Geometric and Topological Approaches to Mode Clustering

Bachelor's Project Mathematics

July 2021

Student: H. Hong

First supervisor: Dr. C.P. Hirsch

Second assessor: Prof.dr. M.A. Grzegorczyk

Abstract

Clustering is one of the most significant tasks nowadays, but the mathematics behind defining clusters are vastly different depending on the choice of metric. Thus, this thesis will focus on density-based clustering which usually fails to give good quality clustering result in high dimensions. We improve the classical density-based clustering into two varying fields of mathematics: geometry and topology. This thesis begins with geometric clustering approach called enhanced mode clustering (EMC). This approach will be provided with some enhancements to mode clustering. Following this, we present topological clustering approach that combines the classical hill climbing algorithm and topological persistence. That is, topological mode analysis tool (ToMATo). Lastly, two implementations of the ToMATo and EMC will be presented in high dimensional data to see if these clustering methods improved the remaining problem from the classical density-based clustering.

Acknowledgement

I would first like to thank my first supervisor, dr. C.P. Hirsch, for proposing such an interesting thesis topic and guiding me through the right path. His enthusiasm and insightful feedback brought my thesis to a higher level.

I would also like to thank my second supervisor, prof. dr. M.A. Grzegorczyk, for being my second supervisor and giving inspirations from the presentation.

Contents

1 Introduction

In the present day, analysing large and complex data sets has become one of the most challenging tasks to solve, and the data available nowadays has become more complex and has more missing data. In this thesis, we will discuss how geometry and topology can make contributions to the analysis of clustered data particularly. Geometry and topology are very fundamental tools in data analysis since geometry is a study of distance and shape, and topology is a study of continuous deformation and shape.

Clustering is especially one of the most important tasks. It has a broad range of applications in various scientific fields such as biology, finance, and medicine [33]. Since there is no concrete definition of cluster there are many different clustering approaches that attempt to define similarity using different notions. Informally, clustering refers to the task of partitioning a set of objects into a number of groups. With regard to finite metric spaces, clustering simply means that the data points in the same cluster are closer to one another than the ones in other clusters. Based on metric, we can construct different schemes of clustering [6], but this thesis will particularly discuss mode clustering and persistence-based clustering, which are subsets of density-based clustering. In density-based clustering, clusters are defined as areas of higher density than the rest of the data set [22].

Here, topological data analysis (TDA) can contribute a significant role in clustering due to homology. That is, topological invariant. In short, TDA is a relatively new branch of data analysis technique whose premise is that the shape of data matters. The intuition of TDA might underestimate its difficulty, but the shape is a somewhat indefinite concept in mathematics as it is difficult to formalise and measure. What makes TDA so interesting is the idea of homology. Basically, the dimension of the n-th dimensional homology group is the number of the n-th dimensional holes. For example, the dimension of the 0-th dimensional homology group is the number of the connected components, which can also be interpreted as the number of the clusters. One of many TDA implementations in clustering is called Topological Mode Analysis Tool (ToMATo) but this thesis will focus exclusively on the ToMATo algorithm for the topological approach of clustering.

The intuitive ideas of ToMATo is to measure the significance, i.e., prominence, of the density peaks using topological persistence [11]. In conventional TDA, we are interested in the n-th dimensional homology of simplicial complexes since constructing simplicial complexes using, for example, Vietoris-Rips complex is one of the easiest ways to transform discrete data into a continuous form. However, when it comes to density-based clustering, we are interested in the super of sublevel sets of the density function. The ToMATo improves a classical hill-climbing algorithm introduced in [21] whose problem arises in its perturbation management. In practice, managing the small perturbations is one of the most significant tasks since only data points drawn from the true density function are known while the true density function is unknown. These perturbations bother to understand the structure of the density function. The improvement comes from topological persistence. The idea of this improvement is to take the whole family of level sets of the density function. We call this family of level sets the level set filtration. For example, if we take the superlevel filtration, we are interested in tracking the evolution of topology throughout the filtration of superlevel sets of the density function. By doing so, the ToMATo offers less sensitivity to perturbations. As the ToMATo is insensitive to perturbations, the choice of density estimator only makes a subtle difference in its results whereas other clustering methods tend to have high dependency on the choice of density estimator. In this thesis, the *distance to a measure* introduced in [9] is chosen for the density estimator. For more choices of density estimators, see [11]. While the choice of density estimator only makes a subtle contribution on the result, the choice of a graph remains a non-trivial task. This thesis will only use δ-Rips graph. If one is interested in the choice of a neighbourhood graph in detail, please have a look at [1, 11].

As mentioned above, this thesis will also study the mode clustering proposed in [13]. There is room for improvement in mode clustering: 1. Mode clustering is a branch of hard clustering. Which implies that the connectivity between clusters cannot be measured. 2. The choice of bandwidth parameter of kernel density estimator is unclear in practice. 3. In high dimensions, the variance creates small clusters in the kernel density estimator, which bothers the performance of mode clustering. Hence, this thesis will suggest a solution called *enhanced mode clustering* (EMC) to these problems followed from [12, 13, 24, 27, 33]. Here, both methods assume that the true density function is a *Morse function* to guarantee the result clusters are disjoint [25]. However, since this thesis will not go deep into Morse theory, the reader can find more information from [25, 26].

Following the theory of the mode- and persistence-based clustering methods, we will implement both the EMC and ToMATo in the three different datasets that can be found in [3, 15, 16]. The goal of the implementation is to compare two introduced clustering methods and k-means using adjusted Rand index introduced in [29] so that we observe how well these clustering methods perform in the high dimensional setting. Here, we chose k-means while it is a centroid-based clustering to objectify the adjusted Rand index of two other clustering methods, i.e., comparing only two methods might lead to violating observer bias. Since it was claimed in [13] that the persistent homology is inappropriate to merge small clusters into significant clusters especially in high dimensions, we would like to check whether this claim is true or not.

The primary goal of this thesis is to provide a clear analysis of mathematics behind EMC and ToMATo. That is, to provide the improvements of the classical mode clustering and the hill climbing algorithm respectively. Also, we motivate these clustering methods so that, for example, the readers can clearly understand why the TDA suggests homology to analyse the data and why we consider the superlevel filtration for the ToMATo algorithm. For transparency, the codes used for implementations and generating plots will be included.

2 Geometric Mode Clustering

Clustering is a huge topic, which implies that it has a broad range of branches using different notions of clusters. This section will consider *mode clustering*, which is a branch of density clustering. Additionally, some enhancements to mode clustering from [13] will be provided. The key information of this section is the notions and properties of mode clustering, soft (mode) clustering, measuring cluster connectivity, and consistency of mode estimation. The following concepts are drawn from [12, 13, 24, 27, 33].

2.1 Mode Clustering

The idea of *mode clustering* is to identify clusters by using the gradient flow of a density function, or equivalently we identify clusters by *basins of attraction* of the peaks of the density function. A basin of attraction is the domain of all points flowing into the same local maximum where the flows are defined by the gradient of the density function. Throughout the thesis, we consider a n-dimensional Riemannian manifold $\mathbb{X}$ and assume that the density function $f : \mathbb{X} \to \mathbb{R}$ is a Morse function and f always has compact support $\mathbb{K} \subset \mathbb{X}$. Note that k-means clustering, which is the most commonly used clustering method, assumes that the data is convex, i.e., k-means fails to give good performance of clustering when the data is non-convex. Unlike k-means clustering, the mode clustering does not need to assume the shape of the clusters and it does not require prior-specification of the number of the clusters[1].

Definition 2.1 (Morse function)**.** The function f is a **Morse function** if the Hessian[2] is nonsingular at each critical point.

Suppose that f has k local modes (or peaks) $\mathcal{M} = \{m_1, \dots, m_k\}$. Assume that f has gradient g and Hessian H. Note that x is a critical point if $g(x) = (0, \dots, 0)^T$. Thus, the modes m_j for $j = 1, \dots, k$ satisfies $g(m_j) = (0, \dots, 0)^T$. Given arbitrary $x \in \mathbb{X}$, if we follow the unique gradient ascent path starting at x, we will eventually reach one of the modes m_j unless x is in a set of Lebesgue measure 0. That is, m_j is the destination of x. In order to introduce a rigorous definition of *destination*, let us first introduce a *path* or also called *integral curve*.

Definition 2.2 (Path)**.** A **path** $\pi_x : \mathbb{R} \to \mathbb{X}$ that leads from x to a mode is defined by the differential equation

$$\pi_x'(t) = \nabla f(\pi_x(t)), \quad \pi_x(0) = x. \tag{1}$$

According to Morse in [25], *paths* never intersect at critical points, hence paths can be used to partition the space. As each x will reach one of the *destinations*, i.e., modes, we define the destination as follows.

Definition 2.3 (Destination)**.** A **destination** for the path starting at x is defined by

$$\text{dest}(x) = \lim_{t \to \infty} \pi_x(t) = m_j \tag{2}$$

for some mode m_j except for $x \in E$ where E with Lebesgue measure 0.

[1]This does not mean that the density-based clustering always gives the correct number of the clusters

[2]Hessian is a square matrix of second-order partial derivatives of a scalar-valued function, or scalar field.

As mentioned above, the basin of attraction of the mode m_j is the domain of all points flowing into m_j.

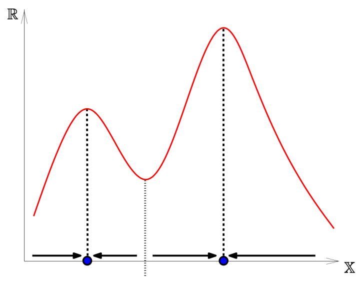


Figure 1: The center-line gives the separation between basins of attraction of two peaks of the density function [11]

Figure 1 shows that the center-line separates the basins of attraction of the two peaks and $x \in \mathbb{X}$ corresponding this center-line has Lebesgue measure 0. Again according to Morse in [25], these two basins of attraction are disjoint.

Definition 2.4 (Basin of attraction)**.** A **basin of attraction** of m_j is defined by

$$B(m_j) = \{x : \text{dest}(x) = m_j\}, \quad j = 1, \ldots, k. \tag{3}$$

In practice, density-based methods for clustering assumes that data points are drawn from an unknown density function f, so we need to estimate it. In order to estimate the density, we use one of the classical estimators called the *kernel density estimator* (KDE).

Definition 2.5 (Kernel density estimator)**.** Let $X_1, \ldots, X_n$ be i.i.d.[3] samples drawn from the density f, and K be a smooth and symmetric kernel function. The KDE with bandwidth $h > 0$ is defined by

$$\widehat{f}_h(x) = \frac{1}{nh^d} \sum_{i=1}^{n} K\left(\frac{\|x - X_i\|}{h}\right). \tag{4}$$

The modes of $\widehat{f}_h$ are given by

$$\widehat{\mathcal{M}} = \{\widehat{m}_1, \ldots, \widehat{m}_{\widehat{k}}\} \tag{5}$$

and $\widehat{\text{dest}}(x)$ is the path destination under the density $\widehat{f}_h$ of any $x \in \mathbb{X}$. Both the modes and destinations are easily found by a *mean shift* algorithm proposed by Comaniciu and Meer in [14]. In detail, we can find the destination (in our case, mode) of the initial point $x^{(0)}$ along the path by

$$x^{(t+1)} = \frac{\sum_{i=1}^{n} X_i K\left(\frac{\|x^{(t)} - X_i\|}{h}\right)}{\sum_{i=1}^{n} K\left(\frac{\|x^{(t)} - X_i\|}{h}\right)}. \tag{6}$$

[3]independent and identically distributed

Thus, the convergent point of (6) is a local mode of $\widehat{f}_h$ of any initial point $x \in \mathbb{X}$. The convergence of the mean shift algorithm has been studied in [2]. The mean shift algorithm can be computationally expensive as its time complexity is given by $O(n^2)$ where n is the number of observations.

By using the mean shift algorithm, we can define the *sample basin of attraction.*

Definition 2.6 (Sample basin of attraction)**.** A **sample basin of attraction** of $\widehat{m}_j \in \widehat{\mathcal{M}}$ is defined by

$$\widehat{B}(\widehat{m}_j) = \left\{x : \widehat{\text{dest}}(x) = \widehat{m}_j\right\}, \quad j = 1, \dots, \widehat{k}. \tag{7}$$

2.2 Soft Mode Clustering

The classical mode clustering is a branch of hard clustering. That is, partitioning a set of points into a number of groups such that each point is only assigned to one cluster. Meanwhile, *soft clustering* methods [23] does not yield a binary answer, i.e., a point can be assigned to one or more than one clusters. This is usually done by introducing a *(soft) assignment vector*, which is a probability distribution over clusters. For instance, let us denote $a(x)$ is the assignment vector where $a_j(x)$ is the probability of x belonging to cluster j. Suppose that the assignment vector of a point x is given by $a(x) = (0.7, 0.02, 0.2, 0.05, 0.03)$. Then, we say that there is high chance that x belongs to cluster 1, but the chance that x might belong to cluster 3 is not negligible. This section will illustrate the idea of *soft mode clustering* and introduce the exact notion of *soft assignment vector* to obtain soft mode clustering in a rigorous manner.

As mentioned above, one characteristic of soft clustering is *soft assignment vector*. Then, recall that the density function is given unknown in practice, so the soft assignment vector is also unknown. To estimate the soft assignment vector, consider a Markov chain starting at $x \in \mathbb{X}$. Then, j-th entry of the soft assignment vector can be defined by the conditional probability that the diffusion starting at x firstly arrives in mode j given that the diffusion starting at x is absorbed in one of the modes. More precisely, we can consider this diffusion process as a Markov process. Let $\left\{\widehat{m}_1, \dots, \widehat{m}_{\widehat{k}}, X_1, \dots, X_n\right\}$ be the state space and the jump probability to x given starting at y is given by

$$q_h(y|x) = \frac{K_h(x, y)}{\int K_h(x, y)\, \mathrm{d}y} \tag{8}$$

where $K_h(x, y) = K\left(\dfrac{\|x - y\|}{h}\right)$. Note that the first $\widehat{k}$ states are absorbing states since we are never left once we reach one of the modes. Thus, the transition probability from state X_i is given by

$$\begin{aligned} \mathbb{P}(X_i \to \widehat{m}_\ell) &= \frac{K_h(X_i, \widehat{m}_\ell)}{\sum_{j=1}^n K_h(X_i, \widehat{m}_j) + \sum_{\ell=1}^{\widehat{k}} K_h(X_i, \widehat{m}_\ell)} \\ \mathbb{P}(X_i \to X_j) &= \frac{K_h(X_i, X_j)}{\sum_{j=1}^n K_h(X_i, X_j) + \sum_{\ell=1}^{\widehat{k}} K_h(X_i, \widehat{m}_\ell)} \end{aligned} \tag{9}$$

for $i, j = 1, \ldots, n$ and $\ell = 1, \ldots, \widehat{k}$. Then, the transition probability matrix $\mathbf{P}$ is given by

$$\mathbf{P} = \begin{bmatrix} \mathbf{I}_{\widehat{k}} & \mathbf{o}_{\widehat{k}\times n} \\ S & T \end{bmatrix} \tag{10}$$

where $\mathbf{I}$ is the identity matrix of size $\widehat{k}$, $\mathbf{o}_{\widehat{k}\times n}$ is the $\widehat{k} \times n$ zero matrix, S is an $n \times \widehat{k}$ matrix with the (i, j)-th element $S_{ij} = \mathbb{P}(X_i \to \widehat{m}_j)$ and T is a $n \times n$ matrix with the (i, j)-th element $T_{ij} = \mathbb{P}(X_i \to X_j)$. Then, the *soft assignment estimator* can be written with respect to the absorbing probability from X_i to $\widehat{m}_j$.

Definition 2.7 (Soft assignment estimator)**.** A soft assignment estimator is defined by

$$\widehat{a}_j(X_i) = \widehat{A}_{ij} \tag{11}$$

where A_{ij} is the (i, j)-th element of the absorbing probability matrix A

$$\widehat{A} = S(\mathbf{I}_{\widehat{k}} - T)^{-1}. \tag{12}$$

2.3 Cluster Connectivity

Classical mode clustering based on mean shift algorithm in [14] is a hard clustering, hence the result of this algorithm yields the binary answer, i.e., a point is only assigned to one cluster. This creates a cluster uncertainty since a point can be related to multiple clusters, e.g., a point placed near the boundary of two basins of attraction is not only related to one but two clusters. In order to capture this cluster uncertainty, Chen et al in [13] introduced the *enhanced mode clustering*. One of many attributes of enhanced mode clustering is *cluster connectivity* that measures the connectivity between two different clusters.

Consider the density function $f : \mathbb{X} \to \mathbb{R}$ and suppose that f has k local modes $\mathcal{M} = \{m_1, \ldots, m_k\}$, thus there are k clusters $B(m_1), \ldots, B(m_k)$ corresponding to the local modes $m_1, \ldots, m_k$.

Definition 2.8 (Connectivity)**.** Given soft assignment vector $a(x) : \mathbb{X} \to \mathbb{R}^k$, the **connectivity** between cluster i to cluster j is defined by

$$\begin{aligned} \Omega_{ij} &= \frac{1}{2}\Big(\mathbb{E}[a_i(X)|X \in B(m_j)] + \mathbb{E}[a_j(X)|X \in B(m_i)]\Big) \\ &= \frac{1}{2}\frac{\int_{B(m_j)} a_i(x)f(x)\,\mathrm{d}x}{\int_{B(m_j)} f(x)\,\mathrm{d}x} + \frac{1}{2}\frac{\int_{B(m_i)} a_j(x)f(x)\,\mathrm{d}x}{\int_{B(m_i)} f(x)\,\mathrm{d}x} \end{aligned} \tag{13}$$

Connectivity between cluster i and j is high if clusters i and j are close. Since the density is usually unknown in practice, we need to define the *connectivity estimator*.

