Asymtopic normality for tessellation-based Betti numbers

Bjarke Hautop Kristensen

Bachelor thesis in statistics

Supervisor: Christian Pascal Hirsch

13. June 2023

1 Abstract

Topological data analysis is a powerful tool for analyzing complex datasets and extracting meaningful topological properties. In this thesis, we explore the use of persistent Betti numbers, which capture a multiscale perspective of topological changes. We present approaches for computing persistent Betti numbers and establish two central limit theorems for persistent Betti numbers. The first theorem applies to general filtrations that satisfy certain conditions, with the Čech filtration being an example that satisfies these conditions. The second theorem is for the Voronoi tessellation-adapted filtration. These theorems provide a fundamental theoretical framework for statistical analysis and hypothesis testing in computational topology. Additionally, we conduct simulations to investigate the behavior of persistent Betti numbers under varying parameters, and propose a hypothesis test to determine if the number of points follows a fixed distribution or stems from a Poisson distribution. Our findings demonstrate the effectiveness of Voronoi tessellation in analyzing the topological structure of datasets.

Contents

2 Introduction

In recent years, topological data analysis has emerged as a powerful tool for analyzing complex datasets and extracting meaningful topological properties [2]. One such property is Betti numbers, which provide valuable insights into the topological changes across different levels of a dataset. Betti numbers intuitively describe the number of q-dimensional holes in a simplicial complex. Simplicial complexes are geometric objects built by joining points, edges, triangles, tetrahedra, tetrahedra and polytopes of higher dimensions. They provide a powerful tool for representing complex topological spaces in a way that is accessible for data analysis. By capturing the fundamental structural elements of these spaces, simplicial complexes offer a valuable framework for exploring and interpreting data in various fields, for example in materials science [5]. We will focus on persistent Betti numbers, where persistent Betti numbers also allow us to understand the time perspective of Betti numbers. Persistent Betti numbers provide information about how long the topological properties are present in the data set and at which levels.

We assume throughout this thesis that our data points come from a homogonous Poisson process, which is a basic stochastic process that is widely used. The Poisson process models the occurrence of events in continuous time or space, and has some important properties such as independence, memoryless and a constant rate. In our context, the homogeneous Poisson process acts as a natural model for the distribution of points in a dataset. These are some ideas that are unlikely to apply in the real world, but this can be used as a model under a null hypothesis.

First, in Section 4.1 we establish a central limit theorem for persistent Betti numbers with a filtration that satisfies certain conditions. The proof is given in Section 5.1 and it consists of showing that the persistent Betti number is weakly stabilizing and that it satisfies the Poisson bounded moment criterion [7].

Then, in Section 4.2, we present a central limit theorem for persistent Betti numbers with Voronoi tessellation-adapted filtering. Voronoi tessellation is a geometric partitioning technique that creates a cell for each point with the associated area that is closer to that point than to any other point. The proof is made in Section 5.2 and the proof is very similar

to the second proof, but some of the arguments need to be adapted to the new setup.

In section 6 we run simulations of the central limit theorem with Voronoi tessellation and investigate what happens when we change the parameters. We also set up two hypothesis tests. One has the null hypothesis of a fixed number of points and the alternative hypothesis that the number of points follows a Poisson distribution. This test has a simulated power of 0.25. The second hypothesis test has the null hypothesis that the number of points follows a Poisson distribution and the alternative hypothesis of a fixed number of points. This test has a simulated power of 0.01.

3 Notation and tools

In this section, we provide relevant definitions that are necessary to go deeper into topological data analysis and tessellations. In Section 3.1 we define, among other things, what a Poisson process is, what a simple complex is, what Betti numbers are and what filtering is. From these definitions we can now formulate Theorem 4.1, which is one of the main theorems we will show.

In Section 3.2 we define what Voronoi tessellation is, which allows us to formulate Theorem 4.2, which is the second main theorem we will show.

3.1 Spatial Statistics and Topological Data Analysis

Poisson process in $\mathbb{R}^N$

In the results shown in this bachelor project, we will use the assumption that our points come from a Poisson process. This is a relevant situation to consider because the assumption that our points come from a Poisson process in $\mathbb{R}^N$ means that the points are independent of each other and that the probability of finding a point in a given spatial domain is proportional to the size of that domain, which are assumptions that often make sense to make.

First, we need some introductory concepts, with definitions based on [3] and [4].

Definition 3.1 (**Locally finite measure**)**.** A Borel measure is a *locally finite measure*, if there for any point $p \in \mathbb{R}^N$ there exists an open neighbourhood N_p at p such that the

$\mu-$measure of N_p is finite.

Definition 3.2 (**Point process**)**.** Let N_{lf} denote the space of locally finite set of points $\varphi \subseteq \mathbb{R}^N$ and equip it with the smallest σ-algebra, $\mathcal{B}$, such that the maps

$$\varphi \mapsto \varphi(A) \coloneqq \#(\varphi \cap A)$$

are measureable for all Borel sets $A \subseteq \mathbb{R}^N$. The corresponding random variable $X = \{X_i\}_{i\geq 1}$ with values in N_{lf} is called a *point process*.

To define what a pointprocess is we will also need the following definitions

Definition 3.3 (**Intensity measure**)**.** For any point process $\mathcal{P}$ on $\mathbb{R}^N$ we define the *intensity measure* $\Lambda : \mathcal{B} :\to [0, \infty]$ as

$$\Lambda(\mathcal{B}) = \mathbb{E}[\mathcal{P}(B)], \quad B \in \mathcal{B}.$$

Definition 3.4 (**Non atomic measure**)**.** A *Non atomic measure* μ is a measure, where there for each measureable set A with $\mu(A) > 0$ exists a subset B of A such that

$$\mu(A) > \mu(B) > 0.$$

We are now ready to define what a poisson process is

Definition 3.5 (**Poisson process**)**.** Let Λ be a locally finite non atomic Borel measure on $\mathbb{R}^N$. A *poisson process* $\mathcal{P}$ with intensity measure Λ is a point process that satifies the following

1. $\mathcal{P}(A) \sim \text{Poi}(\Lambda(A))$ for any set $A \subseteq \mathbb{R}^N$,
2. The random variables $\mathcal{P}(A_1), \ldots, \mathcal{P}(A_n(x))$ are independent for all pairwise disjoint $A_1, \ldots, A_n(x) \subseteq \mathbb{R}^N$.

