Singhal, V. S. Pande, K. A. Dill, and W. C. Swope. Automatic discovery of metastable states for the construction of markov models of macromolecular conformational dynamics. *The Journal of Chemical Physics*, 126(15):155101, 2007.

[10] J. D. Chodera, W. C. Swope, J. W. Pitera, and K. A. Dill. Long-time protein folding dynamics from short-time molecular dynamics simulations. *Multiscale Modeling & Simulation*, 5(4):1214–1226, 2006.

[11] J. D. Chodera, W. C. Swope, J. W. Pitera, C. Seok, and K. A. Dill. Use of the weighted histogram analysis method for the analysis of simulated and parallel tempering simulations. *Journal of Chemical Theory and Computation*, 3(1):26–41, 2007.

[12] D. Cohen-Steiner, H. Edelsbrunner, and J. Harer. Stability of persistence diagrams. *Discrete Comput. Geom.*, 37(1):103–120, 2007.

[13] D. Comaniciu and P. Meer. Mean shift: A robust approach toward feature space analysis. *IEEE Trans. on Pattern Analysis and Machine Intelligence*, 24(5):603–619, May 2002.

[14] H. Edelsbrunner, D. Letscher, and A. Zomorodian. Topological persistence and simplification. *Discrete Comput. Geom.*, 28:511–533, 2002.

[15] M. Ester, H.-P. Kriegel, J. Sander, and X. Xu. A density-based algorithm for discovering clusters in large spatial databases with noise. In *Proc. 2nd Internat. Conf. on Knowledge Discovery and Data Mining (KDD-96)*, pages 226–231. AAAI Press, 1996.

[16] R. Ghrist. Barcodes: the persistent topology of data. *Amer. Math. Soc. Current Events Bulletin*, 45(1):61–75, January 2007.

[17] J. Hartigan. *Clustering Algorithms*. Wiley, 1975.

[18] T. Hastie, R. Tibshirani, and J. H. Friedman. *The Elements of Statistical Learning: Data Mining, Inference, and Prediction*. Springer Series in Statistics. Springer, 2009. Second edition.

[19] W. Huisinga and B. Schmidt. Advances in algorithms for macromolecular simulation, chapter metastability and dominant eigenvalues of transfer operators. *Lecture Notes in Computational Science and Engineering. Springer*, 2005.

[20] W. L. Koontz, P. M. Narendra, and K. Fukunaga. A graph-theoretic approach to nonparametric cluster analysis. *IEEE Trans. on Computers*, 24:936–944, September 1976.

[21] S. P. Lloyd. Least squares quantization in pcm. *IEEE Trans. on Information Theory*, 28(2):129–136, 1982.

[22] S. Paris and F. Durand. A topological approach to hierarchical segmentation using Mean Shift. In *Proc. IEEE conf. on Computer Vision and Pattern Recognition*, 2007.

[23] J. Sander, M. Ester, H.-P. Kriegel, and X. Xu. Density-based clustering in spatial databases: The algorithm GDBSCAN and its applications. *Data Mining and Knowledge Discovery*, 2(2):169–194, 1998.

[24] Y. A. Sheikh, E. Khan, and T. Kanade. Mode-seeking by medoidshifts. In *Proc. 11th IEEE Internat. Conf. on Computer Vision (ICCV 2007)*, October 2007.

[25] D. L. Theobald. Rapid calculation of rmsds using a quaternion-based characteristic polynomial. *Acta Crystallographica Section A: Foundations of Crystallography*, 61(4):478–480, 2005.

[26] G. T. Toussaint. The relative neighbourhood graph of a finite planar set. *Pattern Recognition*, 12:261–268, 1980.

[27] A. Vedaldi and S. Soatto. Quick shift and kernel methods for mode seeking. In *Proc. European Conf. on Computer Vision (ECCV)*, 2008.

[28] U. von Luxburg. A tutorial on spectral clustering. *Statistics and Computing*, 17(4):395–416, 2007.

[29] A. Zomorodian and G. Carlsson. Computing persistent homology. *Discrete Comput. Geom.*, 33(2):249–274, 2005.