Definition 2.9 (Connectivity estimator)**.** An estimator of Ω_{ij} is defined by

$$\widehat{\Omega}_{ij} = \frac{1}{2}\left(\frac{1}{N_j}\sum_{\ell=1}^{n} \widehat{a}_i(X_\ell)\mathbb{1}_{\widehat{B}(\widehat{m}_j)}(X_\ell) + \frac{1}{N_i}\sum_{\ell=1}^{n} \widehat{a}_j(X_\ell)\mathbb{1}_{\widehat{B}(\widehat{m}_i)}(X_\ell)\right), \quad i, j = 1, \ldots, \widehat{k} \tag{14}$$

where $N_i = \sum_{\ell=1}^{n} \mathbb{1}_{\widehat{B}(\widehat{m}_i)}(X_\ell)$ is the number of sample in cluster $\widehat{B}(\widehat{m}_i)$.

The connectivity estimator $\widehat{\Omega}$ is a consistent estimator of the connectivity Ω by the result in Chazal et al [10] (see chapter 4.1).

2.4 Consistency

This section will describe the consistency of local mode estimation. That is, when the data is large enough (desirably infinitely many points), the estimation of the local modes converges in probability[4] to the true local modes. Unlike the consistency for global mode estimation [10], the consistency for estimating local mode is still elusive. Thus, we are interested how close two sets (where one set contains the true local modes and the other set contains the local modes of the density estimator) are. It is a nontrivial question to ask *what is the closeness of two sets* since we have multiple points to measure the distance. Hence, as illustrated in [13], the consistency of local mode estimation will be described with respect to the *Hausdorff distance*. Hence, let us first define the Hausdorff distance.

Definition 2.10 (Hausdorff distance)**.** Let A and B be two non-empty set. Then, their **Hausdorff distance** is defined by

$$d_{\mathbf{H}}(A, B) = \inf \{\epsilon \geq 0 : A \subseteq B_\epsilon \ \text{ and } \ B \subseteq A_\epsilon\} \tag{15}$$

where $A_\epsilon = \{y : \min_{x \in A} \|x - y\| \leq \epsilon\}$.

In order to understand the intuition of the Hausdorff distance, consider sets A and B and expand both the set A and set B such that $A \subseteq B_\epsilon$ and $B \subseteq A_\epsilon$ where A_ϵ is the extension of the set A by ϵ. We want to find the smallest ϵ but still satisfying $A \subseteq B_\epsilon$ and $B \subseteq A_\epsilon$. Then, this yields the distance between two sets A and B.

Let $K^{(\alpha)}$ denote the α-th derivative of K and $\mathbf{BC}^r$ denote the collection of functions with bounded continuously derivatives up to r-th order.

Remark. The following assumption and theorem are directly taken from [13].

Assumption 2.1. *This thesis will assume that the kernel function satisfies two following conditions:*

(i) The kernel function $K \in \mathbf{BC}^3$ is symmetric, non-negative and

$$\int x^2 K^{(\alpha)}(x)\, \mathrm{d}x < \infty, \qquad \int \left(K^{(\alpha)}(x)\right)^2 \mathrm{d}x < \infty$$

for all $\alpha = 0, 1, 2, 3$.

(ii) The kernel function satisfies condition K_1 of Gine and Guillou [17]. That is, there exists some $A, v > 0$ such that for all $0 < \epsilon < 1$, $\sup_Q N(\mathcal{K}, L_2(Q), C_K\epsilon) \leq \left(\frac{A}{\epsilon}\right)^v$ where $N(T, d, \epsilon)$ is the ϵ-covering number for a semi-metric space (T, d) and

$$\mathcal{K} = \left\{ u \mapsto K^{(\alpha)}\left(\frac{x - u}{h}\right) : x \in \mathbb{R}^d, h > 0, |\alpha| = 0, 1, 2, 3 \right\}.$$

Note that ϵ-covering number for a semi-metric space (T, d) is the smallest number of $B(x; \epsilon^*)$[5] covering T where $x \in T$ and $\epsilon^* < \epsilon$.

Assumption 2.1 (i) guarantees the smoothness of the kernel function and Assumption 2.1 (ii) controls the complexity of the kernel function. That is, controlling the degree of a polynomial of the kernel function.

[4] if for all $\epsilon > 0$, $\lim_{n \to \infty} \mathbb{P}(|X_n - X| > \epsilon) = 0$, X_n converges in probability to X

[5] ball of radius ϵ^* and centered at x

Theorem 2.2 (Consistency of local modes estimation)**.** *Suppose $f \in \mathbf{BC}^3$ and the kernel function K satisfies Assumption 2.1. Let B_3 be the bound for the partial derivatives of f up to the third order and $\widehat{\mathcal{M}}_n \equiv \widehat{\mathcal{M}}$ be the collection of local modes of the KDE $\widehat{f}_n$ and $\mathcal{M}$ be the local modes of f. Let $\widehat{k}_n$ be the number of estimated local modes and k be the number of true local modes. Assume*

(a) There exists $\lambda_* > 0$ such that

$$0 < \lambda_* \leq |\lambda_1(m_j)|, \quad j = 1, \dots, k$$

where $\lambda_1(x) \leq \cdots \leq \lambda_d(x)$ are the eigenvalues of Hessian matrix of $f(x)$.

(b) There exists $\eta_1 > 0$ such that

$$\{x : \|\nabla f(x)\| \leq \eta_1, 0 > -\lambda_*/2 \geq \lambda_1(x)\} \subset \mathcal{M}_{\lambda_*/2dB_3}$$

where λ_* satisfies $0 < \lambda_* \leq |\lambda_1(m_j)|$ defined in assumption (a).

If h is sufficiently small and n is sufficiently large,

1. *(Modal consistency) there exits some constants $A, C > 0$ such that*

$$\mathbb{P}(\widehat{k}_n \neq k) \leq Ae^{-Cnh^{d+4}}$$

2. *(Location convergence) the Hausdorff distance between local modes and their estimators satisfies*

$$d_{\mathbf{H}}(\mathcal{M}, \widehat{\mathcal{M}}_n) = O(h^2) + O_P\left(\sqrt{\frac{1}{nh^{d+2}}}\right).$$

Proof. See page 235 – 238 in [13]. □

An intuitive explanation for result 2 is the following: w.l.o.g. assume that d is fixed and let $g_1(h) \in O(h^2)$ and $g_2(h) = O\left(\sqrt{\frac{1}{h^{d+2}}}\right)$. Then, (2) can be written as:

$$d_{\mathbf{H}}(\mathcal{M}, \widehat{\mathcal{M}}_n) = g_1(h) + g_2(h) \cdot X_n \tag{16}$$

where X_n is a random variable. That is,

$$\mathbb{P}(|X_n| > \frac{c}{\sqrt{n}}) \overset{n\to\infty}{\longrightarrow} 0 \tag{17}$$

where c is a constant.

Result 1 (modal consistency) is derived as follows: the idea is that if the gradient and Hessian of $\widehat{f}$ is sufficiently close to the gradient and Hessian of the true density f, each local mode m_j corresponds to $\widehat{m}_j$, i.e., if

$$\left\|\nabla f - \nabla \widehat{f}\right\|_\infty, \quad \left\|\nabla\nabla f - \nabla\nabla \widehat{f}\right\|_\infty \tag{18}$$

are small enough, the true modes and estimated modes are the same.

Result 2 (location convergence) is derived as follows: note that $\nabla f(m_j) = \nabla \widehat{f}(\widehat{m}_j) = 0$. Then, by Taylor expansion, we have

$$\begin{aligned} \nabla \widehat{f}(m_j) &= \nabla \widehat{f}(m_j) - \nabla \widehat{f}(\widehat{m}_j) && (19) \\ &= \nabla\nabla \widehat{f}(\widehat{m}_j)(m_j - \widehat{m}_j) + o(\|m_j - \widehat{m}_j\|). && (20) \end{aligned}$$

Also, we can expand $\nabla \widehat{f}(m_j)$ as

$$\begin{aligned} \nabla \widehat{f}(m_j) &= \nabla \widehat{f}(m_j) - \nabla f(m_j) && (21) \\ &= O(h^2) + O_P\left(\sqrt{\frac{1}{nh^{d+2}}}\right). && (22) \end{aligned}$$

by the rate of pointwise convergence in nonparametric theory in [30]

Hence,

$$\begin{aligned} \nabla \widehat{f}(m_j) &= \nabla\nabla \widehat{f}(\widehat{m}_j)(m_j - \widehat{m}_j) + o(\|m_j - \widehat{m}_j\|) && (23) \\ &= O(h^2) + O_P\left(\sqrt{\frac{1}{nh^{d+2}}}\right). && (24) \end{aligned}$$

This follows

$$\|m_j - \widehat{m}_j\| = O(h^2) + O_P\left(\sqrt{\frac{1}{nh^{d+2}}}\right). \tag{25}$$

Therefore,

$$d_{\mathbf{H}}(\mathcal{M}, \widehat{\mathcal{M}}_n) = \max_{j=1,\ldots,k} \|m_j - \widehat{m}_j\| = O(h^2) + O_P\left(\sqrt{\frac{1}{nh^{d+2}}}\right). \tag{26}$$

2.5 Bandwidth Selection

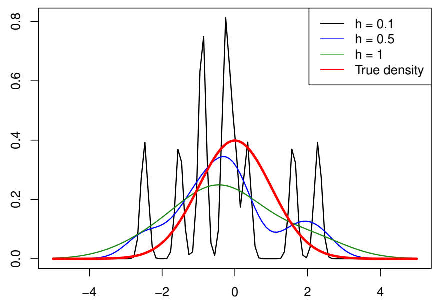


Figure 2: The KDE with different values of the bandwidth parameter.

The choice of bandwidth parameter h is a very important task in density estimation, hence also in mode clustering. The idea of bandwidth parameter is that the bandwidth changes the gradient of the density function, e.g., if the bandwidth is increased, the absolute value of the gradient of the density decreases around the sample points, i.e., smoothing the kernel density estimation (see Figure 2). Hence, in order to optimise the value of bandwidth, we want to minimise the difference between the estimated gradient and the true gradient. Chacón et al [7] suggested a classical performance measure for kernel density estimation called *mean integrated square error* (MISE). The MISE for the gradient of the density f is

$$\mathrm{MISE}(\nabla\widehat{f}_n) = \mathbb{E}\left[\int \left\|\nabla\widehat{f}_n(x) - \nabla f(x)\right\|_2^2 \mathrm{d}x\right] = O(h^4) + O\left(\frac{1}{nh^{d+2}}\right). \tag{27}$$

Hence, the asymptotically optimal bandwidth is

$$h = Cn^{-\frac{1}{d+6}} \tag{28}$$

for some constant C. However, it is usually not clear what the optimal value is for bandwidth h since C is unknown in practice. Due to such practical issue, there are a few methods to automatically choose a bandwidth parameter. One widely used method is called *Silverman's rule of thumb* (or *normal reference rule*) [13]. Silverman's rule of thumb computes an optimal bandwidth parameter h by assuming that the kernel is normally distributed.

$$h_{SR} = \bar{S}_n \cdot \left(\frac{4}{d+4}\right)^{\frac{1}{d+6}} n^{-\frac{1}{n+6}} \tag{29}$$

with

$$\bar{S}_n = \frac{1}{d}\sum_{j=1}^{d} S_{n,j}. \tag{30}$$

where $S_{n,j}$ is the sample standard deviation with respect to j-th coordinate. Note that the Silverman's rule of thumb assumes that the kernel is normally distributed. Meaning that the Silverman's rule of thumb will yield a highly inaccurate smoothing parameter h if the density is not close to a normal distribution.

The Silverman's rule of thumb has two major advantages. One advantage is, according to [31], the Silverman's rule of thumb tends to over-smooth when the standard Gaussian kernel is given. Over-smoothing is usually good for clustering. The reason is that we can merge the small clusters into the significant clusters, especially in the high dimensions. The other advantage is that high dimensions do not affect computation of the bandwidth. Usually the computation becomes more expensive in higher dimensions, however the computation of the Silverman's rule of thumb 29 stays rather simple with respect to d. By normalising the data, the mean of the sample standard deviation in (30) becomes 1. Hence, normalising the data reduces the Silverman's rule of thumb to

$$h_{SR} = \left(\frac{4}{d+4}\right)^{\frac{1}{d+6}} n^{-\frac{1}{n+6}}. \tag{31}$$

2.6 Denoising Small Clusters

Denoising small clusters is not a big issue in low dimensions. However, it becomes a major task to solve in high dimensions since the variance, especially in high dimensions, creates small noises in KDE, hence the location convergence becomes slower (see 2 of Theorem 2.2 for further information). We have a question regarding denoising small clusters, that is *how do we merge (or eliminate) noises during mode clustering?* To do this, we can simply set a cluster size[6] threshold, and for simplicity, let us call it a *denoising threshold* n_0. To obtain only significant clusters, we merge every clusters of size less than a given denoising threshold n_0.

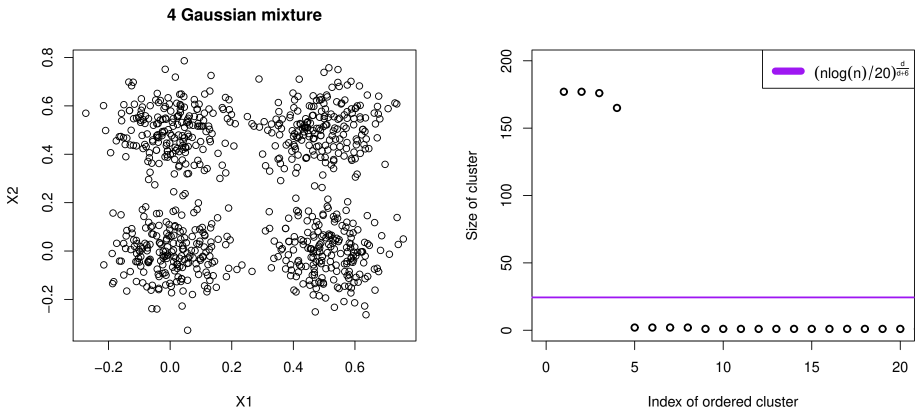


Figure 3: (left) 4 clusters, 800 data points drawn from a Gaussian distribution with $\mathbf{\Sigma} = 0.1\mathbf{I}$; (right) the size of clusters plot of the 4 Gaussian mixture dataset with the denoising threshold.

Choosing the denoising parameter can be obtained by trial-and-error since there exists no theory behind of choosing the parameter n_0. Due to this issue, we use the recommended

[6]the number of data points within the cluster

denoising parameter from [13], which is given by

$$n_0 = \left(\frac{n\log(n)}{20}\right)^{\frac{d}{d+6}}. \tag{32}$$

The left plot of Figure 3 illustrates two dimensional projection of 4 Gaussian clusters in 8 dimensions where each cluster contains 200 point. The right plot shows the ordered size of clusters plot of 4 Gaussian clusters with the denoising threshold given in (32). Note that the smoothing bandwidth parameter h is chosen by the Silverman's rule of thumb in (29). From the ordered size of clusters plot in Figure 3, we can conclude that there are 4 significant clusters since there are 4 clusters with size larger than the denoising threshold (purple line).

An increment of the sample size follows the significantly longer computation time, which can be observed from the time complexity of the mean shift algorithm $O(n^2)$ where n is the sample size. However, the computation time is not affected by the increment of the dimensions, but the accuracy of the algorithm. This is explained as: 1. the computation of the Silverman's rule of thumb in 29 does not become more expensive by increasing the dimension; 2. the location convergence from Theorem 2.2 2 states that the higher dimension implies the larger distance between local modes and their estimators, which does affect the accuracy of the clustering result. In case of 4 Gaussian mixture, the computation took 26.810 seconds for 800 data points. But by increasing the sample size to 1200, the computation took 45.688 seconds and by decreasing the sample size to 200, the computation took 1.568 seconds.

3 Topological Mode Clustering

The goal of this section aims at explaining what *topological data analysis* (TDA) is, and how TDA makes contributions to analysis of mode clustering. In order to do this, let us give an intuition of what TDA is, and give an introduction of TDA clustering algorithm such as *Topological Mode Analysis Tool* (ToMATo), which will naturally give an idea of the concept of clustering powered by TDA. Compared to EMC, the ToMATo performs with less steps, thus the computation time is less than the computation time of EMC. The total computation time of ToMATo is $O(n + m\log(n))$ where m is the number of the edges from δ-Rips graph [11]. In practice, $m = O(n)$ as δ is usually taken small enough. Also, notice that the ToMATo algorithm is not affected by the choice of density estimator, which is an advantage of using ToMATo instead of EMC.

TDA is an approach to analyse data by assigning topological invariants to data. Loosely speaking, we take a look at the shape of the data. However, data is typically given as a discrete sample whose topology is rather uninteresting [5]. Hence, the data must be transformed into a continuous form. For now, it is enough to see that topology is interesting enough to be used in (typical) discrete data. This section will firstly demonstrate some key definitions and theorems to understand TDA from scratch, and investigate how TDA makes contributions to analysis of mode clustering. In the later sections, such techniques will eventually be simulated with real dataset in Comparison on Simulated and Real Dataset section 4. In order to keep the consistency, the concepts in the following subsections are introduced in [4, 5, 6, 18, 20, 33].

3.1 Simplicial Complexes

The first question one might ask is *why do we consider simplicial complexes in TDA?* Loosely speaking, simplicial complex provides a good representations of topological structures in computers as the continuous spaces need to be transformed into a discrete form again before we apply it in computer, and this is done by means of *simplicial complexes* [5]. Before we define simplicial complex, let us discuss what simplices are.

3.1.1 Simplices

Definition 3.1. A set of points $\{p_0, \dots, p_n\} \subset \mathbb{R}^d$ is said to be **geometrically independent**[7] if for any (real) scalars $\{t_i\}_{i=0}^n$, the equations

$$\sum_{i=0}^{n} t_i = 0, \quad \sum_{i=0}^{n} t_i p_i = 0 \tag{33}$$

imply that $t_0 = t_1 = \cdots = t_n = 0$. It is straightforward to verify that $p_0, \dots, p_n$ are *geometrically independent* if and only if the vectors $p_1 - p_0, \dots, p_n - p_0$ are linearly independent.