We often need to consider a poisson process only on a subset of $\mathbb{R}^N$, so the following definition is natural

Definition 3.6 (**Restricted poisson process**). For a poisson process $\mathcal{P}$ we define for any set $A \subseteq \mathbb{R}^N$ the *restricted poisson process* $\mathcal{P}_A$ as the poisson process defined on A.

A special class of poisson processes are homogeneous poisson processes, where the expected number of points in a set only depends on the volume of the set.

Definition 3.7 (**homogeneous poisson processes**). A poisson process, where the intensity measure $\Lambda = \nu\lambda$, where ν is the Lebesgue measure and λ is a constant, is called a *homogeneous poisson processes*.

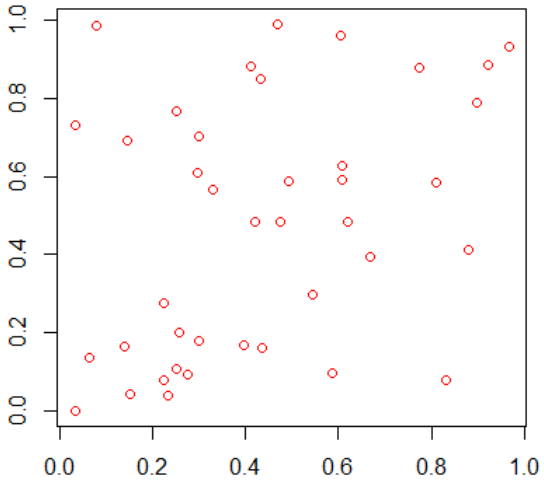

Figure 1: *Simulation of* 40 *points from a homogeneous poisson processes in* $[0,1] \times [0,1]$.

Simplicial complex

Simplicial complex is a geometric object that is made up of the union of points, edges, triangles, tetrahedra and higher dimensional polytopes. Formally, we use the following definition

Definition 3.8 (**Simplicial complex**). A *simplicial complex* is a collection of K non-empty subsets of a set K_0, such that $\{v\} \in K$ for all $v \in K_0$, and if $\tau \subset \sigma$ and $\sigma \in K$, then $\tau \in K$.

We call the elements of K_0 kalder vi *edges* of K, and we call the of elements K the *simplicies*. Furthermore we say that a simplex has *dimension* q or is a q*-simplex*, if it has cardinality $q+1$. We will use K_q to denote the collection of q-simplicies.

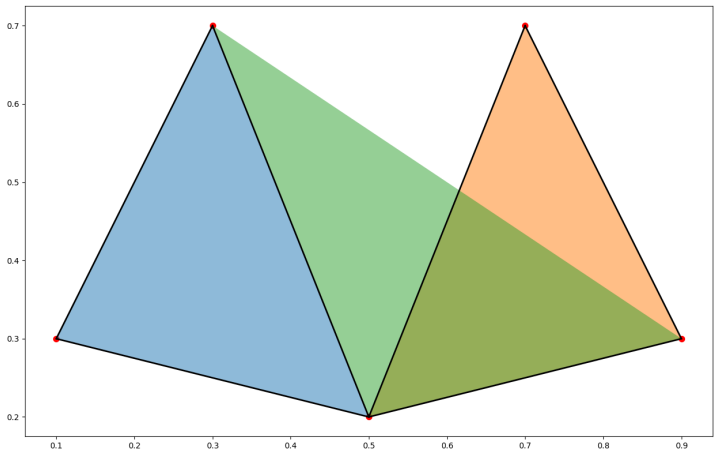

Figure 2: *An example of a simplical complex.*

An important example of a simplicial complex is the *Čech complex* which is constructed as follos: Given a finite set of points K_0 and a $\epsilon > 0$ we can construct the *Čech complex* $\check{\mathrm{C}}_\epsilon(K_0)$ by taking the elmenets of K_0 as edges in $\check{\mathrm{C}}_\epsilon(K_0)$ and for each $\sigma \subset K_0$ we let $\sigma \in \check{\mathrm{C}}_\epsilon(K_0)$, if the ϵ-ball with center in σ has a non-empty intersection, that is

$$\sigma = \{x_0, \ldots, x_q\} \in \check{\mathrm{C}}_\epsilon(K_0) \iff \bigcap_{i=0}^{q} \overline{B}_t(x_i) \neq \varnothing,$$

where $\overline{B}_t(x_i)$ denotes the ball with center x_i and radius t.

Homology

The idea behind homology is to understand the topological structure of your dataset in a way that is invariant under rotation, scaling and more. In particular, we will focus on Betti numbers and later persistent Betti numbers, which intuitively say something about the number of q-dimensional holes in a simplicial complex K.

We consider a q-simplex $\sigma = \{i_0, \ldots, i_q\}$ as a sorted set. For $q > 0$ we define an equivalence relation $i_{j_0}, \ldots, i_{l_0} \sim i_{l_q}, \ldots, i_{j_0}$ on two sorts of σ with even permutations. Let $\langle\sigma\rangle = \langle i_{j_0}, \ldots, i_{j_q}\rangle$ be the equivalence class where $i_{j_0} < \cdots < i_{j_q}$.