Definition 3.2 (Simplex)**.** Let $\{p_0, \dots, p_n\}$ be geometrically independent set in $\mathbb{R}^d$. We define the n-**simplex** σ spanned by the points p_i to be the set of all points $x \in \mathbb{R}^d$ of the form

$$x = \sum_{i=0}^{n} t_i p_i \tag{34}$$

[7]it is also called *affinely independent*

where $\sum_{i=0}^{n} t_i = 1$, and $t_i \geq 0$ for all i. The point x is a **convex combination** of the points $p_0, \ldots, p_n$, and the uniquely determined numbers t_i are called the **barycentric coordinates**.

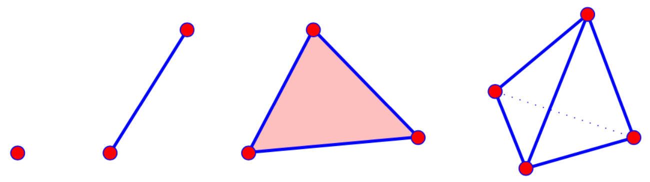


Figure 4: Simplices of dimension zero, one, two and three. [4]

From definition 3.2, one might already observed that a 0-simplex is a point, a 1-simplex is a line segment, a 2-simplex is a triangle, 3-simplex is a tetrahedron and so on (Figure 4). Note that the points $p_0, \ldots, p_n$ that span a simplex σ are called the **vertices** of σ, and the number n is the **dimension** of σ. Lastly, we define the **boundary** of σ as the union of the proper face[8] of σ, and let us denote the boundary of σ as $\partial\sigma$.

3.1.2 Simplicial Complexes

A topological spaces called *simplicial complex* can be constructed by gluing simplices along faces, and the definition is given by the following.

Definition 3.3 (Simplicial complex)**.** A **simplicial complex** K in some Euclidean space $\mathbb{R}^d$ is a finite collection of simplices in $\mathbb{R}^d$ such that

(i) Every face of a simplex in K is in K, i.e., if σ is a simplex of K and τ is a face of σ, then τ is a simplex of K.

(ii) The non-empty intersection of any two simplices of K is a common face of each of them.

Note that the **dimension** of K is the maximum dimension of its simplices, and a subcomplex L of K is a subset of K that is a simplicial complex. The p**-skeleton** of K denoted by K^{α} is the union of all simplicies of K of dimension at most p.

Since the precise geometry of simplicial complex does not take a crucial role in this thesis, we will restrict ourselves to abstract simplicial complex. That is in brief, a purely combinatorial construct of simplicial complex. Hence, if it is said a *simplicial complex* in this thesis, it indicates an *abstract simplicial complex.*

Definition 3.4 (Abstract simplicial complex)**.** An abstract simplicial complex is a finite collection A of finite non-empty sets, such that if α is an element of A, then so is every nonempty subset of α.

We will use one of the most widely used simplicial complexes in TDA called *Vietoris-Rips complex* as we will implement this complex later of this thesis.

[8] τ is a proper face of σ if $\tau \subset \sigma$, i.e., $\tau \neq \sigma$

Definition 3.5 (Vietoris-Rips complex)**.** Given a set of points $P = \{p_0, \dots, p_n\}$ and a metric diam[9], the **Vietoris–Rips complex** of P at scale ϵ is the simplicial complex

$$\mathrm{VR}_\epsilon(P) = \{\sigma \subseteq P : \mathrm{diam}(\sigma) \leq 2\epsilon\}. \tag{35}$$

ϵ is often called as *distance threshold.*

3.2 Persistent Homology

3.2.1 Simplicial Homology

This section will present the basic notions and properties of *simplicial homology.* But before that, let us dive into why homology is interesting in TDA. One of the most well-known example in topology is that donut and mug are homeomorphic, which is clear that they have the same shape in terms of their surfaces are homeomorphic. But in reality, it is notoriously hard to show that there exists a homeomorphism between two topological spaces (in our case, it would be between two simplicial complexes). Then, one might ask *how can we show whether two or more spaces are homeomorphic or not?* or you can rephrase it into *how can we show whether the shapes of objects are the same or not?* To this question, algebraic topology suggests an approach called *homology*, which is a topological invariant. Hence, if two spaces have different homology groups, we can conclude that two spaces are not homeomorphic. Loosely speaking, the dimension of homology groups gives ideas of connectedness, loops and cavities, which are different dimensional holes. Which implies that to define the homology, we need to describes *holes* in a rigorous manner.

In order to define holes rigorously, we must discuss a boundary operator. The following definitions will firstly be served as preliminaries for the boundary operator and ultimately for homology groups.

Definition 3.6. Let S be a finite set and k be an arbitrary field. Then, a **free k-vector space** generated by S is the vector space $F(S)$ with elements given by formal linear combinations $\sum_{s_i \in S} a_i s_i$ where each $a_i \in k$ and $s_i \in S$.

Definition 3.7 (n-chains)**.** Let $n \geq 0$ be an integer and K be a finite simplicial complex. The **vector space of n-chains** in K is the free $\mathbb{Z}^2$-vector space $C_n(K)$ generated by the n-simplices of K.

By defining the boundary of a n-simplex to be the formal sum of its $(n-1)$-dimensional faces, we get a linear transformation $\partial_n : C_n(K) \to C_{n-1}(K)$. The formal definition of the *boundary operator* is the following.

Definition 3.8 (Boundary operator)**.** The **boundary operator** $\partial_n : C_n(K) \to C_{n-1}(K)$ is the linear transformation defined on simplicies by

$$\partial_n(\{p_0, \dots, p_n\}) = \sum_{i=0}^{n} \{p_0, \dots, \hat{p}_i, \dots, p_n\} \tag{36}$$

where $\{p_0, \dots, \hat{p}_i, \dots, p_n\}$ denotes the $(n-1)$-simplex obtained by omitting the vertex p_i.

Trivially we have $\partial_0 : C_0(K) \to 0$, thus we have the following relationship:

$$\cdots \xrightarrow{\partial_{n+1}} C_n(K) \xrightarrow{\partial_n} C_{n-1}(K) \xrightarrow{\partial_{n-1}} \cdots \xrightarrow{\partial_2} C_1(K) \xrightarrow{\partial_1} C_0(K) \xrightarrow{\partial_0} 0.$$

[9]recall that the diameter of a subset $\sigma \subseteq P$ is $\mathrm{diam}(\sigma) = \max_{p_i, p_j \in P} \|p_i - p_j\|$

The following lemma is the fundamental property that makes it useful to look at homology. Essentially, it tells that a boundary has no boundary.

Lemma 3.1. *The composition $\partial_n \circ \partial_{n+1} = 0$ for all $n \geq 0$.*

Proof. By linearity it is sufficient to show that $\partial_n \circ \partial_{n+1}(\sigma) = 0$ where σ is a $(n+1)$-simplex given by $\sigma = \{p_0, \ldots, p_{n+1}\}$. Then,

$$\begin{aligned}
\partial_n \circ \partial_{n+1}(\sigma) &= \partial_n \left(\sum_{i=0}^{n+1} \{p_0, \ldots, \hat{p}_i, \ldots, p_{n+1}\} \right) \\
&= \sum_i \partial_n(\{p_0, \ldots, \hat{p}_i, \ldots, p_{n+1}\}) \\
&= \sum_i \sum_{j \neq i} \{p_0, \ldots, \hat{p}_i, \ldots, \hat{p}_j, \ldots, p_{n+1}\}
\end{aligned}$$

For $i \neq j$ we see that the $(n-1)$-simplex $\{p_0, \ldots, \hat{p}_i, \ldots, \hat{p}_j, \ldots, p_{n+1}\}$ appears precisely two times in the sum. Since we are working over $\mathbb{Z}_2$, $\partial_n \circ \partial_{n+1}(\sigma) = 0$. □

Definition 3.9 (Cycles and boundaries)**.** We say that $c \in C_n(K)$ is a ***n*-cycle** if $\partial_n(c) = 0$ and we denote the associated vector space of ***n*-cycles** by

$$Z_n(K) := \ker \partial_n = \{c \in C_n(K) : \partial_n(c) = 0\}. \tag{37}$$

The image of ∂_{n+1} for some $d \in C_{n+1}(K)$ is a ***n*-boundary** and we denote the associated vector space of ***n*-boundaries** by

$$B_n(K) = \mathrm{Im}\partial_{n+1} = \{\partial_{n+1}(d) : d \in C_{n+1}(K)\}. \tag{38}$$

Note that lemma 3.1 guarantees that $B_n(K) \subseteq Z_n(K)$. Thus, we now have all the material to define homology.

Definition 3.10 (Homology and Betti numbers)**.** The ***n*-th simplicial homology vector space** of a simplicial complex K is the quotient space

$$H_n(K) = Z_n(K)/B_n(K). \tag{39}$$

The ***n*-th Betti number** represents the dimension of the n-th simplicial homology vector space, i.e., $\beta_n(K) = \dim H_n(K) = \dim Z_n(K) - \dim B_n(K)$ for some simplicial complex K.

Note that 0-th Betti number β_0 represents the number of connected components, first Betti number β_1 represents the number of loops, i.e., one-dimensional holes second Betti number β_2 represents the number of voids, etc. However, we rather want to focus only on 0-th Betti number β_0 since our primary goal is to analyse cluster data and we can view clusters as connected components. To use the fact that $\beta_0(K)$ is equal to the number of path-connected components of $|K|$[10], we must introduce the following lemma.

[10]Let $|K|$ denote the subset of $\mathbb{R}^d$ given by the union of all the simplices in K, equipped with the subspace topology.

Lemma 3.2. *$\beta_0(K)$ equals the number of path-connected components of $|K|$.*

Proof. **Intuition.** If $n = 0$, a cycle is a linear combination of 0-simplices in K, and a boundary is a linear combination of 0-simplices lying in the same connected component of K such that the sum of their coefficients is zero. So Z_0/B_0 is precisely the free abelian group on the connected components of K.

More rigorous proof. Let us order the connected components of $|K|$ from 1 to u and let $\{p_1^i, \ldots, p_{m_i}^i\}$ be the vertex set of the i-th path-connected component for $i = 1, \ldots, u$.

want. $\dim Z_0(K) - \dim B_0(K) = u$.

Note that $Z_0(K) = \ker \partial_0 = \{c \in C_0(K) : \partial_0(c) = 0\}$ and $C_0(K) \xrightarrow{\partial_0} 0$. Hence, the basis for $Z_0(K)$ is given by

$$\bigcup_{i=1}^{u} \{p_1^i, p_2^i + p_1^i, \ldots, p_{m_i}^i + p_1^i\}. \tag{40}$$

Also, note that $B_0(K) = \text{Im}\partial_1 = \{\partial_1(d) : d \in C_1(K)\}$ and $C_1(K) \xrightarrow{\partial_1} C_0(K)$. Similarly, path-connectivity implies that there cannot be a 1-chain c whose boundary is contained in $\{p_1^1, \ldots, p_1^u\}$. Therefore, $\dim Z_0(K) - \dim B_1(K) = u$. □

3.2.2 Persistent Betti Numbers

In the previous section, it is motivated that *homology* is an important tool as it is a topological invariant that measures *connectedness* in a given dimension, i.e., the dimension of the n-th dimensional homology group is the number of n-dimensional *holes* the space has. However, the Betti numbers have a vital issue. That is, for a point set $P \subseteq \mathbb{R}^d$, the Betti numbers $\beta_i(P_r)$ are notoriously unstable under small perturbations of the point set.

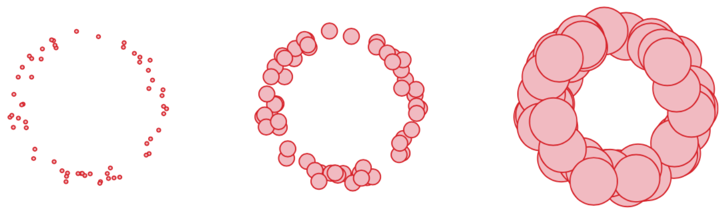


Figure 5: Union of balls $\bigcup_i^n B(X_i, r)$, approximately 60 data points with circularity, with $r = 0.03, 0.10, 0.30$. [33]

That is to say, a slight change of scale r possibly follows totally different Betti numbers. In Figure 5, the 0-th Betti numbers of the left circle must be 60 as all data points are not connected, i.e., there are 60 connected components. However, by increasing scale r, we have completely different 0-th Betti numbers (the right circle has only one connected component). Moreover, there does not exist a perfect scale parameter r that can give robustness to perturbation. Due to this reason, applied algebraic topology proposes a tool, which guarantees *robustness* to perturbation and *scale-free.* In order to compute the *persistent Betti numbers*, we do not only consider a single simplicial complex, but a nested family of subcomplexes of some fixed simplicial complex K called a *filtration* (or a *filtered simplicial complex*).

Definition 3.11 (Simplicial filtration)**.** A filtration of a simplicial complex K is given by

$$K_0 \subseteq K_1 \subseteq \cdots \subseteq K_n = K \tag{41}$$

where $\{K_i\}_{0 \le i \le n-1}$ are subcomplexes of K.

From the filtration, we get maps $f_*^{i,j} : H_k(K_i) \to H_k(K_j)$ for all $i \le j$, that is vector spaces. This property will allow us to apply persistent Betti number for density functions in the mode clustering.

Definition 3.12 (k-th persistent Betti numbers)**.** Consider a filtration $\{K_i\}_{0 \le i \le n}$. Then, the **k-th persistent Betti numbers** are given by

$$\beta_k^{i,j} = \dim \mathrm{Im} f_*^{i,j}. \tag{42}$$

Note that $\beta_k^{i,i} = \beta_k^i = \beta_k(K_i)$. Hence, we can always redeem the ordinary Betti numbers while using persistent Betti numbers.

In TDA, computing the persistence diagram is done in almost everywhere in TDA since we want to graphically represent the persistent homology. The persistence diagram is a neat graphical representation as it shows the significant (high dimensional) topological and geometric information in two dimensional setting. Intuitively, a point $(i, j) \in \mathbb{N} \times \overline{\mathbb{N}}$[11] of the persistence diagram indicates a *topological feature* that is born at K_i and dies at K_j when the simplicial filtration is given by K in (41).

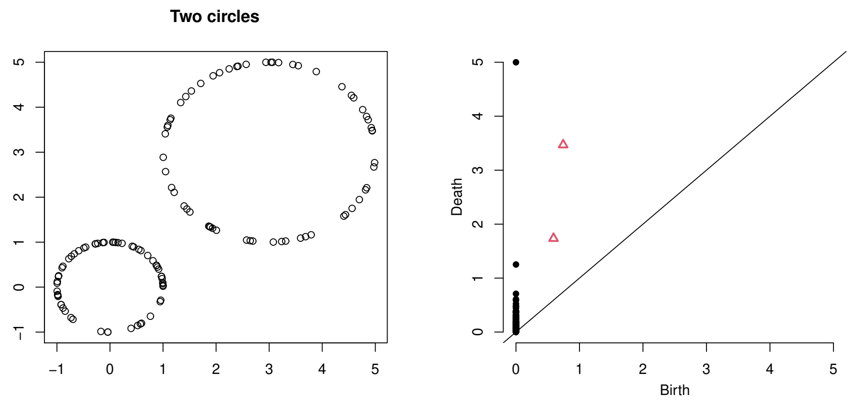


Figure 6: (left) 60 data points generated from two circles; (right) the persistence diagram corresponding to the two circles dataset. Black dots represent the 0-th persistent Betti numbers and red triangles represent the first persistent Betti numbers.

Before giving the definition of persistence diagram, let us introduce the following notation with regard to Definition 3.12: For $i, j \in \mathbb{N} \cup \{-1\}$ with $j > i$,

$$N_k^{i,j} = \begin{cases} 0, & \text{if } i = -1 \\ \beta_k^{i,j-1} - \beta_k^{i,j}, & \text{otherwise.} \end{cases} \tag{43}$$

[11]$\overline{\mathbb{N}} = \mathbb{N} \cup \infty$

Note that $\beta_n^{b,d}$ is the number of classes born before or at K_b that last until K_d. Hence, $N_n^{b,d}$ the number of classes born before or at K_b that die at K_d is equal to the number of classes born before or at K_b that last until K_{d-1} subtracted by the number of classes born before or at K_b that last until K_d (as classes die at K_d means that they last until $d-1$), i.e., $\beta_n^{b,d-1} - \beta_n^{b,d}$, and $N_n^{b,d}$ denotes the number of classes born before or at K_b that die at K_d. Similarly, $N_n^{b-1,d}$ the number of classes born before K_b that die at K_d is $\beta_n^{b-1,d-1} - \beta_n^{b-1,d}$. Therefore, the number of classes born at K_b that die at K_d is $N_n^{b,d} - N_n^{b-1,d}$.

Definition 3.13 (Persistence diagram)**.** Let K be a filtration. Then, the corresponding n-th **persistence diagram** $Dgm_n(K)$ is a finite multi-set of $\mathbb{N} \times \overline{\mathbb{N}}$ where a point $(b,d) \in \mathbb{N} \times \overline{\mathbb{N}}$ with $d > b$. Each point appears at (b,d)

$$\mu_n^{b,d} = N_n^{b,d} - N_n^{b-1,d} \tag{44}$$

if $N_n^{b,d} > N_n^{b-1,d}$. Conventionally, b is called a *birth* and d is called a *death*.

Depending on the context, the birth and death can imply the index of the filtration, time, or even level of the function. But here we assume that the birth and death are the indices of the given filtration.

From the persistence diagram in Figure 6, we can conclude that there is one significant connected component and two significant loops.

3.3 TDA in Mode Clustering

There are sufficiently many ways of identifying clusters in clustering analysis such as using distances, hierarchy, etc. As this thesis solely focuses on density clustering, we identify clusters by *basins of attraction* of the peaks of the density function. Intuitively, consider density function as a terrain. Then, a basin of attraction is the domain of all points flowing into the same local maximum where the flows are defined by the gradient of the density function [11]. Koontz et al already proposed such technique in [21], and in 2002, Comaniciu and Meer followed this notion of clusters and proposed a mean shift algorithm [14], which is one of the most widely-using mode seeking algorithms. However, these techniques have one major flaw.