We consider the field $\mathbb{F}_2$ with basis given by the q-simplicies of K, since we do not need the orientation of our simplex. We can then construct a $\mathbb{F}_2$-vector space $C_q(K)$ as

$$C_q(K) = \mathrm{Span}_{\mathbb{F}_2}\{\langle\sigma\rangle \mid \sigma \in K_q\},$$

for $K_q \neq \varnothing$ and $C_q(K) = 0$ for $K_q = \varnothing$. Here $\mathrm{Span}_{\mathbb{F}_2}(A)$ for a set A denotes a vector space over $\mathbb{F}_2$, so the elements of A form a basis of the vector space. We define the boundary map $\partial_q : C_q(K) \to C_{q-1}(K)$ as

$$\partial_q(\langle i_0, \ldots, i_q\rangle) = \sum_{l=0}^{q} \langle i_0, \ldots, \widetilde{i_l}, \ldots, i_q\rangle, \tag{1}$$

where $\widetilde{i_l}$ means, that i_l is removed. We set $Z_q(K) = \ker \partial_q$ and $B_q(K) = \mathrm{image}\, \partial_{q+1}$. We can consider the boundary map as a matrix

$$M_q = (M_{\sigma,\tau})_{\sigma \in K_{q-1}, \tau \in K_q},$$

where the coefficients are given by (1).

For our definition of the Betti number to make sense, which we will give below, we need the following theorem

Theorem 3.9. *For any $q \geq 0$ we have*

$$B_q(K) \subseteq Z_q(K)$$

Proof. If we show that $\partial_q \circ \partial_{q+1}(\sigma) = 0$ for any $\sigma = \langle i_0, \ldots, i_{q+1}\rangle \in K_{q+1}$ then we are done. So if $q = 0$ then we have that $\partial_q \circ \partial_{q+1}(\sigma) = 0$ as per the definition. So suppose that $q > 0$.

Then we have

$$\begin{aligned}
\partial_q \partial_{q+1}(\sigma) &= \partial_q \sum_{l=0}^{q+1} \langle i_0, \ldots, \widetilde{i_l}, \ldots, i_{q+1} \rangle \\
&= \sum_{l=0}^{q+1} \partial_q \langle i_0, \ldots, \widetilde{i_l}, \ldots, i_{q+1} \rangle \\
&= \sum_{l=0}^{q+1} \left[\sum_{j=0}^{l-1} \langle i_0, \ldots, \widetilde{i_j}, \ldots, \widetilde{i_l}, \ldots, i_{q+1} \rangle + \sum_{j=l+1}^{q+1} \langle i_0, \ldots, \widetilde{i_l}, \ldots, \widetilde{i_j}, \ldots, i_{q+1} \rangle \right] \\
&= 0.
\end{aligned}$$

□

We can now define the Betti number.

Definition 3.10 (**Betti number**)**.** For $q \in \mathbb{N}_0$ we define the q'th homology of a simplicial complex K as the quotient vector space

$$H_q(K) \coloneqq \operatorname{kernel}(\partial_q) / \operatorname{image}(\partial_{q+1}).$$

We call the dimension

$$\beta_q(K) \coloneqq \dim H_q(K) = \dim \operatorname{kernel}(\partial_q) - \dim \operatorname{image}(\partial_{q+1})$$

the q'th *Betti number* of K.

$\beta_0(K)$ can geometrically be understood as the number of connected components and $\beta_1(K)$ can be understood as the number of 1-dimensional holes.

Example 3.11 (**Computation of Betti numbers**)**.** We will compute the Betti numbers for the following two simplicial complexes

$$\begin{aligned}
K &= \big\{\{a\}, \{b\}, \{c\}, \{a,b\}, \{a,c\}, \{b,c\}, \{a,b,c\}\big\} \\
K' &= \big\{\{d\}, \{e\}, \{f\}, \{d,e\}, \{d,f\}, \{e,f\}\big\}
\end{aligned}$$

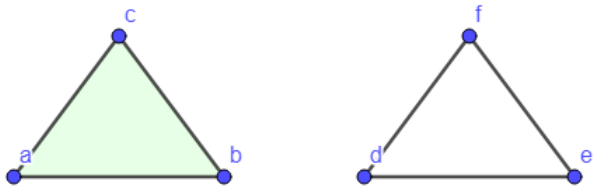

Figure 3: *Geometric realization of K and K' respectively. The triangle on the left is colored because we have $\{a, b, c\}$ in our simplicial complex, while the triangle on the right is not colored because $\{d, e, f\}$ is not in our simplicial complex.*

We start by computing the Betti number for K. We have the following sequence of vector spaces and linear maps:

$$0 \longrightarrow \mathbb{F}_2 \xrightarrow{\partial_2} \mathbb{F}_2^3 \xrightarrow{\partial_1} \mathbb{F}_2^3 \xrightarrow{\partial_0} 0.$$

If we sort the bases of the vector space lexicographically, we have that the matrix representation of ∂_2 and ∂_1 is given by

$$M_2 = \begin{pmatrix} 1 & 1 & 1 \end{pmatrix}^T$$
$$M_1 = \begin{pmatrix} 1 & 1 & 0 \\ 1 & 0 & 1 \\ 0 & 1 & 1 \end{pmatrix} \sim \begin{pmatrix} 1 & 0 & 1 \\ 0 & 1 & 1 \\ 0 & 0 & 0 \end{pmatrix},$$

and the kernel of ∂_0 is $\big\{\{a\}, \{b\}, \{c\}\big\}$. We notice the row space of ∂_1 is

$$\left\{ \begin{bmatrix} 1 & 0 & 0 \end{bmatrix}^T, \begin{bmatrix} 0 & 1 & 0 \end{bmatrix}^T \right\}.$$

Hence we have

$$\beta_0(K) = \dim \ker(\partial_0) - \dim \operatorname{image}(\partial_1) = 3 - 2 = 1.$$

We have that the dimension of the image of M_2 is 1, and that the kernel of M_1 is the zero vector, so we have that

$$\beta_1(K) = \dim \ker(\partial_1) - \dim \operatorname{image}(\partial_2) = 1 - 1 = 0,$$

and that all higher Betti numbers are 0.

We will now compute the Betti numbers of K'. Here we have the following sequence of vector spaces and linear maps

$$0 \longrightarrow \mathbb{F}_2^3 \xrightarrow{\partial_1} \mathbb{F}_2^3 \xrightarrow{\partial_0} 0.$$

The only difference in the computation of Betti numbers of K' is that $\dim \text{image}(\partial_2) = 0$ here, so we have

$$\beta_0(K) = \dim \ker(\partial_0) - \dim \text{image}(\partial_1) = 3 - 2 = 1$$
$$\beta_1(K) = \dim \ker(\partial_1) - \dim \text{image}(\partial_2) = 1 - 0 = 1,$$

and that all higher Betti numbers are 0.