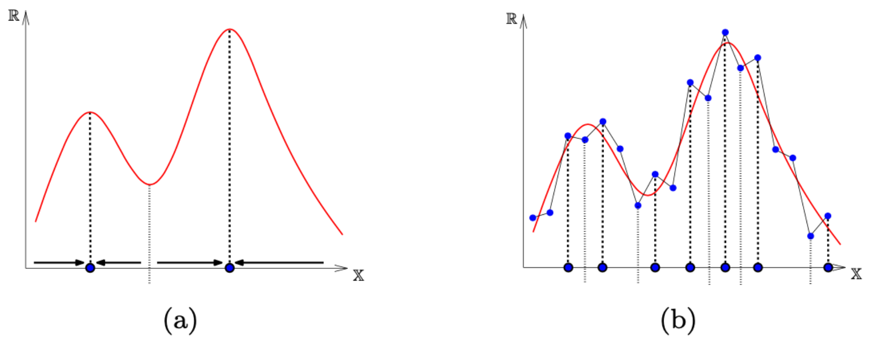


Figure 7: (a) The center-line gives the separation between basins of attraction of two peaks of the density function f; (b) a piecewise-linear approximation and the separating lines of the basins of attraction. [11]

That is, it is highly sensitive to perturbations of the density function. This problem will arise in practice since density-based methods for clustering assumes that data points are drawn from an unknown density function f, thus we have to rely on an estimator $\widehat{f}$ [11]. See Figure 7. It is noticeable that the basins in the piecewise-linear interpolation of f are not necessarily corresponding to the basins in the true density function f.

This section will consider *topological persistence* to solve the problem that is proposed above. This thesis focuses on *density clustering*, which means that we would rather look into the persistent homology of the *superlevel sets filtration* instead of simplicial filtration. A well-known approach is to consider the connected components of the superlevel set $F^\alpha = f^{-1}([\alpha, +\infty))$ for some fixed threshold α as clusters and the rest as noise. However, this technique is not highly responsive to hierarchical dataset due to the fixed threshold [11]. To this problem, *persistence* offers an answer by ranging α from $+\infty$ to $-\infty$ instead of a fixed level α. The following concepts and notions are drawn from [5, 11, 33]

Our ultimate goal is to distinguish which clusters are significant and not significant, and thus be regarded as noise. In density-based clustering, we call this significance as *prominence*. In other words, our goal is to estimate the *prominence* of clusters, or equivalently in density-based techniques, the *prominence* of the density peak. It is questionable how to compute the prominence of the density peak, so let us consider the following figures to figure this out.

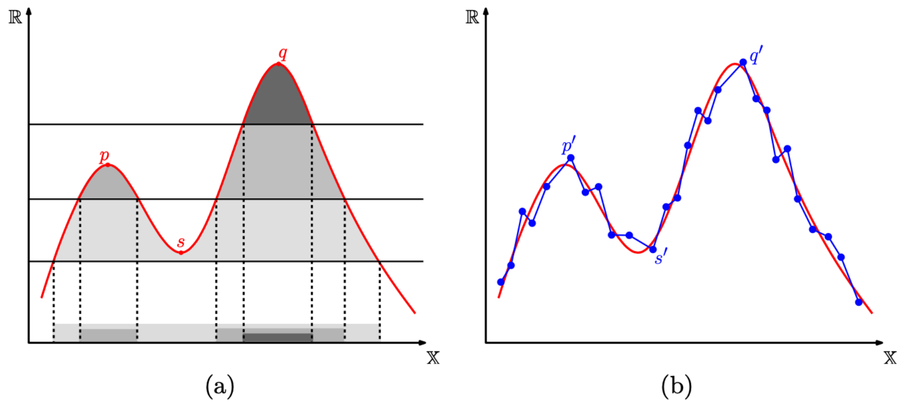


Figure 8: (a) a new connected component, which is born in $F^\alpha = f^{-1}([\alpha, \infty))$ when $\alpha = p$, and C dies in F^α when $\alpha = s$; (b) a piecewise-linear approximation $\widehat{f}$ of f. [11]

It is easily observable from Figure 8 (a) that there are two prominent clusters (or density peaks). Let C denote a new connected component, which is born in $F^\alpha = f^{-1}([\alpha, \infty))$ when $\alpha = p$, and C dies in F^α when $\alpha = s$. Hence, the *prominence* of the density peak p is simply the height difference between birth and death values of C. Obviously, it follows that the larger the height difference between birth and death values is, the more prominent the density peak is. In case of Figure 8, the density peak q is more prominent than p. As mentioned above, the primary goal of clustering is to understand the structure of the density function f while f is unknown. In practice, we thus have an approximation $\widehat{f}$ of f (Figure 8 (b)). Here is an issue to determine the prominence of the density peak. Since now there are more than one local minimum in $\widehat{f}$, a connected component born in F^α when $\alpha = p'$

seems not dying when $\alpha = s'$. But it actually dies when $\alpha = s'$ by the *elder rule*. The *elder rule* states "precedence is given to the class with the earliest birth time" [5] and this rule guarantees to remove the noise from the approximation.

3.4 ToMATo

This thesis will specifically implement a clustering scheme called *Topological Mode Analysis Tool* (ToMATo). ToMATo of Chazal et al [11] pairs the classical *hill climbing algorithm*[12] [21] with persistent homology. As already mentioned above, the hill climbing algorithm proposed by Koontz et al [21] is highly sensitive to perturbations of the density function f. This major flaw of hill climbing algorithm can be improved by computing the *persistence diagram*. Since topological persistence can capture the perturbations arisen from a density estimator $\widehat{f}$, computing the persistence diagram can distinguish the prominent and non-prominent density peaks of $\widehat{f}$.

In section 2, *enhanced mode clustering* has a mathematical threshold n_0 for cluster size given in (32), which let us determine which clusters are actually prominent (or significant) with regards to the size of clusters. Then, one might ask *is there a threshold for persistence to detect the prominent clusters?* Unlike in enhanced mode clustering, ToMATo does not suggest a merging threshold based on the size of clusters, but ToMATo suggests a merging threshold based on the prominence of clusters. That is, we merge a basin of attraction with prominence less than the merging threshold into a neighbour basin of attraction. In practice, we firstly run ToMATo with the *merging threshold* (or *prominence threshold*) $\tau = +\infty$ so that merge every cluster in the dataset. Then based on the computed persistence diagram in the first run, we set an appropriate τ and run ToMATo again to compute the modes and their basins of the attraction. Since the persistence diagram clearly indicates the number of connected components regardless of the dimensions of the data, choosing the merging threshold τ is not a challenging task unless the data is given with much noise. The running time of ToMATo is $O(n\log(n))$ where n is the number of the data points, thus we can consider the algorithm is highly efficient. However, Paris and Durand in [28] presented that the TDA in mode clustering does not achieve good performances in higher dimensions since the topology changes at some saddle points and finding such saddle points is computationally expensive.

In the following sections, we will firstly discuss the continuous case of the ToMATo algorithm. That is, we take the true density function as an input. It will give more intuition of the prominence of the peak without worrying about the density estimation. Secondly, we consider the discrete setting, which is more common in practice. Here, we study the ToMATo algorithm with discrete data points, so we will also study the density estimator as an input in the discrete setting.

3.4.1 The continuous setting

Consider the density function $f : \mathbb{X} \to \mathbb{R}$ is a Morse function where $\mathbb{X}$ is a n-dimensional Riemannian manifold as assumed in section 2. Suppose that f has k local modes $\mathcal{M} = \{m_1, \ldots, m_k\}$. As mentioned already, we consider the superlevel sets of the density function.

[12] The mean shift algorithm is a hill climbing algorithm that seeks modes of a density iteratively instead of explicitly finding a density function [34]

Definition 3.14 (Superlevel sets filtration)**.** A filtration of superlevel sets $F^\alpha = f^{-1}([\alpha, +\infty))$ is given by

$$\{F^\alpha\}_{\alpha \in \mathbb{R}} \tag{45}$$

for α ranging from $+\infty$ to $-\infty$[13].

Since we take the superlevel sets filtration, the birth level is always larger than death level. Hence, the persistence diagram is drawn in lower diagonal, but the idea behind does not change from Definition 3.13.

For any $\alpha \in \mathbb{R}$ and $x \in \mathbb{X}$, $C(m, \alpha) \subseteq F^\alpha$ denotes the connected component containing m at level α. For example, let m_p and m_q be local maximums of the density f that born when $\alpha = p$ and $\alpha = q$ respectively in Figure 8. Also let $b(m_p)$ and $d(m_p)$ denote the birth and death times of the peak m_p (similarly for m_q). Then at $\alpha = d(m_p)$, the connected component $C(m_p, \alpha)$ will be *merged* to $C(m_q, \alpha)$ by the *elder rule*. Formally, we can write it as $m(m_p) = m_q$ where r is a merging map.

Definition 3.15 (Merging map)**.** If $C(m_p, \alpha)$ is merged to $C(m_q, \alpha)$, a merging map $M : \mathbb{X} \to \mathcal{M}$ is defined by

$$M(m_p) = m_q. \tag{46}$$

The prominence can be measured as the following manner.

Definition 3.16 (Prominence)**.** Let m_α be the mode of the density function f at level α. Then, the **prominence** of the mode m_α is defined by

$$\tau = b(m_\alpha) - d(m_\alpha) \geq 0. \tag{47}$$

That is, the difference between the birth and death times. We say that m_α is τ-prominent.

For convenience, let $M_\tau(m_i) = m_j$ if the prominence of m_i is smaller than τ. Note that we can iterate M_τ until it reaches some peak with prominence at least τ, thus let M_τ^* be the converged map, i.e., M_τ^* maps to some peak with prominence at least τ.

As briefly mentioned above, merging threshold restricts the *prominence* of the peak. More precisely, if we set the merging threshold τ, the peaks of f with prominence at least τ are only considered as significant peaks. Hence, the basin of attraction with merging threshold τ can be defined as the following:

Definition 3.17 (Merging basin of attraction)**.** A **merging basin of attraction** of m_j with **merging threshold** τ is defined by

$$B_\tau(m_q) = \bigcup_{M_\tau^*(m_p) = m_q} B(m_p) \tag{48}$$

where the basin of attraction of the peak m_p $B(m_p)$[14] is defined in (3).

Note that $B(m_p) = B_0(m_p) \subseteq B_\tau(m_p)$ for any $m_p \in \mathcal{M}$ and $\tau \geq 0$ and also note that for some point m, $M_\tau^*(m)$ is unique.

[13]A filtration of sublevel sets can be obtained by taking α from $-\infty$ to $+\infty$

[14]The basin of attraction in (3) is equivalent to the *ascending region* in [11]

3.4.2 The discrete setting

As mentioned above, the true density is not given in practice. Hence, we need to estimate the density and there are numerous ways to obtain this estimate. In this thesis, we use the one of the many methods proposed in [11] called *δ-Rips graph* (or *δ-neighborhood graph*). Note that the Rips graph is equivalent to the 1-skeleton of $\mathrm{VR}_\delta(P)$ when P is the given set of points.

As already motivated, the topology is rather uninteresting when the data is given discrete. Thus, we want to construct a graph so that the given discrete points are transformed into a continuous form. Consider δ-Rips graph $R_\delta(\mathcal{J})$ whose vertex set corresponds to the given point set $\mathcal{J} = \{1, \ldots, n\} \subset \mathbb{X}$. Then, each vertex of $R_\delta(\mathcal{J})$ represents the estimated value of the density f at that point, e.g., a vertex i of $R_\delta(\mathcal{J})$ represents the value of $\widehat{f}(i)$ where $\widehat{f}$ is the estimated density. Hence, the pipeline is the following:

1. (Clustering) We sort $\mathcal{J} = \{1, \ldots, n\} \subset \mathbb{X}$ so that $\widehat{f}(1) \geq \widehat{f}(2) \geq \cdots \geq \widehat{f}(n)$. Then, we iterate over every vertex of $R_\delta(\mathcal{J})$ in the decreasing order of $\widehat{f}$-value to find the clusters. Note that before merging with respect to merging threshold, every peak represents a cluster. If the set of neighbours of the vertex i with the indices smaller than i is empty, i is the peak of $\widehat{f}$ within $R_\delta(\mathcal{J})$. Consequently, we obtain a forest of $R_\delta(\mathcal{J})$ where each tree starting from the peak is considered as a basin of attraction of the peak.

2. (Merging) Let $\mathcal{U}$ be a union-find data structure whose entry containing the vertex i is e_i. Then, note that, in graph theory, the merging map defined in (46) is equivalent to the root. Hence, the root of an entry e_i is denoted by $M(e_i)$, i.e., $M(e_i)$ is assigned to the vertex with the highest $\widehat{f}$-value in e_i. We again iterate over every vertex of $R_\delta(\mathcal{J})$ in the decreasing order of $\widehat{f}$-value. Then, there are two cases:

   (a) If the vertex i is the root of the tree T_i, i is attached to the entry e_i in $\mathcal{U}$, where T_i is stored.

   (b) If the vertex i is not the root of the tree T_i, i.e., i belongs to some tree T_j already stored in e_j, we want to check if e_j can be merged into other entries. Iterating over $\ell \in \mathcal{N}$ where $\mathcal{N}$ is the set of neighbours of the vertex i in $R_\delta(\mathcal{J})$ with $\widehat{f}(\ell) \geq \widehat{f}(i)$, we merge e_ℓ into e_j if $\widehat{f}(M(e_\ell)) < \min\left\{\widehat{f}(M(e_j)), \widehat{f}(i) + \tau\right\}$. In other words, we merge e_ℓ into e_j if the peak $M(e_\ell)$ is lower than the peak $M(e_j)$ and the prominence of $M(e_\ell)$ is less than τ. Similarly, we merge e_j into e_ℓ if $\widehat{f}(M(e_j)) < \min\left\{\widehat{f}(M(e_\ell)), \widehat{f}(i) + \tau\right\}$

The following algorithm is copied from [11], which contains both the clustering and merging processes.

Algorithm 1: ToMATo

```
Input: simple graph $R_\delta(\mathcal{J})$ where $\mathcal{J} = \{1, \dots, n\}$ is a given point set, $n$-dimensional
        vector $\widehat{f}$, merging threshold $\tau \geq 0$.
1  Sort $\mathcal{J}$ so that $\widehat{f}(1) \geq \widehat{f}(2) \geq \cdots \geq \widehat{f}(n)$;
2  Initialise a union-find data structure $\mathcal{U}$ and two vectors $g, r$ of size $n$;
3  for $i = 1$ to $n$ do
4      Let $\mathcal{N}$ be the set of neighbours of $i$ in $R_\delta(\mathcal{J})$ that have indices lower than $i$ ;
5      if $\mathcal{N} \neq \emptyset$ then
           // if vertex $i$ is a peak of $\widehat{f}$ within $R_\delta(\mathcal{J})$
6          Create a new entry $e$ in $\mathcal{U}$ and attach vertex $i$ to it;
           // $r(e)$ stores the root vertex associated with the entry $e$
7          $M(e) \leftarrow i$;
8      else
           // if vertex $i$ is not a peak of $\widehat{f}$ within $R_\delta(\mathcal{J})$
           // $g(i)$ stores the approximate gradient at vertex $i$
9          $g(i) \leftarrow \operatorname{argmax}_{j \in \mathcal{N}} \widehat{f}(j)$;
10         $e_i \leftarrow \mathcal{U}.\texttt{find}(g(i))$;
11         Attach vertex $i$ to the entry $e_i$;
12         for $j \in \mathcal{N}$ do
13             $e \leftarrow \mathcal{U}.\texttt{find(j)}$;
14             if $e \neq e_i$ && $\min\left\{\widehat{f}(M(e)), \widehat{f}(M(e_i))\right\} < \widehat{f}(i) + \tau$ then
15                 $\mathcal{U}.\texttt{union}(e, e_i)$;
16                 $M(e \cup e_i) \leftarrow \operatorname{argmax}_{\{M(e), M(e_i)\}} \widehat{f}$;
17                 $e_i \leftarrow e \cup e_i$;
18             end
19         end
20     end
21 end
Output: the collection of entries $e$ of $\mathcal{U}$ such that $\widehat{f}(M(e)) \geq \tau$.
```

Chazal et al [11] argued that we do not have to take δ-Rips graph as a neighbourhood graph and the choice of a graph remains a non-trivial task. They introduced some neighbourhood graphs such as k-nn graph, and Delaunay graphs, which have pros and cons. If one is interested in the choice of a neighbourhood graph in detail, please have a look at [1, 11]. In this thesis, δ-Rips graph will be primarily used for every experiment.

As we can see from Algorithm 1, ToMATo take three inputs: the simple neighbourhood graph $R_\delta(\mathcal{J})$, the n-dimensional density estimator $\widehat{f}$, and the merging threshold τ. Since it is already introduced how the neighbourhood graph works and how the merging threshold is obtained, we now study the choice of density estimator for ToMATo. In short, the choice of density estimator does not affect the result of the ToMATo algorithm unlike other clustering methods that have high dependency on the choice of density estimator. Thus, we can choose an arbitrary density estimator. However, based on many experiments, there are two recommendations of the density estimator in [11]: a truncated Gaussian estimator and the *distance to a measure* introduced in [9].

Definition 3.18 (Truncated Gaussian estimator)**.** Let $X_1, \ldots, X_n$ be i.i.d. samples from the density f. Then, the **truncated Gaussian estimator** with bandwidth $h > 0$ is defined by

$$\widehat{f}_h(x) = \frac{1}{n} \sum_{i=1}^{n} K(\|x - X_i\|) \tag{49}$$

where $K(\cdot)$ is defined by

$$K(\|x - X_i\|) = \begin{cases} \exp\left(-\frac{\|x-X_i\|^2}{2h}\right), & \text{if } \|x - X_i\| \leq h \\ 0, & \text{otherwise.} \end{cases} \tag{50}$$

Definition 3.19 (Distance to a measure)**.** Let $X_1, \ldots, X_n$ be i.i.d. samples from the density f. Then, the **distance to a measure** of the k nearest neighbours is defined by

$$\widehat{f}_k(x) = \sqrt{\frac{1}{k} \sum_{i=1}^{k} \|x - p_i\|^2} \tag{51}$$

where p_i is the i-th nearest neighbour of x among the given samples $\{X_i\}_{1 \leq i \leq n}$.