Filtration

A filtration is a growing sequence of simplicial complexes that gives us a tool to examine the structure of our data.

Let $\mathcal{F}(\mathbb{R}^N)$ denote all finite (non-empty) subsets of $\mathbb{R}^N$. For a function f on $\mathcal{F}(\mathbb{R}^N)$, there exists a permutation invariant function $\widetilde{f}_k$ on $(\mathbb{R}^N)^k$ for any $k \geq 1$ such that $f(\{x_1, \ldots, x_k\}) = \widetilde{f}_k(x_1, \ldots, x_k)$. We say that if f is measurable, then $\widetilde{f}_k$ is measurable on $(\mathbb{R}^N)^k$ for any $k \geq 1$.

Let $\kappa : \mathcal{F}(\mathbb{R}^N) \to [0, \infty]$ be a measurable function that satisfies the following:

(K1) $0 \leq \kappa(\sigma) \leq \kappa(\tau)$, if σ is a subset of τ,

(K2) κ is translation invariant, that is $\kappa(\sigma + x) = \sigma$ for all $x \in \mathbb{R}^N$, where $\sigma + x \coloneqq \{y + x : y \in \sigma\}$,

(K3) There is an increasing function $\rho : [0, \infty] \to [0, \infty]$ with $\rho(t) < \infty$ for $t < \infty$ such that

$$\|x - y\| \leq \rho(\kappa(\{x, y\})),$$

where $\|.\|$ is the Euclidean norm of $\mathbb{R}^N$.

An important example of κ is

$$\kappa_C(\{x_0, \ldots, x_q\}) = \inf_{w \in \mathbb{R}^N} \max_{0 \leq i \leq q} \|x_i - w\|$$

which defines the Čech filtration $\mathbb{C}(\varphi) = \{C(\varphi, t)\}_{t\geq 0}$. It induces the Čech complexet, as we defined in ealier.

Given such a function κ we construct a filtration $\mathbb{K}(\Xi) = \{K(\Xi, t) : 0 \leq t < \infty\}$ of simplicial complexes from a finite point configuration $\Xi \in \mathcal{F}(\mathbb{R}^N)$ with

$$K(\Xi, t) = \{\sigma \subset \Xi : \kappa(\sigma) \leq t\},$$

that is $\kappa(\sigma)$ is the birthtime of a simplex σ in the filtration $\mathbb{K}(\Xi)$.

Definition 3.12 (Persistent Betti number). For a filtration $\mathbb{K}$ the (r, s)'th *persistent Betti number* is defined as

$$\beta_q^{r,s}(\mathbb{K}) = \dim \frac{Z_q(K_r)}{Z_q(K_r) \cap B_q(K_s)}, \quad (r \leq s),$$

where $Z_q(K_r)$ and $B_q(K_r)$ are respectively the q'th cycle group and q'th boundary group.

The persistent Betti number can tell us the number of q-dimensional holes in $\mathbb{K}$ which exists between r and s.

For a filtration $\mathbb{K}(\Xi)$ we denote the q'th persistent diagram with

$$D_q(\Xi) = \{(b_i, d_i) \in \Delta : i = 1, \ldots, n_q\},$$

where $\Delta = \{(x, y) \in \overline{\mathbb{R}}^2 : 0 \leq x < y \leq \infty\}$ is determined by the unique decompoisiton of the persistent homology. This means the points $\{(b_i, d_i) \in \Delta\}$ with $\beta_q^{r,s}(\mathbb{K}) = \#\{i : b_i \leq r, d_i > s\}$. So b_i denotes the birth and d_i when it disappears again. We set $d_i = \infty$ if the i'th homology class survives forever. So we can understand the (r, s)-persistent Betti number $\beta_q^{r,s}(\mathbb{K})$ as counting the number of birth-death pairs in a persistence diagram $D_q(K)$ with birth before r and death after s.

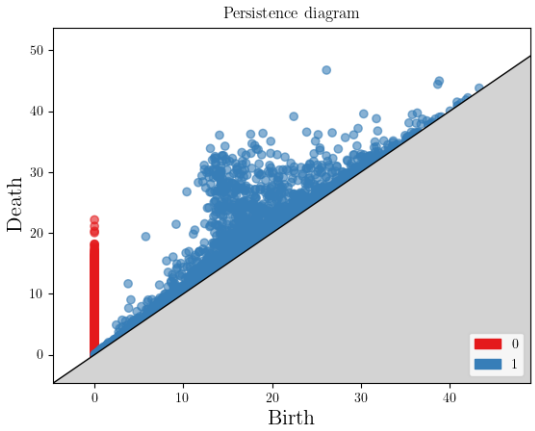

Figure 4: *An example of a persistence diagram.*

3.2 Tessellations

Tessellations divide a space of points into different cells based on some criteria. A widely used tessellation is the Voronoi tessellation, which creates a cell for each point with the associated area that is closer to that point than to any other point. This creates a partition of space into cells, which are called Voronoi cells. An example of a Voronoi tessellation can be seen in Figure 5. We use the following definition

Definition 3.13 (**Voronoi tessellation**)**.** Let φ be a locally finite subset of $\mathbb{R}^N$. We define the Voronoi cell $C(x, \varphi)$ for a $x \in \varphi$ as

$$C(x, \varphi) = \{y \in \mathbb{R}^N : \|y - x\| \leq \|y - x'\| \quad \text{for all } x' \text{ in } \varphi.\}$$

The *Voronoi tessellation* of φ is the set $V(\varphi) = \{C(x, \varphi) : x \in \varphi\}$.