Note that the *distance to a measure* $\widehat{f}_k(x)$ is a distance not a density. Hence, $-f$ is taken as an input for ToMATo since high density indicates small nearest-neighbor distances.

3.4.3 Application on Synthetic Data

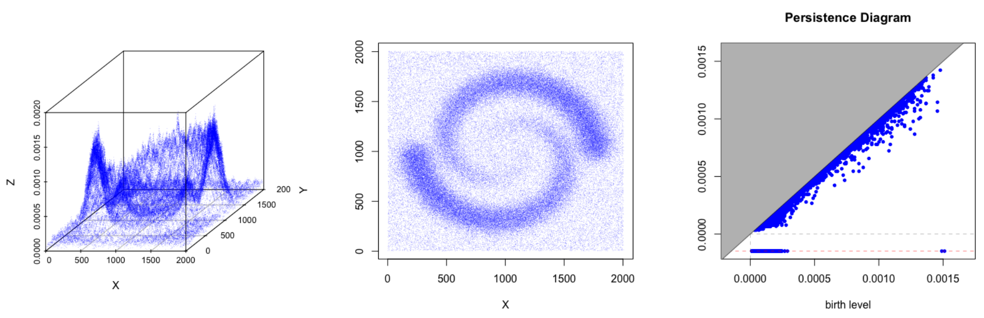


Figure 9: (left) spiral dataset with density (114, 561 points); (middle) 2 dimensional projection of the spiral dataset; (right) persistence diagram computed with $\tau = +\infty$.

In this section, we consider spiral dataset consisting of 114, 561 data points sampled from two spirals in $\mathbb{R}^3$. Note that the data lays on $\mathbb{R}^3$, not $\mathbb{R}^2$ since it also includes the density information, which can be observed from Figure 9 (left). As already mentioned in Algorithm 1, the ToMATo took three inputs: neighbourhood graph $R_\delta(\mathcal{J})$, density estimator $\widehat{f}$, and merging threshold τ. In the first run, we took $\delta = 10$ for δ-Rips graph, and $\tau = +\infty$ (the true density is already given, hence no need to use density estimator). As a result, we could compute the persistence diagram. As shown in Figure 9 (right), there are two points infinitely far, thus we can conclude that there are two prominent clusters in the spiral dataset. We re-run the algorithm but with the merging threshold $\tau = 10^{-3}$ based on the persistence diagram in the first run.

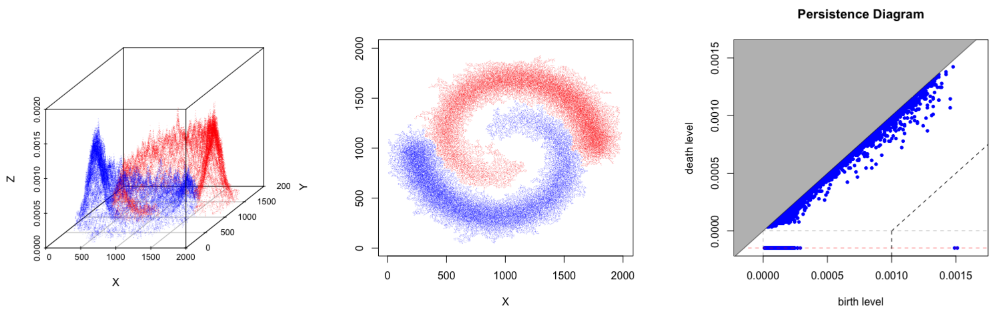


Figure 10: (left) clustered spiral dataset with density; (middle) 2 dimensional projection of the clustered spiral dataset; (right) persistence diagram computed with $\tau = 10^{-3}$.

From Figure 10, one can observe that the prominence of two points are larger than $\tau = 10^{-3}$ and the prominent clusters are well separated from the noise. As already mentioned in the beginning of 3, the total computation time of ToMATo is $O(n + m\log(n))$ and $m = O(n)$ when δ of δ-Rips graph is small enough. Since we took $\delta = 10$, we can assume that the time complexity of the algorithm is approximately $O(n\log(n))$.

4 Comparison on Real Dataset

In order to compare the performance of each clustering methods that we discussed in section 2 and section 3, we need to decide which metrics to use to measure the quality of clustering. In practice, it is not a trivial task to assess the performance of clustering since the clustering analysis is widely used in unsupervised learning, i.e., the clustering analysis mainly deals with the unlabelled data. To prevent such a problem, the dataset used in this section is limited to the data with ground truth labels. Hence, to compare the performance of clustering methods, we can use the *adjusted Rand index* (ARI) introduced in [29]. In short, the adjusted Rand index measures the similarity of two different assignment of labels to the clusters. Thus, the higher value of the adjusted Rand index indicates the more similar of two label assignments are, e.g., if ARI is equal to 1, then two assignments are exactly the same. Note that ARI ranges from -1 to $+1$. Negative ARI between the result assignment and ground truth labels implies worse performance of the clustering than random partition. Meaning that it is pointless to use the clustering. For more rigorous definition of the adjusted Rand index, given a set S with $|S| = n$, let $X = \{X_1, \dots, X_p\}$ and $Y = \{Y_1, \dots, Y_q\}$ be the partitions of S. Let n_{ij} be the number of overlapping elements of X_i and Y_i, i.e., $n_{ij} = |X_i \cap Y_j|$. For simplicity, let $a_i = \sum_{k=1}^{q} n_{ik}$ and $b_j = \sum_{k=1}^{p} n_{kj}$. Consider the following contingency table:

	Y_1	Y_2	$\cdots$	Y_q	sums
X_1	n_{11}	n_{12}	$\cdots$	n_{1q}	a_1
X_2	n_{21}	n_{22}	$\cdots$	n_{2q}	a_2
$\vdots$	$\vdots$	$\vdots$	$\ddots$	$\vdots$	$\vdots$
X_p	n_{p1}	n_{p2}	$\cdots$	n_{pq}	a_p
sums	b_1	b_2	$\cdots$	b_q	n

Then, the adjusted Rand index is defined by

$$ARI = \frac{\sum_{ij} \binom{n_{ij}}{2} - \left(\sum_i \binom{a_i}{2} \sum_j \binom{b_j}{2}\right) / \binom{n}{2}}{\frac{1}{2}\left(\sum_i \binom{a_i}{2} + \sum_j \binom{b_j}{2}\right) - \left(\sum_i \binom{a_i}{2} \sum_j \binom{b_j}{2}\right) / \binom{n}{2}}. \tag{52}$$

To dive into the intuition behind the adjusted Rand index, let us define the Rand index

$$RI = \frac{a+b}{a+b+c+d} = \frac{a+b}{\binom{n}{2}} \tag{53}$$

where

- a is the number of pairs of elements that are both in the same respective cluster of each partition X and Y;
- b is the number of pairs of elements that are both in the different respective cluster of each partition X and Y;
- c is the number of pairs of elements that are both in the same respective cluster of partition X, but in the different respective cluster of partition Y;
- d is the number of pairs of elements that are both in the same respective cluster of partition Y, but in the different respective cluster of partition X.

In short, $a+b$ is the number of agreements between X and Y and $c+d$ is the number of disagreements between X and Y. However, the Rand index cannot consider the possibility of agreement by chance between X and Y [8]. To improve this problem, Hubert and Arabie [19] found $\mathbb{E}[RI]$. That is, the expectation of the Rand index when the partitions are random. Then, (52) can be written with respect to $\mathbb{E}[RI]$:

$$ARI = \frac{RI - \mathbb{E}[RI]}{1 - \mathbb{E}[RI]}. \tag{54}$$

For example, assume that two partitions are equal, i.e., $X = Y = \{X_1, \dots, X_r\}$. Then, we have

$$n_{ij} = \begin{cases} |X_i| & \text{if } i = j \\ 0 & \text{if } i \neq j. \end{cases}$$
$$a_i = b_i = n_{ii} \quad \text{for } i = 1, \dots, r$$

Hence, we have

$$\begin{aligned} ARI &= \frac{\sum_{ij}\binom{n_{ij}}{2} - \left(\sum_i \binom{a_i}{2} \sum_j \binom{b_j}{2}\right)/\binom{n}{2}}{\frac{1}{2}\left(\sum_i \binom{a_i}{2} + \sum_j \binom{b_j}{2}\right) - \left(\sum_i \binom{a_i}{2} \sum_j \binom{b_j}{2}\right)/\binom{n}{2}} \\ &= \frac{\sum_i \binom{n_{ii}}{2} - (\sum_i \binom{n_{ii}}{2} \sum_i \binom{n_{ii}}{2})/\binom{n}{2}}{\sum_i \binom{n_{ii}}{2} - (\sum_i \binom{n_{ii}}{2} \sum_i \binom{n_{ii}}{2})/\binom{n}{2}} = 1. \end{aligned}$$

Therefore, we check that if two partitions are exactly the same, the ARI value is equal to 1.

More information of the adjusted Rand index can be found in [19, 29, 32].

This section will implement both the enhanced mode clustering (EMC) and Topological Mode Analysis Tool (ToMATo) in the three different datasets used in [13]. The ultimate goal of this section is to reproduce the results from [13] and show how ToMATo works in high dimensions settings. Note that before we run both the EMC and ToMATo, the data is always normalised to use the simplified Silverman's rule of thumb (31) and for the fairness, i.e., ToMATo does not use the Silverman's rule of thumb, however it might affect the result if I do not normalise the data for the ToMATo but for EMC.

In the following experiments, the EMC implementation was done in R and the ToMATo implementation was done in C++. In each run, only one core was used and the specification of the computer used in the following experiments is: Processor is 1.6 GHz dual core Intel core i5 and 8 GB of RAM. Notice that the two implementations of EMC and ToMATo are done in different programming languages (R and C++), thus it is rather uninteresting to compare the time of computations in the following experiments. Hence, please have a look at the time complexity of two methods mentioned in Section 2 and Section 3.

4.1 Olive oil data

In this section, we will apply both the EMC and ToMATo to the olive oil data in [16]. The olive oil data is 10 dimensional data with 572 observations. Since this data already consisted of ground truth labels (produced areas) and the information of the produced regions such as north and south, the clustering analysis was done with the other 8 attributes: *palmitic, palmitoleic, stearic, oleic, linoleic, linolenic, arachidic, eicosenoic.*

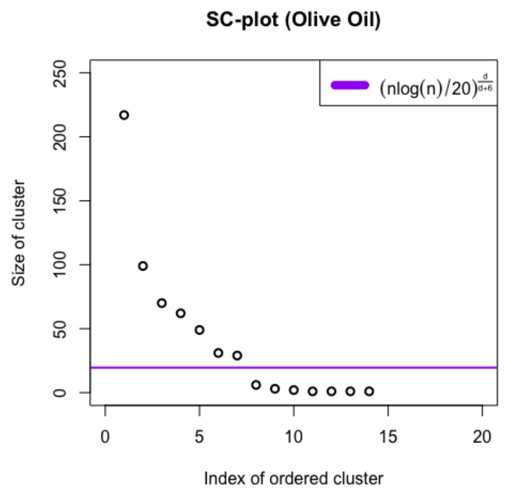


Figure 11: The size of clusters plot of the olive oil dataset with the denoising threshold $n_0 = 19.54$ and bandwidth parameter $h = 0.587$.

According to Figure 11, by the denoising threshold (32), insignificant clusters are merged to the closest significant clusters. After the denoising process, the EMC yielded 7 remaining significant clusters while there are 9 true clusters: *North-Apulia, Calabria, South-Apulia, Sicily, Inland-Sardinia, Coast-Sardinia, East-Liguria, West-Liguria, Umbria.*

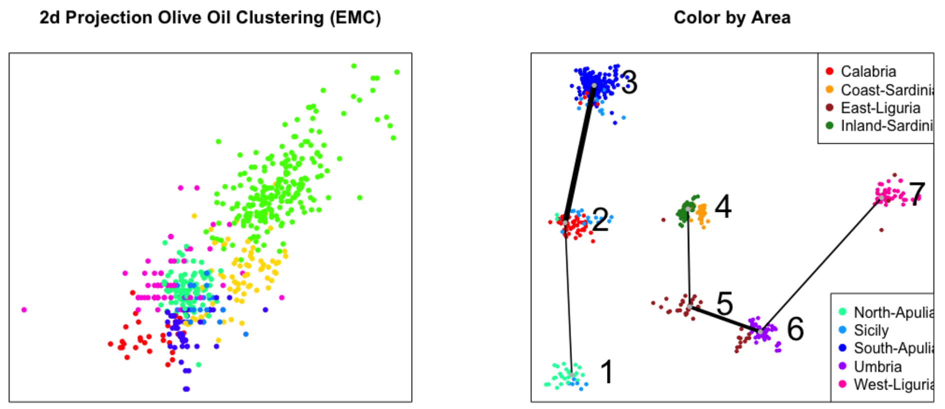


Figure 12: (left) 2 dimensional projection of the clustered olive oil data by EMC. It is the projection on first and second columns; (right) the connectivity plot of the olive oil data. Note that the thicker line between two clusters implies the higher connectivity between two clusters.

For Figure 12 (left), one can visualise performing PCA instead of projecting on the first and second column. However due to the high dimensions of the dataset, it does not change the visual representation too much. Thus, this thesis visualised the clustering result by the projection on two columns of the given dataset.

The connectivity plot from Figure 12 (right) shows that there are high connectivity between cluster $1, 2, 3$ as Sicily is spread out over cluster $1, 2, 3$. Also, it is illustrated in Figure 12 (right) that East-Liguria is spread out over cluster $4, 5, 6, 7$ though it is mostly assigned to cluster 5.

Confusion matrix (EMC):

Area	1	2	3	4	5	6	7
Calabria	0	51	5	0	0	0	0
Coast-Sardinia	0	0	0	33	0	0	0
East-Liguria	0	0	0	1	32	11	6
Inland-Sardinia	0	0	0	65	0	0	0
North-Apulia	23	2	0	0	0	0	0
Sicily	6	18	12	0	0	0	0
South-Apulia	0	0	206	0	0	0	0
Umbria	0	0	0	0	0	51	0
West-Liguria	0	0	0	0	0	0	50

The above confusion matrix (area versus cluster) gives more precise understanding of the connectivity. From the matrix, one can observe that 6 points from cluster 1, 18 points from cluster 2, and 12 points from cluster 3 are from Sicily while cluster $1, 2, 3$ are mostly assigned to North-Apulia, Calabria and South-Apulia. We could conclude that this happened due to the produced region of these areas, i.e., North-Apulia, Calabria and South-Apulia are located at the south side of Italy. The other 4 clusters are commonly located either at the north side of Italy or Sardinia.

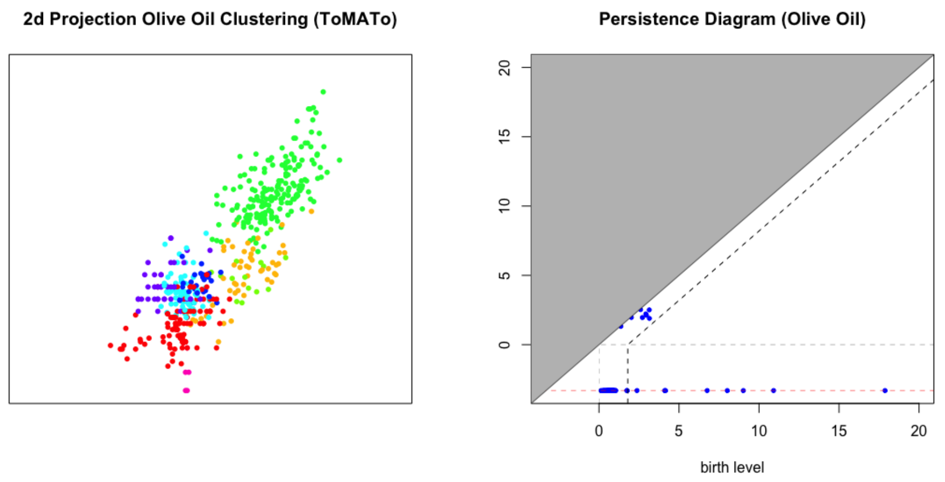


Figure 13: (left) 2 dimensional projection of the clustered olive oil data by ToMATo; (right) the persistence diagram of the olive oil data. Rips parameter $\delta = 1$ for the Rips graph, $k = 5$ for the distance to a measure, $\tau = 1.8$ for the merging threshold.

Note that Figure 13 (left) is similar to Figure 12. Meaning that two methods separated the data into the similar clusters. From Figure 13 (right), we do observe that there are 8 prominent peaks (i.e., clusters) and the merging threshold τ could separate the 8 prominent peaks from topological noise. Here, it is not very clear how to set τ. This thesis distinguished the topological noise near the diagonal line and the noise chunk whose birth level is near 0. For more details, let us have a look at the confusion matrix of the ToMATo algorithm.

Confusion matrix (ToMATo):

Area	1	2	3	4	5	6	7	8
Calabria	0	33	13	2	0	0	0	0
Coast-Sardinia	0	0	0	0	5	28	0	0
East-Liguria	37	0	0	0	0	0	2	0
Inland-Sardinia	0	0	0	0	64	0	0	0
North-Apulia	20	0	0	0	0	0	0	0
Sicily	6	18	1	5	0	0	0	0
South-Apulia	0	1	0	183	1	0	0	0
Umbria	46	0	0	0	0	0	0	5
West-Liguria	0	0	0	0	0	0	37	0

From the result of ToMATo, Sicily is again spread out over cluster $1, 2, 3, 4$. However, one can observe that cluster $1, 2, 3, 4$ correspond to Unbria, Calabria, Calabria, South-Apulia respectively, which does not show any correspondence between the produce areas and regions (although Apulia is located at the south side of Italy and Liguria is at the north side of Italy, cluster 1 contains both Apulia areas and east-Liguria). However, ToMATo could separate

Sardinia areas from the areas located at north side better than the result of EMC. Lastly, the confusion matrix also captures the small connectivity between west-Liguria and east-Linguria. This suggests that both methods could capture hidden region information and produce area based on the chemical measurements.

4.2 Banknote authentication data

To perform both the EMC and ToMATo, banknote authentication data given in [3] were used. The banknote authentication data were extracted from the images of both real and forged banknote examples. To extract the features out of the images, a Wavelet Transform technique was applied. The banknote authentication data contains 5 variables: *variance of Wavelet Transformed image, skewness of Wavelet Transformed image, curtosis of Wavelet Transformed image, entropy of image, class.* Since the class attribute contains the ground truth labels: *real, forgery*, it needs to be excluded before performing the clustering methods. Thus, the clustering analysis was done with 4 attributes: *variance of Wavelet Transformed image, skewness of Wavelet Transformed image, curtosis of Wavelet Transformed image, entropy of image.*

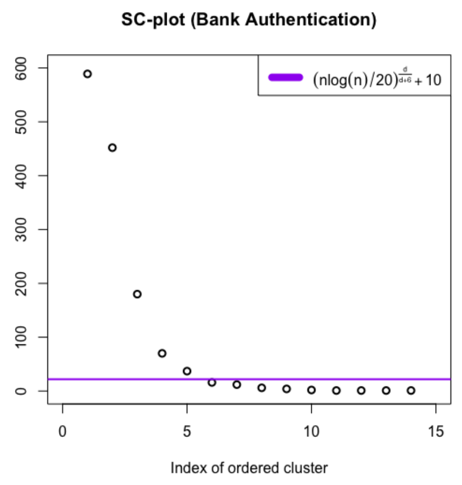


Figure 14: The size of clusters plot of the bank authentication dataset with the denoising threshold $n_0 = 21.97$ and bandwidth parameter $h = 0.453$.

Figure 14 illustrates that there are 5 significant clusters and $n_0 + 10$ term could denoise the small clusters. Here, instead of the normal denoising threshold n_0, $n_0 + 10$ was used since there was a considerable size difference between the fifth (size: 37) and sixth (size: 16) clusters.

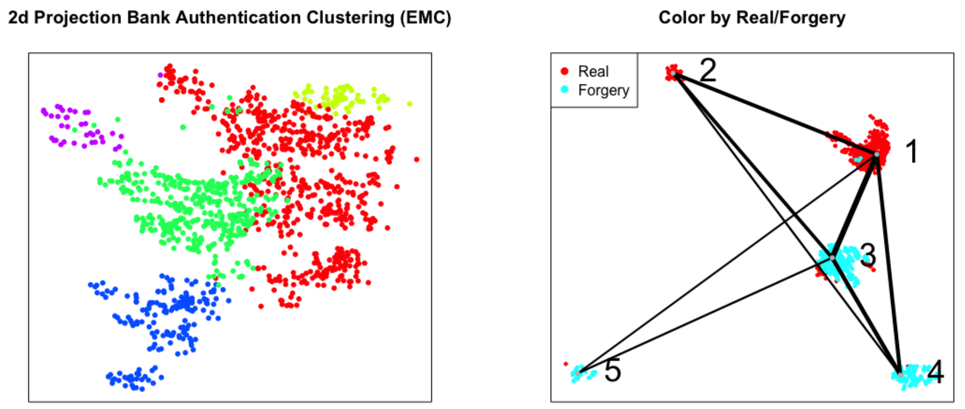


Figure 15: (left) 2 dimensional projection of the clustered bank authentication data by EMC; (right) the connectivity plot of the bank authentication data. Note that the thicker line between two clusters implies the higher connectivity between two clusters.

From Figure 15 (right), we do observe that the connectivity exists in every cluster. The connectivity is especially strong between cluster $1, 2, 3$ and between cluster $3, 4, 5$. From this, we can hypothesise that although cluster $1, 2$ mainly represent the real banknotes, cluster 3 also contains some real banknotes.

Confusion matrix (EMC):

Class	1	2	3	4	5
Real	631	70	59	1	1
Forgery	4	0	390	179	37

From the above confusion matrix (class versus cluster), we can confirm our hypothesis that cluster 3 contains some real banknotes as 59 real banknotes were assigned to cluster 3. However, except 59 real banknotes assigned to cluster 3, the real banknotes are overall well separated from the forged banknotes.

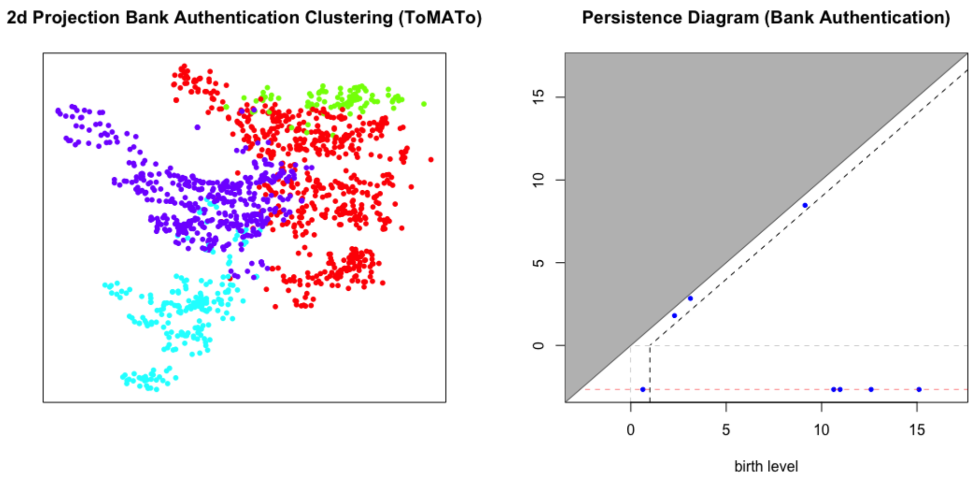


Figure 16: (left) 2 dimensional projection of the clustered bank authentication data by ToMATo; (right) the persistence diagram of the bank authentication data. Rips parameter $\delta = 1$ for the Rips graph, $k = 35$ for the distance to a measure, $\tau = 1$ for the merging threshold.

Figure 15 (left) and Figure 16 (left) share very similar separated shape except that the ToMATo result divided the data into 4 clusters while EMC divided the data into 5 clusters. From Figure 16 (right) we can observe that there are 4 prominent peaks that are far off the diagonal line. The merging threshold $\tau = 1$ successfully separated the 4 prominent peaks from topological noise near the diagonal line.

Confusion matrix (ToMATo):

Class	1	2	3	4
Real	604	97	25	35
Forgery	1	0	191	418

The above confusion matrix (class versus cluster) illustrates that cluster $1, 2$ correspond to the real banknotes and cluster $3, 4$ correspond to the forged banknotes. Unlike the result of EMC, both cluster $3, 4$ consists of the real banknotes while they corresponds to the forged banknotes. However, we can still conclude that cluster $3, 4$ are corresponding to the forgery since cluster 3 consists of 191 forgeries while only 25 are real. Likewise, cluster 4 consists of 418 forgeries while only 35 are real. We can conclude that real and forged banknotes are, in fact, distinguishable based on Wavelet Transformed images as each class are divided well in both methods.

4.3 Red wine quality data

To perform both the EMC and ToMATo methods, wine quality data given in [3] were used. The wine quality data contains both red and white wine samples, thus only red wine samples are chosen to use due for performance time. The red wine quality data contains 1599 observations and 12 variables: *fixed acidity, volatile acidity, citric acid, residual sugar,* *chlorides, free sulfur dioxide, total sulfur dioxide, density, pH, sulphates, alcohol, quality* whereas the quality attribute is the ground truth. Thus, the clustering analysis was done with 11 attributes: *fixed acidity, volatile acidity, citric acid, residual sugar, chlorides, free sulfur dioxide, total sulfur dioxide, density, pH, sulphates, alcohol.* Note that there are 6 different wine quality scores as $3, 4, 5, 6, 7, 8$ where higher score indicates the better wine it is. More information of the wine quality data can be found in [15].

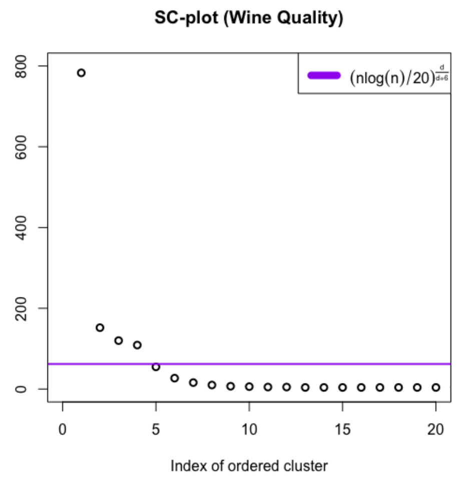
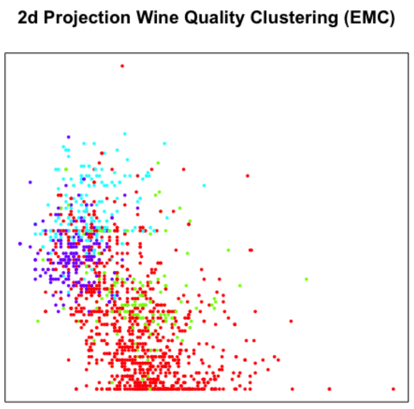


Figure 17: The size of clusters plot of the red wine quality dataset with the denoising threshold $n_0 = 62.06$ and bandwidth parameter $h = 0.599$.

Figure 17 yields that there are 4 significant clusters and the normal denoising threshold n_0 could denoise small clusters. Unfortunately, 4 significant clusters are much less than 6 true clusters. The EMC did not perform well in the sense of the number of clusters since the wine quality score is very subjective, i.e., the score may vary depending on the person who rates the wine quality. Let us see if the connectivity plot gives more idea of the data.



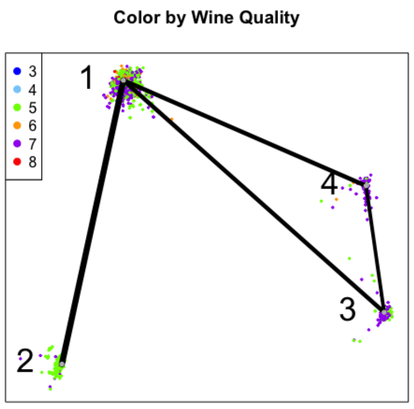


Figure 18: (left) 2 dimensional projection of the clustered red wine quality data by EMC; (right) the connectivity plot of the red wine quality data. Note that the thicker line between two clusters implies the higher connectivity between two clusters.

Figure 18 (right) shows that the majority of quality 5 wines are assigned to cluster $1, 2$ and also cluster 2 does not contain much high quality wines, e.g., quality $7, 8$ wines. This explains that there is no high connection between cluster 2 and cluster $3, 4$.

Confusion matrix (EMC):

Quality	1	2	3	4
3	10	0	0	0
4	49	0	1	3
5	488	135	39	19
6	434	25	91	88
7	68	3	48	80
8	5	0	5	8

As we could already see from Figure 18 (right), cluster 2 does not contain many high quality wines, but more of middle quality wines such as quality 5 and 6. Cluster 1 does not have a visible structure as it contains all quality wines. However, cluster $3, 4$ tend to contain only middle-high quality wines.

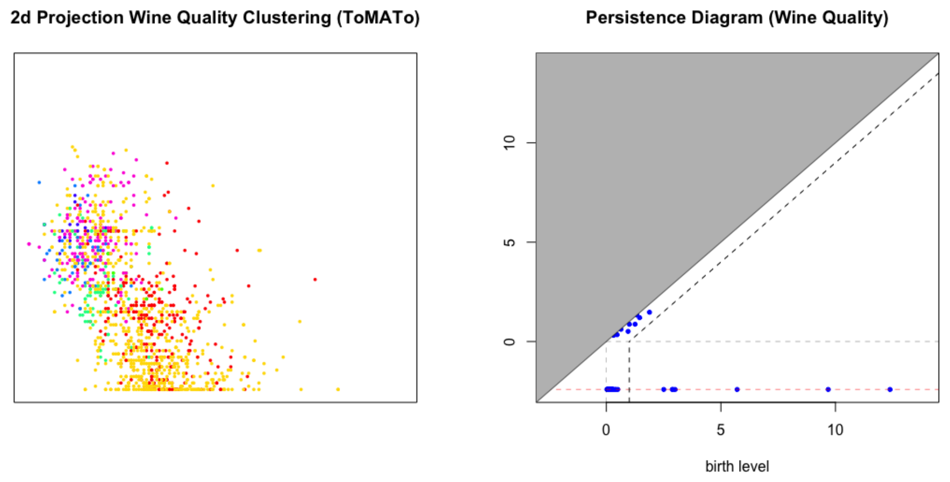


Figure 19: (left) 2 dimensional projection of the clustered red wine quality data by ToMATo; (right) the persistence diagram of the red wine quality data. Rips parameter $\delta = 2$ for the Rips graph, $k = 5$ for the distance to a measure, $\tau = 1$ for the merging threshold.

Both Figure 18 (left) and Figure 19 (left) have distinguishable figures since ToMATo could divide the red wine samples into 7 significant clusters as we do observe from Figure 19 (right). Figure 19 (right) suggests that there are 7 prominent peaks and the merging threshold $\tau = 1$ separated the 7 prominent peaks from topological noise near the diagonal line and noise whose birth level is near 0.

Confusion matrix (ToMATo):

Quality	1	2	3	4	5	6	7
3	0	7	0	0	0	0	0
4	6	35	0	2	0	0	2
5	200	342	10	41	6	0	12
6	70	381	1	39	27	13	65
7	3	75	2	6	18	6	74
8	0	9	0	0	0	0	6

Similarly but worse than in EMC, each result cluster does not show any tendency. Only cluster $2, 7$ contain the high quality wines (quality 8), however cluster $2, 7$ contain the low quality wines (quality 4) simultaneously.

4.4 Seed data

To perform both the EMC and ToMATo methods, seed data given in [3] were used. To extract the features of wheat grains, a soft X-ray technique was used. The seed data contains 210 wheat grains and each wheat grain belongs to three different varieties: *Kama, Rosa, Canadian*. There are 7 attributes from the images taken by the soft X-ray: *area, perimeter, compactness, length of kernel, width of kernel, asymmetry coefficient, length of kernel groove*. The clustering analysis was done with these 7 attributes.

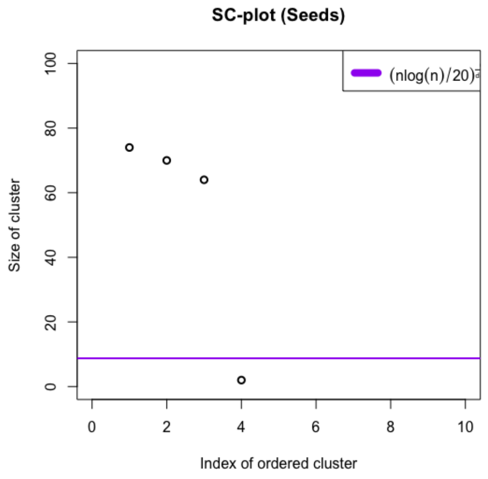


Figure 20: The size of clusters plot of the seed dataset with the denoising threshold $n_0 = 8.75$ and bandwidth parameter $h = 0.613$.

Figure 20 yields that there are 3 significant clusters, which exactly correspond to the true clusters: *Kama, Rosa, Canadian*. The denoising threshold n_0 could denoise one cluster noise with size of 2. As the third largest cluster has size 64, n_0 successfully separated the significant clusters from cluster noise.

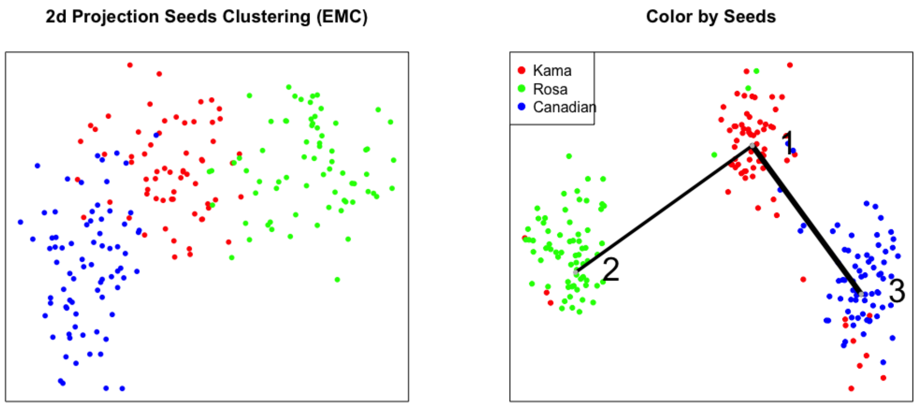


Figure 21: (left) 2 dimensional projection of the clustered seed data by EMC; (right) the connectivity plot of the seed data. Note that the thicker line between two clusters implies the higher connectivity between two clusters.

From Figure 21 (right), one can observe that each result cluster of EMC is assigned to one of varieties, e.g., cluster 1 corresponds to Kama, cluster 2 corresponds to Rosa, and cluster 3 corresponds to Canadian. Also, it is shown that there exist high connectivity between cluster pair $(1, 2)$ and cluster pair $(1, 3)$ and cluster 2. These connectivity's are also illustrated in Figure 21 (left) as cluster 1 is located in the middle of cluster 2 and cluster 3.

Confusion matrix (EMC):

Variety	1	2	3
Kama	58	3	9
Rosa	3	67	0
Canadian	3	0	67

From the above confusion matrix (variety versus cluster), we can confirm our observation from Figure 21. Cluster $1, 2, 3$ are clearly corresponding to Kama, Rosa, and Canadian respectively. Note that cluster 1 contains both the wheat grains from Rosa and Canadian, which explains the connectivity from Figure 21 (right).