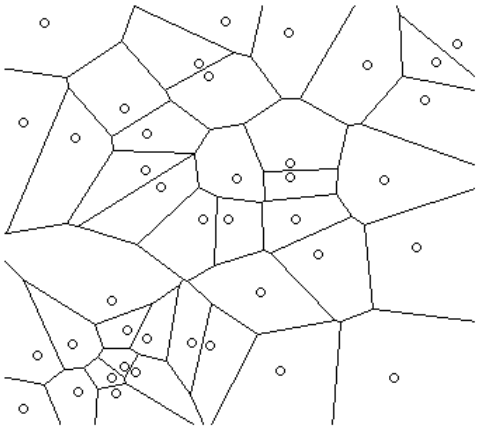

Figure 5: *Points from Figure 1 with Voronoi tessellation added.*

To compute persistent Betti numbers, we will use a tessellation-adapted filtration where entire simplices are added at filtering times given by their circumscribed circle radii. In other words, a q-simplex $f \in \Xi_n^q$, defined by the points $P_0, \ldots, P_m$, belongs to the filtering at level $t > 0$ if and only if its circumscribed circle radius, $r_C(f)$, satisfies $r_C(f) \leq t$, where $r_C(f)$ is defined as

$$r_C(f) \coloneqq \min_{y \in \mathbb{R}^N} \max_{i \leq m} |y - P_i|.$$

We will refer to this as the tessellation-customized filtering associated with the surfaces in the tessellation. Let $\mathbb{K}_{\text{Vor}}$ denote the Voronoi tessellation-customized filtering. Note that $\mathbb{K}_{\text{Vor}}$ does not satisfy (K3).

This is very similar to Čech filtering, the difference being when simplicies are added to the simplical complex. In Čech filtering, simplicies are added based on pairwise distance between points.

4 Central limit theorems for persistent Betti numbers

In this section we will formulate a central limit theorem for persistent Betti numbers with a filtration satisfying (K1-K3). We will also formulate a central limit theorem persistent Betti numbers with a tessellation-adapted Voronoi filtration. Using the existence of these central limit theorems, one can test whether one's points come from a homogeneous Poisson process with unit intensity as we will do in Section 6.

We state here the necessary properties of functionals to obtain a version of the central limit theorem. Let $\{W_n\}$ be Borel subsets of $\mathbb{R}^N$ satisfying the following conditions:

(A1) $|W_n| = n$ for all $n \in \mathbb{N}$.

(A2) $\bigcup_{n\geq 1} \bigcap_{m\geq n} W_m = \mathbb{R}^N$

(A3) $\lim_{n\to\infty} \|(\partial W_n)^{(r)}\|/n = 0$ for all $r > 0$ where $A^{(r)}$ is the set of points which are at most r away from A.

(A4) There exists a constant $\gamma > 0$ such that $\sup_{x,y\in W_n} |x - y| \leq \gamma n^{\gamma}$.

Given such a sequence, let $\mathcal{W} = \mathcal{W}(\{W_n\})$ be the collection of all subsets A in $\mathbb{R}^N$ of the form $A = W_n + x$ for a W_n in the sequence and a point $x \in \mathbb{R}^N$.

Let H be a functional that takes real values in $\mathcal{F}(\mathbb{R}^N)$. We say that H is *translation invariant* if $H(\mathcal{X} + y) = H(\mathcal{X})$ for all $\mathcal{X} \in \mathcal{F}(\mathbb{R}^N)$ and $y \in \mathbb{R}^N$. Let D_0 denote the increment in H obtained by adding $\{0\}$

$$D_0 H(\mathcal{X}) = H(\mathcal{X} \cup \{0\}) - H(\mathcal{X}), \quad \mathcal{X} \in \mathcal{F}(\mathbb{R}^N).$$

We say the function H is *weakly stabilized* on W if there exists a random variable $D(\infty)$, so $D_0 H(\mathcal{P}_{A_n(x)}) \overset{n.s.}{\to} D(\infty)$ for $n \to \infty$ for any sequence $\{A_n(x) \in \mathcal{W}\}_{n\geq 1}$ going towards $\mathbb{R}^N$. The Poisson limited moment criterion on $\mathcal{W}$ is given by

$$\sup_{0\in A\in\mathcal{W}} \mathbb{E}[(D_0 H(\mathcal{P}_A))^4] < \infty.$$

We will now formulate the two central limit theorems mentioned above.

4.1 Poisson process with filtration

We will prove the following central limit theorem for a poisson process with a filtration satisfying (K1-K3).

Theorem 4.1. *Let $\mathcal{P}$ be a homogeneous Poisson point process with unit intensity. Suppose that the sequence $\{W_n\}$ satisfies conditions (A1-A4). Then for any $0 \leq r \leq s < \infty$ we have*

$$\frac{\beta_q^{r,s}(\mathbb{K}(\mathcal{P}_{W_n})) - \mathbb{E}[\beta_q^{r,s}(\mathbb{K}(\mathcal{P}_{W_n}))]}{n^{1/2}} \xrightarrow{d} N(0, \sigma_{r,s}^2) \quad \textit{for } n \to \infty.$$

It can be shown, that $\sigma_{r,s}^2 > 0$ and Proposition 9.1 in [1] contains some ideas as to how one can show this.

4.2 Poisson process with tessellation-adapted Voronoi filtration

We will show the following central limit theorem for a Poisson process with a Voronoi tessellation. In the context of Voronoi tessellation, we will always consider the window $W_n = [-\frac{n}{2}, \frac{n}{2}]^N$. We cannot use Theorem 4.1, as Voronoi tessellation-adapted filtration does not satisfy (K3).

Theorem 4.2. *Let $\mathcal{P}$ be a homogeneous Poisson point process with unit intensity. Then for any $0 \leq r \leq s < \infty$ we have*

$$\frac{\beta_q^{r,s}(\mathbb{K}_{Vor}(\mathcal{P}_{W_n})) - \mathbb{E}[\beta_q^{r,s}(\mathbb{K}_{Vor}(\mathcal{P}_{W_n}))]}{n^{1/2}} \xrightarrow{d} N(0, \sigma_{r,s}^2) \quad \textit{for } n \to \infty.$$

5 Proofs

5.1 Poisson process with filtration

The proofs in this section are mainly based on [3].