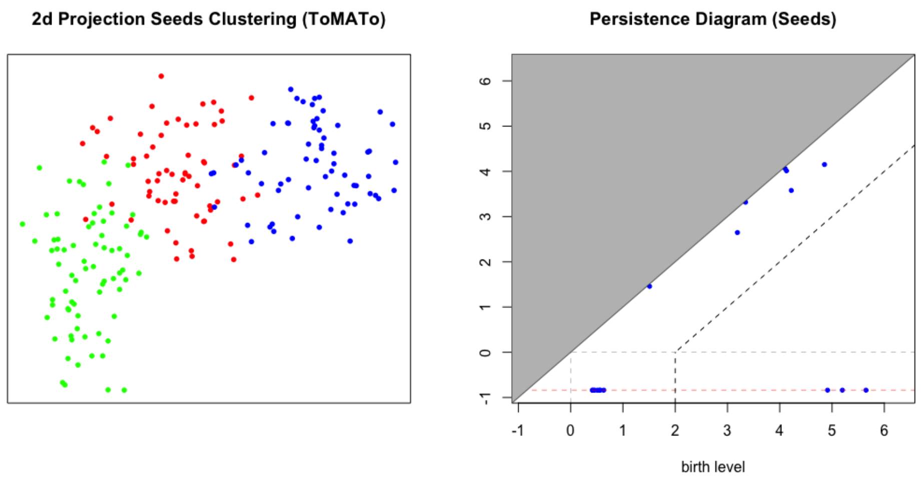


Figure 22: (left) 2 dimensional projection of the clustered seed data by ToMATo; (right) the persistence diagram of the seed data. Rips parameter $\delta = 1$ for the Rips graph, $k = 5$ for the distance to a measure, $\tau = 2$ for the merging threshold.

Figure 21 (left) and Figure 22 (left) have almost exactly the same clustered shapes. Figure 22 (right) shows that there are 3 prominent peaks, i.e., 3 significant clusters, which exactly correspond to the given qualities. These three prominent peaks are distinguishable far off the diagonal line. Meaning that these peaks are highly prominent.

Confusion matrix (ToMATo):

Variety	1	2	3
Kama	57	8	1
Rosa	4	0	64
Canadian	2	66	0

The confusion matrix for ToMATo is similar to the confusion matrix for EMC. Again, cluster

$1, 2, 3$ are clearly corresponding to Kama, Canadian, and Rosa and cluster 1 has connectivity with both cluster 2 and cluster 3.

4.5 Comparison

In this section, we compare the EMC and ToMATo based on adjusted Rand index (ARI) obtained from the previous experiments. Here, the ARI was computed between the ground true labels and the result assignment. As the ARI computes the similarity between two assignments, the higher ARI will directly imply the better performance of the clustering method.

Dataset/Method	EMC	ToMATo	k-means
Olive Oil	**0.826**	0.810	0.787
Bank Authentication	0.563	**0.571**	0.013
Red wine Quality	**0.073**	0.056	0.052
Seed	0.765	0.794	**0.797**

The above table contains information of the ARI of both clustering method in 4 different dataset. In the olive oil data and red wine quality data, the EMC performed better than ToMATo. In the bank authentication data and seed data, the ToMATo performed better than EMC. Notice that the olive oil data and red wine quality data are higher dimensional data compared to bank authentication data and seed data. From this, we do observe that the EMC performs slightly better than ToMATo in higher dimensions. However, notice that the difference of the ARI is insignificant between the ARI of the EMC and ToMATo, thus both methods performed well overall. Our observation (the EMC works better on high dimensional data than ToMATo) can be explained by saddle points problem mentioned in [28]. Recall from Section 3.4 that the topology changes at some saddle points and these changes of topology might affect the performance of clustering.

Compared to k-means, both the EMC and ToMATo performed better in almost every experiments except for the seed data. For k-means, k is set to be equal to the number of the true clusters from ground truth, which is the optimal value for the k-means. By doing so, k-means would perform the best so that we can compare the results of the EMC and ToMATo with the best results of k-means. Thus, the results of k-means have the advantage of having prior information of the correct parameter while the EMC and ToMATo do not have. Note that the k-means fails to give good quality result of clustering when the clusters have non-convex shapes. This disadvantage of the k-means followed poor ARI for the bank authentication data as the clusters have non-convex shapes in Figure 15 (left) and Figure 16 (left). k-means showed better performance for the seed data due to the convex shapes of the clusters for the seed data. However, the ARI does not vary too much to conclude that the k-means outperforms compared to EMC and ToMATo.

5 Conclusion

Throughout this thesis, classical density-based clustering was improved in two different directions: Enhanced Mode Clustering and Topological Mode Analysis Tool introduced in [13] and [11] respectively. The classical density-based clustering methods such as mean-shift in [14] and classical hill climbing algorithm in [21] have one major disadvantage that we opted to improve. That is, the density-based clustering fails to give good quality clustering results in high dimensions. However, the density-based clustering can be used in any shapes of clusters unlike, for example, k-means that is the most commonly used clustering method. Two clustering methods introduced in Section 2 and 3 improve this remaining disadvantage and keep the advantages of the density-based method.

The core contents of this thesis start by giving ideas of mode clustering and enhancements to mode clustering in Section 2. The enhanced mode clustering from Section 2 focused on the following rooms for improvement of the mean shift algorithm in [14]: 1. Mode clustering is a branch of hard clustering. Which implies that the connectivity between clusters cannot be measured. 2. The choice of bandwidth parameter of kernel density estimator is rather unclear in practice. 3. In high dimensions, the variance creates small clusters in the kernel density estimator. Hence, we want to merge these small clusters, especially in the high dimensional setting. To improve these, Section 2 presented the notions and properties of (soft) mode clustering, measuring clustering connectivity using soft assignment vectors, Silverman's rule of thumb to select the optimal bandwidth of the kernel density estimator, and denoising threshold to denoise small clusters created in high dimensions. Additionally, the consistency of estimated modes was studied in Section 2.4. The result of Theorem 2.2 gives two significant results for the enhanced mode clustering. The first result shows that the number of estimated modes is equal to the number of true local modes asymptotically, i.e., the number of estimated modes is closer to the number of true local modes by increasing the size of the data. The second result shows that the estimated modes are close to true modes asymptotically distance-wise. Combining two results tells us that if the data is given large enough, the estimation of the local modes converges in probability to the true local modes. Before applying it to real datasets, it was applied to synthetic data to see if everything worked well.

Following enhanced mode clustering, which is a geometric approach to the mode clustering, the topological mode analysis tool is presented in Section 3. The topology can be applied to the mode clustering mainly due to Lemma 3.2. Lemma 3.2 shows that the dimension of 0-th homology group is equal to the number of connected components. That is, the number of clusters can be computed by the 0-th homology. The topological mode analysis tool is the improved result of combining the classical hill climbing algorithm introduced in [21] with topological persistence. The classical hill climbing algorithm has two major issues: 1. Hill climbing algorithm is highly sensitive to perturbations. 2. Hill climbing algorithm uses the absolute height to measure the prominence, i.e., significance, of the clusters. That is a very unstable measure since the small clusters generated from the density estimation can cause small bumps that might have high absolute height in the density function, however, it is not significant. The solutions to these problems are given in persistent homology. From theory, the topological mode analysis tool is better than enhanced mode clustering with regards to computation time. However, note that the study of topological data analysis in density-based clustering is in the early stages. Thus, for example, the choice of hyperparameters (e.g., the merging threshold τ, the radius of the δ-Rips graph) still needs to be studied in the future.

Last but not least, the two mode clustering approaches, which originated from two different fields of mathematics, were applied to the real datasets and compared with respect to the adjusted Rand index in Section 4. Unfortunately, two implementations were done in different programming languages, so comparing the running time is pointless. However, as the time complexities of the two clustering methods were already given in Section 2 and Section 3, we conclude that the running time in the topological mode analysis tool is less than in enhanced mode clustering. The adjusted Rand indices gave a broad idea of clustering quality for two clustering approaches. For all of the experiments, both methods resulted in similar high adjusted Rand indices. Meaning that the performances of the two clustering methods are overall good except in the red wine quality data. It was shown from the bank authentication dataset that both mode- and persistence-based clustering work effectively with non-convex-shaped clusters while k-means failed to give good quality clustering output. Both clustering methods yielded a better adjusted Rand index for the red wine quality data than in k-means, but they still showed poor performance due to the noisiness of the data. The hyperparameters of the enhanced mode clustering were selected based on the Silverman's rule of thumb in (29) and the suggested denoising threshold in (32) while the hyperparameters (except the merging threshold τ) of the topological mode analysis tool were chosen by trial-and-error, which is one of the disadvantages compared to the enhanced mode clustering. Although we choose δ by trial-by-error, it is recommended to take δ small enough so that the time complexity can be reduced. Finally, we note that both improved methods of density-based clustering can now show good performances in the high dimensional setting. Recall that the conventional density-based clustering schemes such as mean shift and hill climbing algorithms do not scale well with high dimensional data. However, a series of results presented in Section 4 and the adjusted Rand index show that the high dimension problem has been fixed. Ultimately, the results obtained in this thesis contradict the remark from [13] (see page 221) where the authors found that denoising small clusters using persistence homology is inefficient in high dimensions.

References

[1] Hirokazu Anai, Frédéric Chazal, Marc Glisse, Yuichi Ike, Hiroya Inakoshi, Raphaël Tinarrage, and Yuhei Umeda. DTM-based filtrations. In *Topological Data Analysis*, pages 33–66. Springer, 2020.

[2] Ery Arias-Castro, David Mason, and Bruno Pelletier. On the estimation of the gradient lines of a density and the consistency of the mean-shift algorithm. *The Journal of Machine Learning Research*, 17(1):1487–1514, 2016.

[3] Arthur Asuncion and David Newman. Uci machine learning repository, 2007.

[4] Jean-Daniel Boissonnat and Monique Teillaud. *Effective Computational Geometry for Curves and Surfaces*. Springer, 2006.

[5] Magnus Bakke Botnan. Topological data analysis. https://www.few.vu.nl/~botnan/lecture_notes.pdf, 2020.

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

[7] José E Chacón, Tarn Duong, and MP Wand. Asymptotics for general multivariate kernel density derivative estimators. *Statistica Sinica*, pages 807–840, 2011.

[8] José E Chacón and Ana I Rastrojo. Minimum adjusted rand index for two clusterings of a given size. *arXiv preprint arXiv:2002.03677*, 2020.

[9] Frédéric Chazal, David Cohen-Steiner, and Quentin Mérigot. Geometric inference for measures based on distance functions. *Foundations of Computational Mathematics*, 11(6):733–751, 2011.

[10] Frédéric Chazal, Brittany Fasy, Fabrizio Lecci, Bertrand Michel, Alessandro Rinaldo, Alessandro Rinaldo, and Larry Wasserman. Robust topological inference: Distance to a measure and kernel distance. *The Journal of Machine Learning Research*, 18(1):5845–5884, 2017.

[11] Frédéric Chazal, Leonidas J Guibas, Steve Y Oudot, and Primoz Skraba. Persistence-based clustering in Riemannian manifolds. *Journal of the ACM (JACM)*, 60(6):1–38, 2013.

[12] Yen-Chi Chen. Statistical machine learning: Clustering. http://faculty.washington.edu/yenchic/19A_stat535/Lec8_clustering.pdf, 2019.

[13] Yen-Chi Chen, Christopher R Genovese, and Larry Wasserman. A comprehensive approach to mode clustering. *arXiv preprint arXiv:1406.1780*, 2014.

[14] Dorin Comaniciu and Peter Meer. Mean shift: A robust approach toward feature space analysis. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 24(5):603–619, 2002.

[15] Paulo Cortez, António Cerdeira, Fernando Almeida, Telmo Matos, and José Reis. Modeling wine preferences by data mining from physicochemical properties. *Decision Support Systems*, 47(4):547–553, 2009.

[16] Michele Forina, C Armanino, Sergio Lanteri, and E Tiscornia. Classification of olive oils from their fatty acid composition. In *Food Research and Data Analysis: Proceedings from the IUFoST Symposium, September 20-23, 1982, Oslo, Norway/edited by H. Martens and H. Russwurm, Jr.* London: Applied Science Publishers, 1983., 1983.

[17] Evarist Giné and Armelle Guillou. Rates of strong uniform consistency for multivariate kernel density estimators. 38(6):907–921, 2002.

[18] Allen Hatcher. *Algebraic Topology.* Cambridge University Press, 2002.

[19] Lawrence Hubert and Phipps Arabie. Comparing partitions. *Journal of Classification*, 2(1):193–218, 1985.

[20] Kim Jisu. *Statistical Inference for Geometric Data.* PhD thesis, Yale University, 2018.

[21] Warren L. G. Koontz, Patrenahalli M. Narendra, and Keinosuke Fukunaga. A graph-theoretic approach to nonparametric cluster analysis. *IEEE Computer Architecture Letters*, 25(09):936–944, 1976.

[22] Hans-Peter Kriegel, Peer Kröger, Jörg Sander, and Arthur Zimek. Density-based clustering. *Wiley Interdisciplinary Reviews: Data Mining and Knowledge Discovery*, 1(3):231–240, 2011.

[23] Geoffrey J McLachlan, Sharon X Lee, and Suren I Rathnayake. Finite mixture models. *Annual Review of Statistics and Its Application*, 6:355–378, 2019.

[24] John Milnor. *Morse Theory.(AM-51), Volume 51*, volume 51. Princeton University Press, 2016.

[25] Marston Morse. Relations between the critical points of a real function of n independent variables. *Transactions of the American Mathematical Society*, 27(3):345–396, 1925.

[26] Marston Morse. The foundations of a theory of the calculus of variations in the large in m-space (second paper). *Transactions of the American Mathematical Society*, 32(4):599–631, 1930.

[27] James R Munkres. *Elements of Algebraic Topology.* CRC press, 2018.

[28] Sylvain Paris and Frédo Durand. A topological approach to hierarchical segmentation using mean shift. In *2007 IEEE Conference on Computer Vision and Pattern Recognition*, pages 1–8. IEEE, 2007.

[29] William M Rand. Objective criteria for the evaluation of clustering methods. *Journal of the American Statistical Association*, 66(336):846–850, 1971.

[30] David W Scott. *Multivariate Density Estimation: Theory, Practice, and Visualization.* John Wiley & Sons, 2015.

[31] Simon J Sheather. Density estimation. *Statistical Science*, pages 588–597, 2004.

[32] Nguyen Xuan Vinh, Julien Epps, and James Bailey. Information theoretic measures for clusterings comparison: Variants, properties, normalization and correction for chance. *The Journal of Machine Learning Research*, 11:2837–2854, 2010.

[33] Larry Wasserman. Topological data analysis. *Annual Review of Statistics and Its Application*, 5:501–532, 2018.

[34] Changjiang Yang, Ramani Duraiswami, Daniel DeMenthon, and Larry Davis. Mean-shift analysis using quasinewton methods. In *Proceedings 2003 International Conference on Image Processing (Cat. No. 03CH37429)*, volume 2, pages II–447. IEEE, 2003.

A Implementation

R implementation of the Enhanced Mode Clustering (EMC) can be found in https://github.com/yenchic/EMC from the authors of [13]. However, you might encounter some errors in generating the size of cluster plot (SC-plot). Then, please see https://github.com/HyunminHong/EMC where the slight adjustment on the SC-plot. For C++ implementation of the Topological Mode Analysis Tool (ToMATo), ToMATo codes and the introduction of the codes (written by the authors of [11]) can be found in https://github.com/locklin/tomato.

A.1 Visualisation

Plots from Figure 6 are generated by:

```r
library(TDA)

# Generate two circles
Circle1 <- circleUnif(60)
Circle2 <- circleUnif(60, r = 2) + 3
Circles <- rbind(Circle1, Circle2)

plot(Circles, xlab = "", ylab = "", main = "Two circles")

maxscale <- 5  # limit of the filtration
maxdimension <- 1  # components and loops

# Compute the persistence diagram of the Rips filtration built on two circles
    data
DiagRips <- ripsDiag(X = Circles, maxdimension, maxscale,
                     library = c("GUDHI", "Dionysus"), location = TRUE)

plot(DiagRips[["diagram"]])
```

In order to plot the persistence pairs of the superlevel filtration, the following codes are used:

```r
setwd("~/Downloads/ToMATo/")

library(grDevices)

diagram <- read.csv("diagram.txt", header = F, sep = " ")

local_minmax = function(diagram) {
    ## min_birth = interval min; max_birth = interval max
    # corresponds to birth level
    unique_val = unique(diagram[,1])
    min_birth = min(unique_val)
    max_birth = max(unique_val)

    ## min_death = non-infinity min of range; max_death = range max
    # corresponds to death level
    fin_entry = which(is.finite(diagram[,2]))
    min_death = min(diagram[fin_entry, 2])
    max_death = max(diagram[fin_entry,2])

    return(c(min_death, max_death, min_birth, max_birth))
}

```

```r
replace_infinity = function(diagram, inf_delta = 0.618) {
    # infinite row entry in birth
    inf_birth = which(is.infinite(diagram[,1]))
    # infinite row entry in death
    inf_death = which(is.infinite(diagram[,2]))

    # find local min and max
    min_death = local_minmax(diagram)[1]
    max_death = local_minmax(diagram)[2]
    min_birth = local_minmax(diagram)[3]
    max_birth = local_minmax(diagram)[4]

    # replace infinity values with more readable value
    delta = (min_death - max_birth) * inf_delta

    # if all entries of birth are finite, return the original
    diagram[inf_death, 2] = delta

    return(diagram)
}

display_diagram = function(diagram, tau = NaN, inf_delta = 0.618, sp = 0.1,
    alpha = 0) {
    # convert infinity values to plotable values
    diagram = replace_infinity(diagram, inf_delta)

    # find local min and max
    min_death = local_minmax(diagram)[1]
    max_death = local_minmax(diagram)[2]
    min_birth = local_minmax(diagram)[3]
    max_birth = local_minmax(diagram)[4]

    # increasing sp yields wider plot and vice versa
    step_size = abs((min_death - max_birth) * sp)

    # range max and min
    Rmax = max(max_birth + step_size, max_death + step_size)
    Rmin = min(min_birth - step_size, min_death)

    plot(diagram[,1], diagram[,2], pch = 19, cex = 0.6, col = "blue",
         xlim = c(Rmin, Rmax),
         ylim = c(Rmin, Rmax),
         xlab = "birth level", ylab = "death level",
         main = "Persistence Diagram")

    abline(a = 0, b = 1, lty = 1, lwd = 0.3)
    abline(h=0, v=0, lty = 2, lwd = 0.3)
    abline(h = min(diagram[,2]), lty = 8, lwd = 1, col = adjustcolor("red",
    alpha.f = 0.5))
    polygon(x = c(min(diagram[,2]) - 10*step_size, max(diagram[,2]) + 10*step
    _size, min(diagram[,2]) - 10*step_size),
            y = c(min(diagram[,2]) - 10*step_size, max(diagram[,2]) + 10*step
    _size, max(diagram[,2]) + 10*step_size),
            col = "grey",
            border = adjustcolor("black", alpha.f = 0.5))

    if (!is.nan(tau)) {
        clip(x1 = tau, x2 = Rmax + alpha, y1 = 0, y2 = Rmax + alpha)
        abline(a = -tau, b = 1, lty = 8, lwd = 1)
        clip(x1 = min(diagram)*10, x2 = max(diagram), y1 = min(diagram)*10,
    y2 = 0)
```

```r
        abline(v = tau, lty = 8, lwd = 1)
    }
}
# display_diagram(diagram, tau = 2, inf_delta = 0.2, alpha = 1) # example
```

To change the dotted bar ($-\infty$-line) lower, you can increase the value of `inf_delta`. The larger value of `alpha` makes sure the merging threshold τ line longer to the right side.

A.2 Comparison on Real Dataset Plots

Olive oil dataset:

```r
setwd("/Users/hyunminhong/Documents/Programmings/R_R studio/Bachelor Thesis/
    EMC-master/")
source("EMC.R")
library(scatterplot3d)
library(pdfCluster)

# import the olive oil data used in Chen et al
library(freqparcoord)
data(oliveoils)

olive = scale(oliveoils[,3:10]) # extract numeric data
                                # scale the data so that we can use the
    simplified Silverman's rule of thumb

# the ground truth labels
ground_truth = as.numeric(oliveoils$Area)

#################################################################
########### EMC ################################################
#################################################################

# Enhanced mode clustering
olive_EMC = EMC(olive)


# SC-plot
plot(olive_EMC$SC.plot, type = "p", lwd = 2,
     xlim = c(0,20), ylim = c(0,250),
     xlab = "Index of ordered cluster",
     ylab = "Size of cluster",
     main = "SC-plot (Olive Oil)"
)
abline(h = olive_EMC$size.threshold, lwd = 2, col = "purple") # results that
    there are 7 clusters
legend("topright", expression((n*log(n)/20)^{frac(d,d+6)}), col="purple", lwd
    =8, cex=1)


# plot clusters
n_emc <- 7 # the number of clusters
palette_emc <- rainbow(n_emc) # generate n distinct colors to plot the
    clusters
palette_emc = palette_emc[olive_EMC$labels] # assign each point to the color
    where colors represent the clusters

plot(olive[,1:2], pch = 16, cex = 0.8, col = palette_emc,
     xlab = "", ylab = "", xaxt="n", yaxt="n", main = "2d Projection Olive
    Oil Clustering (EMC)") # plot clusters (2d projection)
```

```

scatterplot3d(olive[,1:3], pch = 16, type = "p",
              cex.symbols = 0.8, color = palette_emc,
              xlab = "X", ylab = "Y", zlab = "Z") # plot clusters

palette <- rainbow(max(ground_truth))
palette[3] <- "brown"
palette[4] <- "forestgreen"

plot(olive_EMC, pch=20, cex=0.7,
     xlab="", ylab="", main="Color by Area",
     xaxt="n", yaxt="n", txt.pos=4, col=palette[ground_truth])

legend("topright", levels(oliveoils[,1])[1:4], col=palette[1:4], pch=rep
    (19,9),cex=1)
legend("bottomright", levels(oliveoils[,1])[5:9], col=palette[5:9], pch=rep
    (19,9),cex=1)

# compute the adjusted Rand index
adj.rand.index(olive_EMC$labels, ground_truth) # results in 0.8260962

# confusion matrix
confusion_mat_emc <- table(ground_truth, olive_EMC$labels)
row.names(confusion_mat_emc) <- levels(oliveoils[,1])
colnames(confusion_mat_emc) <- c(1:nrow(olive_EMC$modes))
confusion_mat_emc

#################################################################
############ ToMATo #############################################
#################################################################

ToMATo_labels <- read.csv("/Users/hyunminhong/Desktop/Bachelor Thesis/
    Information_TDA/clusters_olive.txt", header = F, sep = " ")
ToMATo_labels <- as.numeric(unlist(ToMATo_labels))

# plot clusters
n <- 8 # the number of clusters
palette_tom <- rainbow(n) # generate n distinct colors to plot the clusters
palette_tom = palette_tom[ToMATo_labels] # assign each point to the color
    where colors represent the clusters

plot(olive[,1:2], pch = 16, cex = 0.8, col = palette_tom,
     xlab = "", ylab = "", xaxt="n", yaxt="n", main = "2d Projection Olive
    Oil Clustering (ToMATo)") # plot clusters (2d projection)

scatterplot3d(olive[,1:3], pch = 16, type = "p",
              cex.symbols = 0.8, color = palette_tom,
              xlab = "X", ylab = "Y", zlab = "Z") # plot clusters

# compute the adjusted Rand index
adj.rand.index(ToMATo_labels, ground_truth) # results in 0.8100825

# confusion matrix
confusion_mat_tom <- table(ground_truth, ToMATo_labels)
row.names(confusion_mat_tom) <- levels(oliveoils[,1])
colnames(confusion_mat_tom) <- c(1:max(na.omit(ToMATo_labels)))
confusion_mat_tom

table(ToMATo_labels, ground_truth)

#################################################################
```

```r
############ k-means ############################################
###############################################################

olive_kmean <- kmeans(olive, 9)

# compute the adjusted Rand index
adj.rand.index(ground_truth, olive_kmean$cluster) # results in 0.7871385
```

Banknote authentication dataset:

```r
library(nonet)
library(pdfCluster)
setwd("/Users/hyunminhong/Documents/Programmings/R_R studio/Bachelor Thesis/
    EMC-master/")
source("EMC.R")

data(banknote_authentication)

# write.table(banknotes, file = "banknotes_w_o_labels.txt", sep = " ",
#             row.names = FALSE, col.names = FALSE)

# the ground truth labels
ground_truth = banknote_authentication$class

banknotes <- scale(banknote_authentication[,1:4])

###############################################################
############ EMC ################################################
###############################################################

# Enhanced mode clustering
banknotes_EMC <- EMC(banknotes)


# SC-plot
plot(banknotes_EMC$SC.plot, type = "p", lwd = 2,
     xlim = c(0,15), ylim = c(0,600),
     xlab = "Index of ordered cluster",
     ylab = "Size of cluster",
     main = "SC-plot (Bank Authentication)",
)
abline(h = banknotes_EMC$size.threshold+10, lwd = 2, col = "purple") #
    results that there are 5 clusters
                                                                     # used
    denoising threshold is 21.97
legend("topright", expression((n*log(n)/20)^{frac(d,d+6)}+10), col="purple",
    lwd=8, cex=1)


# plot clusters
n_emc <- 5 # the number of clusters
palette_emc <- rainbow(n_emc) # generate n distinct colors to plot the
    clusters
palette_emc = palette_emc[banknotes_EMC$labels] # assign each point to the
    color where colors represent the clusters

plot(banknotes[,1:2], pch = 16, cex = 0.8, col = palette_emc,
     xlab = "", ylab = "", xaxt="n", yaxt="n", main = "2d Projection Bank
    Authentication Clustering (EMC)") # plot clusters (2d projection)

palette <- rainbow(max(ground_truth))
```

```r

plot(banknotes_EMC, pch=20, cex=0.7,
     xlab="", ylab="", main="Color by Real/Forgery",
     xaxt="n", yaxt="n", txt.pos=4, col=palette[ground_truth])

legend("topleft", c("Real", "Forgery"), col=palette, pch=rep(19,2), cex=1)


# compute the adjusted Rand index
adj.rand.index(ground_truth, banknotes_EMC$labels) # results in 0.563353


# confusion matrix
confusion_mat_emc <- table(ground_truth, banknotes_EMC$labels)
row.names(confusion_mat_emc) <- c("Real", "Forgery")
colnames(confusion_mat_emc) <- c(1:nrow(banknotes_EMC$modes))
confusion_mat_emc

##############################################################
############ ToMATo ##########################################
##############################################################

ToMATo_labels <- read.csv("~/Downloads/ToMATo/clusters.txt", header = F, sep
    = " ")
ToMATo_labels <- as.numeric(unlist(ToMATo_labels))

# plot clusters
n <- 4 # the number of clusters
palette_tom <- rainbow(n) # generate n distinct colors to plot the clusters
palette_tom = palette_tom[ToMATo_labels] # assign each point to the color
    where colors represent the clusters

plot(banknotes[,1:2], pch = 16, cex = 0.8, col = palette_tom,
     xlab = "", ylab = "", xaxt="n", yaxt="n", main = "2d Projection Bank
    Authentication Clustering (ToMATo)") # plot clusters (2d projection)

# compute the adjusted Rand index
adj.rand.index(ToMATo_labels, ground_truth) # results in 0.5714921

# confusion matrix
confusion_mat_tom <- table(ground_truth, ToMATo_labels)
row.names(confusion_mat_tom) <- c("Real", "Forgery")
colnames(confusion_mat_tom) <- c(1:max(na.omit(ToMATo_labels)))
confusion_mat_tom

table(ToMATo_labels, ground_truth)

##############################################################
############ k-means #########################################
##############################################################

bank_kmean <- kmeans(banknotes, 2)

# compute the adjusted Rand index
adj.rand.index(ground_truth, bank_kmean$cluster) # results in 0.01321583
```

Red wine quality dataset:

```r
library(tidyverse)
library(nonet)
library(pdfCluster)
```

```r
setwd("/Users/hyunminhong/Documents/Programmings/R_R studio/Bachelor Thesis/
    EMC-master/")
source("EMC.R")

wine <- read.csv("/Users/hyunminhong/Downloads/ToMATo/inputs/wine.txt",
    header = T, row.names = 1, sep = ",")

sort(unique(wine$quality)) # 3 4 5 6 7 8 (ground truth)

# the ground truth labels
ground_truth = wine$quality

wine <- scale(wine[,-12])

#write.table(wine, file = "wine_w_o_labels.txt", sep = " ",
#             row.names = FALSE, col.names = FALSE)



#################################################################
########### EMC ################################################
#################################################################

# Enhanced mode clustering
wine_EMC <- EMC(wine)


# SC-plot
plot(wine_EMC$SC.plot, type = "p", lwd = 2,
     xlim = c(0,20), ylim = c(0,800),
     xlab = "Index of ordered cluster",
     ylab = "Size of cluster",
     main = "SC-plot (Wine Quality)",
)
abline(h = wine_EMC$size.threshold, lwd = 2, col = "purple") # results that
    there are 4 clusters
# used denoising threshold is 21.97
legend("topright", expression((n*log(n)/20)^{frac(d,d+6)}), col="purple", lwd
    =8, cex=1)


# plot clusters
n_emc <- 4 # the number of clusters
palette_emc <- rainbow(n_emc) # generate n distinct colors to plot the
    clusters
palette_emc = palette_emc[wine_EMC$labels] # assign each point to the color
    where colors represent the clusters

plot(wine[,2:3], pch = 16, cex = 0.5, col = palette_emc,
     xlab = "", ylab = "", xaxt="n", yaxt="n", main = "2d Projection Wine
    Quality Clustering (EMC)") # plot clusters (2d projection)

palette <- rainbow(length(unique(ground_truth)))
palette[1] <- "blue"
palette[2] <- "lightskyblue"
palette[3] <- "red"
palette[4] <- "orange"
palette[5] <- "lawngreen"
palette[6] <- "purple"

plot(wine_EMC, pch=19, cex=0.3,
     xlab="", ylab="", main="Color by Wine Quality",
     xaxt="n", yaxt="n", txt.pos=2, col=palette[ground_truth])

sort(unique(wine$quality))

legend("topleft", c("3","4","5","6","7","8"), col=palette[c(1,2,5,4,6,3)],
    pch=rep(19,6), cex=1)


# compute the adjusted Rand index
adj.rand.index(ground_truth, wine_EMC$labels) # results in 0.07330196


# confusion matrix
confusion_mat_emc <- table(ground_truth, wine_EMC$labels)
row.names(confusion_mat_emc) <- c(3,4,5,6,7,8)
colnames(confusion_mat_emc) <- c(1:nrow(wine_EMC$modes))
confusion_mat_emc

#################################################################
############ ToMATo #############################################
#################################################################

ToMATo_labels <- read.csv("~/Downloads/ToMATo/clusters.txt", header = F, sep
    = " ")
ToMATo_labels <- as.numeric(unlist(ToMATo_labels))

# plot clusters
n <- 7 # the number of clusters
palette_tom <- rainbow(n) # generate n distinct colors to plot the clusters
palette_tom = palette_tom[ToMATo_labels] # assign each point to the color
    where colors represent the clusters

plot(wine[,2:3], pch = 16, cex = 0.5, col = palette_tom,
     xlab = "", ylab = "", xaxt="n", yaxt="n", main = "2d Projection Wine
    Quality Clustering (ToMATo)") # plot clusters (2d projection)

# compute the adjusted Rand index
adj.rand.index(ground_truth, ToMATo_labels) # results in 0.05554523

# confusion matrix
confusion_mat_tom <- table(ground_truth, ToMATo_labels)
row.names(confusion_mat_tom) <- c(3,4,5,6,7,8)
colnames(confusion_mat_tom) <- c(1:max(na.omit(ToMATo_labels)))
confusion_mat_tom

#################################################################
############ k-means ############################################
#################################################################

wine_kmean <- kmeans(wine, 6)

# compute the adjusted Rand index
adj.rand.index(ground_truth, wine_kmean$cluster) # results in 0.05159205
```

Seed dataset:

```r
library(mclust)
library(datasetsICR)
library(pdfCluster)
library(scatterplot3d)
```

```r
setwd("/Users/hyunminhong/Documents/Programmings/R_R studio/Bachelor Thesis/EMC-master/")
source("EMC.R")

data(seeds)

unique(seeds$variety) # Kama Rosa Canadian (ground truth)

# the ground truth labels
ground_truth = as.numeric(seeds$variety)

seeds <- scale(seeds[,-8])

#write.table(seeds, file = "seeds_w_o_labels.txt", sep = " ",
#             row.names = FALSE, col.names = FALSE)



#################################################################
############ EMC ################################################
#################################################################

# Enhanced mode clustering
seed_EMC <- EMC(seeds)


# SC-plot
plot(seed_EMC$SC.plot, type = "p", lwd = 2,
     xlim = c(0,10), ylim = c(0,100),
     xlab = "Index of ordered cluster",
     ylab = "Size of cluster",
     main = "SC-plot (Seeds)",
)
abline(h = seed_EMC$size.threshold, lwd = 2, col = "purple") # results that there are 3 clusters
legend("topright", expression((n*log(n)/20)^{frac(d,d+6)}), col="purple", lwd=8, cex=1)


# plot clusters
n_emc <- 3 # the number of clusters
palette_emc <- rainbow(n_emc) # generate n distinct colors to plot the clusters
palette_emc = palette_emc[seed_EMC$labels] # assign each point to the color where colors represent the clusters

plot(seeds[,2:3], pch = 16, cex = 0.8, col = palette_emc,
     xlab = "", ylab = "", xaxt="n", yaxt="n", main = "2d Projection Seeds Clustering (EMC)") # plot clusters (2d projection)

scatterplot3d(seeds[,1:3], pch = 16, type = "p",
              cex.symbols = 0.8, color = palette_emc,
              xlab = "X", ylab = "Y", zlab = "Z", main = "3d Projection Seeds Clustering (EMC)")

palette <- rainbow(length(unique(ground_truth)))

plot(seed_EMC, pch=19, cex=0.7,
     xlab="", ylab="", main="Color by Seeds",
     xaxt="n", yaxt="n", txt.pos=4, col=palette[ground_truth])

```

```r
legend("topleft", levels(seeds$variety), col=palette, pch=rep(19,6), cex=1)


# compute the adjusted Rand index
adj.rand.index(ground_truth, seed_EMC$labels) # results in 0.7647773


# confusion matrix
confusion_mat_emc <- table(ground_truth, seed_EMC$labels)
row.names(confusion_mat_emc) <- levels(seeds$variety)
colnames(confusion_mat_emc) <- c(1:nrow(seed_EMC$modes))
confusion_mat_emc

#################################################################
############ ToMATo #############################################
#################################################################

ToMATo_labels <- read.csv("~/Downloads/ToMATo/clusters.txt", header = F, sep
    = " ")
ToMATo_labels <- as.numeric(unlist(ToMATo_labels))

# plot clusters
n <- 3 # the number of clusters
palette_tom <- rainbow(n) # generate n distinct colors to plot the clusters
palette_tom = palette_tom[ToMATo_labels] # assign each point to the color
    where colors represent the clusters

plot(seeds[,2:3], pch = 16, cex = 0.8, col = palette_tom,
     xlab = "", ylab = "", xaxt="n", yaxt="n", main = "2d Projection Seeds
    Clustering (ToMATo)") # plot clusters (2d projection)

# compute the adjusted Rand index
adj.rand.index(ground_truth, ToMATo_labels) # results in 0.794092

# confusion matrix
confusion_mat_tom <- table(ground_truth, ToMATo_labels)
row.names(confusion_mat_tom) <- levels(seeds$variety)
colnames(confusion_mat_tom) <- c(1:max(na.omit(ToMATo_labels)))
confusion_mat_tom

#################################################################
############ k-means ############################################
#################################################################

seed_kmean <- kmeans(seeds, 3)

# compute the adjusted Rand index
adj.rand.index(ground_truth, seed_kmean$cluster) # results in 0.7974953
```