Lemma 5.1. *Let H be a functional that takes real values defined on $\mathcal{F}(\mathbb{R}^N)$. Assume that H is translation invariant and weakly stabilizing on $\mathcal{W}$ and satisfies the Poisson bounded moment condition. Then there exists a constant $\sigma^2 \in [0, \infty)$ such that $n^{-1}\,\mathbb{V}[H(\mathcal{P}_{W_n})] \to \sigma^2$ and*

$$\frac{H(\mathcal{P}_{W_n}) - \mathbb{E}[H(\mathcal{P}_{W_n})]}{n^{1/2}} \xrightarrow{d} N(0, \sigma^2) \quad \textit{for } n \to \infty.$$

Proof. See Lemma 3.1 in [7].

Lemma 5.2. *Let $\mathbb{K} = \{K_t\}_{t\geq 0}$ and $\widetilde{\mathbb{K}} = \{\widetilde{K}_t\}_{t\geq 0}$ be filtrations on $K_t \subset \widetilde{K}_t$ for $t \geq 0$. Then*

$$|\beta_q^{r,s}(\widetilde{\mathbb{K}}) - \beta_q^{r,s}(\mathbb{K})| \leq \sum_{j=q,q+1} \big(\big|\widetilde{K}_{s,j}\backslash K_{s,j}\big| + \big|\{\sigma \in K_{s,j}\backslash K_{r,j} : \widetilde{t}_\sigma \leq r\}\big|\big),$$

where $\widetilde{K}_{t,j}$ (or $K_{t,j}$) is a set of j-simplicies in $\widetilde{K}_t$ (or K_t) and $\widetilde{t}_\sigma$ (or t_σ) is the birth-time of σ a filtration $\widetilde{\mathbb{K}}$ (or $\mathbb{K}$).

Proof. See Lemma 2.11 in [3].

Korollar 5.3. *The functional $\beta_q^{r,s}(\mathbb{K})$ is weakly stabilizing.*

Proof of Theorem 4.1. For a fixed $r \leq s$, we consider the persistent Betti number $\beta_q^{r,s}(\mathbb{K}(\cdot))$ as a functional on $\mathcal{F}(\mathbb{R}^N)$ and check the 3 conditions given in Lemma 5.1. We have that it is translation invariant, since κ is translation invariant from (K2). Korollar 5.3 gives us that it is weakly stabilizing, so all we need is to show the Poisson bounded moment criterion on $\mathcal{W}$. Let $F_q(\Phi, r)$ be the number of q-simplices in $K(\Phi, r)$. We then have the following

$$\begin{aligned}
|D_0\beta_q^{r,s}(\mathbb{K}(\mathcal{P}_A))| &= |\beta_q^{r,s}(\mathbb{K}(\mathcal{P}_A \cup \{0\})) - \beta_q^{r,s}(\mathbb{K}(\mathcal{P}_A))| \\
&\leq \sum_{j=q,q+1} |K_j(\mathcal{P}_A \cup \{0\}, s)\backslash K_j(\mathcal{P}_A, s)| \\
&\leq \sum_{j=q,q+1} F_j(\mathcal{P}_{\bar{B}_{\rho(s)}(0)}, s) \\
&\leq \sum_{j=q+1,q+2} \mathcal{P}\big(\overline{B}_{\rho(s)}(0)\big)^j
\end{aligned}$$

Where in the first inequality we used Lemma 5.2, in the second inequality we used the condition (K3), which gives us that the distance is bounded by the sphere with radius $\rho(s)$. The last inequality comes from the fact that we can bound the number of q-simplicies by the number of points to the power of $q+1$. We now have

$$\begin{aligned}
\mathbb{E}\left[|D_0\beta_q^{r,s}(\mathbb{K}(\mathcal{P}_A))|^4\right] &\leq \mathbb{E}\left[\left(\mathcal{P}\big(\bar{B}_{\rho(s)}(0)\big)^{q+1} + \mathcal{P}\big(\bar{B}_{\rho(s)}(0)\big)|^{q+2}\right)^4\right] \\
&\leq \mathbb{E}\left[\left(2 \cdot \max\left\{|\mathcal{P}_{\bar{B}_{\rho(s)}(0)})|^{q+1}, |\mathcal{P}_{\bar{B}_{\rho(s)}(0)})|^{q+2}\right\}\right)^4\right] \\
&\leq \mathbb{E}\left[2^4 \cdot \max\left\{|\mathcal{P}_{\bar{B}_{\rho(s)}(0)})|^{4q+4}, |\mathcal{P}_{\bar{B}_{\rho(s)}(0)})|^{4q+8}\right\}\right] \\
&\leq 2^4\,\mathbb{E}\left[|\mathcal{P}_{\bar{B}_{\rho(s)}(0)})|^{4q+4}\right] + 2^4\,\mathbb{E}\left[|\mathcal{P}_{\bar{B}_{\rho(s)}(0)})|^{4q+8}\right].
\end{aligned}$$

Since $\mathcal{P}\big(\bar{B}_{\rho(s)}(0)\big)$ is poisson distributed, we have that all moments are bounded and thus we have that the above expression is finite. Since the final expression does not depend on A, we have that

$$\sup_{0\in A\in\mathcal{W}} \mathbb{E}[D_0\beta_q^{r,s}(\mathbb{K}(\mathcal{P}_A))^4] < \infty,$$

as desired. □

Lemma 5.4. *Let $P \subseteq \mathbb{R}^N$ be a locally finite set. For any fixed $r \leq s$ there exists D_∞ and $R > 0$ such that*

$$D_0\beta_q^{r,s}(\mathbb{K}(P_{\bar{B}_a(0)})) = D_\infty$$

for all $a \geq R$.

To prove Lemma 5.4 we will need the two following Lemmas.

Lemma 5.5. *Let A, B, U, V be subspaces of a vector space where $A \subset U$ and $B \subset V$. Then we have*

$$\dim \frac{U \cap V}{A \cap B} \leq \dim \frac{U}{A} + \dim \frac{V}{B}.$$

Proof. Since we have the formula $\dim(U\cap V)+\dim(U\cup V) = \dim U+\dim V$ and $\dim(U/A) = \dim U - \dim A$, we have that

$$\dim \frac{U \cap V}{A \cap B} = \dim \frac{U}{A} + \dim \frac{U}{B} + (\dim U \cup V - \dim A \cup B) \leq \dim \frac{U}{A} + \dim \frac{V}{B},$$

as desired. □

Lemma 5.6. *Let $D = \begin{bmatrix} A & B \end{bmatrix}$ be a matrix with submatrices A and B. Let l be the number of collons in B. Then we have*

$$\dim \ker D \leq \dim \ker A + l$$

Proof. Lad $B = \begin{bmatrix} b_1 & \dots & b_l \end{bmatrix}$, where b_i is the i'th coloumn vector of B and set $D^{(i)} = \begin{bmatrix} A\, b_1 & \dots & b_i \end{bmatrix}$. Then for any i we have

$$\dim \ker D^{(i)} \leq \dim \ker D^{(i-1)} + 1.$$

By applying this iteratively, we get the desired result. □

We will use that we almost certainly have that a homogeneous Poisson process $\mathcal{P}$ has infinite points in $\mathbb{R}^N$ that do not have accumulation points.

Proof of Lemma 5.4 Set $P' = P \cup \{0\}$. Let $K_{r,a} = K(P_{\bar{B}_a(0)}, r)$ be the simplicial complex defined on $P_{\bar{B}_a(0)}$ with parameter r and correspondingly set $K'_{r,a} = K(P'_{\bar{B}_a(0)}, r)$. Using Definition 3.12 we can write $D_0\beta_q^{r,s}(\mathbb{K}(P_{\bar{B}_a(0)}))$ as

$$\begin{aligned} D_0\beta_q^{r,s}(\mathbb{K}(P_{\bar{B}_a(0)})) &= \dim \frac{Z_q(K'_{r,a})}{Z_q(K'_{r,a}) \cap B_q(K'_{s,a})} - \dim \frac{Z_q(K_{r,a})}{Z_q(K_{r,a}) \cap B_q(K_{s,a})} \\ &= (\dim Z_q(K'_{r,a}) - \dim Z_q(K_{r,a})) \\ &\quad - \dim\big(Z_q(K'_{r,a}) \cap B_q(K'_{s,a})\big) + \dim\big(Z_q(K_{r,a}) \cap B_q(K_{s,a})\big) \end{aligned}$$

So we can consider the stability with respect to a for $\dim Z_q(K_{r,a})$ and $\dim Z_q(K_{r,a}) \cap B_q(K_{s,a})$. So let's start with $\dim Z_q(K_{r,a})$. Since dim only takes non-negative values, it is enough to show that it is bounded and non-decreasing for the limit to exist. We have that $K_{r,a} \subseteq K'_{r,a}$ and therefore we have that $Z_q(K_{r,a}) \subseteq Z_q(K'_{r,a})$. We now write $K'_{r,a}$ as a disjoint union $K'_{r,a} = K_{r,a} \sqcup K^0_{r,a}$, where $K^0_{r,a}$ is the set of simplices, that have point 0 and let $K^0_{r,a,q}$ be the set of simplices of order q, so that $K^0_{r,a,q} = \{\sigma \in (K'_{r,a})_q : 0 \in \sigma\}$.

Let $\partial_{q,a}$ and $\partial'_{q,a}$ be the qth boundary mapping on $K_{r,a}$ and $K'_{r,a}$ respectively. Then we get the following block matrix

$$\partial'_{q,a} = \begin{bmatrix} M_{1,\rho} & \mathbf{0} \\ M_{2,\rho} & \partial_{q,a} \end{bmatrix},$$

where the first columns and columns are ordered by simplices in $K^0_{r,a,q}$ and $K_{r,a,q-1}$ and the second columns and columns correspond to simplices in $K_{r,a}$.

For any simplex $\sigma \in K(P, r)$ containing the point 0, we have that it is included in $\overline{B}_{\rho(r)}(0)$. Therefore, the set $K^0_{r,a,q}$ is independent of a for $a \geq \rho(r)$, which we denote by $K^0_{r,*,q}$. Now we can use Lemma 5.6 on the matrix form given above, where we set $D = \partial'_{q,a}$, $A = \begin{bmatrix} \mathbf{0} \\ \partial_{q,a} \end{bmatrix}$ and $B = \begin{bmatrix} M_{1,\rho} \\ M_{2,\rho} \end{bmatrix}$. Then we have

$$\dim Z_q(K'_{r,a}) - \dim Z_q(K_{r,a}) \leq |K^0_{r,a,q}| = |K^0_{r,*,q}|,$$

for $a \geq \rho(r)$ and it is thus bounded for any fixed r.

We will now show that it is non-decreasing. So we define a homomorphism given by

$$f : \frac{Z_q(K'_{r,a_1})}{Z_q(K_{r,a_1})} \ni [c] \mapsto [c] \in \frac{Z_q(K'_{r,a_2})}{Z_q(K_{r,a_2})}$$

for $a_1 \leq a_2$. This mapping is well-defined, since $Z_q(K_{r,a_1}) \subset Z_q(K_{r,a_2})$ and correspondingly that $Z_q(K'_{r,a_1}) \subset Z_q(K'_{r,a_2})$. We have that the kernel of f is trivial, since if $f[c] = 0$ then $c \in Z_q(K_{r,a_2})$ and since $a_1 \leq a_2$ we also have $c \in Z_q(K_{r,a_1})$. Since the kernel is trivial, f is injective. Thus we have the inequality

$$\dim Z_q(K'_{r,a_1})/Z_q(K_{r,a_1}) \leq \dim Z_q(K'_{r,a_2}/Z_q(K_{r,a_2}),$$

which shows that it is non-decreasing. So we have now shown the stability of $\dim Z_q(K_{r,a})$.

We will now show the stability of $\dim(Z_q(K_{r,a}) \cap B_q(K_{s,a}))$. We will use the same proof technique as above. From Lemma 5.5 we have that

$$\dim \frac{Z_q(K'_{r,a}) \cap B_q(K'_{s,a})}{Z_q(K_{r,a}) \cap B_q(K_{r,a})} \leq \dim \frac{Z_q(K'_{r,a})}{Z_q(K_{r,a})} + \dim \frac{B_q(K'_{r,a})}{B_q(K_{r,a})}.$$

As before, we again have that $|K^0_{s,a,q+1}| = |K^0_{s,*,q+1}|$ for all a large enough. Thus, we have that

$$\dim Z_q(K'_{r,a}) \cap B_q(K'_{s,a}) - \dim Z_q(K_{r,a}) \cap B_q(K_{s,a}) \leq |K^0_{r,*,q}| + |K^0_{r,*,q+1}|,$$

and thus we have shown it is bounded. Similarly as before, we can show that the kernel is trivial and thus injective in the following homomorphism

$$f : \frac{Z_q(K'_{r,a_1}) \cap B_q(K'_{s,a_1})}{Z_q(K_{r,a_1}) \cap B_q(K_{s,a_1})} \ni [c] \mapsto [c] \in \frac{Z_q(K'_{r,a_2}) \cap B_q(K'_{s,a_2})}{Z_q(K_{r,a_2}) \cap B_q(K_{s,a_2})},$$

for $a_1 \leq a_2$. This implies that it is not decreasing as desired. This concludes the proof of the Lemma. □

Proof of Korollar 5.3 Let $R > 0$ be chosen such that it satisfies Lemma 5.4 and let $\{A_n \in \mathcal{W}\}_{n \geq 1}$ be a sequence going towards $\mathbb{R}^N$. Then if there exists an $n_0 \in \mathbb{N}$ then $B_R(0) \subset A_n$ for all $n \geq n_0$.

For $n \geq n_0$ we define $L_{r,n} = K(\mathcal{P}_{A_n}, r)$ to be the simplicial complex defined on $\mathcal{P}_{A_n}$ with parameter r. Since A_n is bounded there exists $a > R$ so

$$B_R(0) \subset A_n \subset B_a(0).$$

Using the same approach as in Lemma 5.4, where we showed injectivity, we can show that

$$\frac{Z_q(K'_{r,R})}{Z_q(K_{r,R})} \subseteq \frac{Z_q(L'_{r,n})}{Z_q(L_{r,n})} \subseteq \frac{Z_q(K'_{r,a})}{Z_q(K_{r,a})},$$

where $K_{r,a} = K(\mathcal{P}_{\bar{B}_a(0)}, r)$ as before. From our choice of R, we have that the dimension of $Z_q(K'_{r,R})/Z_q(K_{r,R})$ and $Z_q(K'_{r,a})/Z_q(K_{r,a})$ are equal for $n \geq n_0$. Thus, we have that

$$\dim Z_q(K'_{r,R} - \dim Z_q(K_{r,R}) = \dim Z_q(L'_{r,n}) - \dim Z_q(L_{r,n}).$$

Similarly, we can show that $\dim Z_q(L'_{r,n}) \cap B_q(L'_{r,n}) - Z_q(L_{r,n}) \cap B_q(L_{r,n})$ is invariant for $n \geq n_0$. We have shown that it becomes constant and thus we have shown that $\beta_q^{r,s}(\mathbb{K})$ is weakly stabilizing, as desired. □

5.2 Poisson process with tessellation-adapted Voronoi filtration

There are only a few differences in the proof of Theorem 4.2 compared to Theorem 4.1. The differences are

(1) Some of the arguments in Lemma 5.4 to be used to show that $B_q^{r,s}(\mathbb{K}_{\text{Vor}})$ is weakly stabilizing, analogous to the proof of Corollary 5.3.

(2) Show that $B_q^{r,s}(\mathbb{K}_{\text{Vor}}(\cdot))$ satisfies the Poisson bounded moment criterion on $\mathcal{W}$, as analogous to the proof of Theorem 4.1.

We will start with some introductory remarks. Set $C_{A,0}$ to be the Voronoi cell for 0 in $\mathcal{P}_A \cup \{0\}$. When the cell $C_{A,0}$ is added to the cells for $\mathcal{P}_A$, the cells closest to it will change. So set $R_{A,0}$ to be the smallest positive integer so that all finite Voronoi cells that are changed by the addition of $C_{A,0}$ are contained in $B_{R_{A,0}}(0)$. The idea behind the proof is that we can ensure that $R_{A,0}$ does not get too big by making a box around $\{0\}$ and dividing the box into a grid, and showing that only points inside this grid are affected by the addition of $\{0\}$. Furthermore, we set $\Xi_{A,0}^{(0)}$ to be Voronoi vertices for $\mathcal{P}_A \cup \{0\}$ and $\Xi_A^{(0)}$ to be Voronoi vertices for $\mathcal{P}_A$.

Proof of (1). In the proof of Lemma 5.4, some of the arguments need to be changed. To restrict $\dim Z_q(K'_{r,a}) - \dim Z_q(K_{r,a})$ we can do the following. Let $K_{r,a,q}^U$ denote the set of

simplices outside the stability radius $R_{A,0}$ and $K^I_{r,a,q}$ denote the set of simplices inside the stability radius $R_{A,0}$. We note that $K^U_{r,a,q}$ is independent of a for $a > R_{A,0}$, which we denote by $K^U_{r,*,q}$. Using Lemma 5.6 twice, we have that

$$\begin{aligned}
\dim Z_q(K'_{r,a}) - \dim Z_q(K_{r,a}) &= \dim Z_q(K'_{r,a}) - \dim Z_q(K_{r,a}) - |K^U_{r,a,q}| + |K^U_{r,a,q}| \\
&\leq \left|\dim Z_q(K_{r,a}) - |K^U_{r,a,q}|\right| + \left|\dim Z_q(K'_{r,a}) - |K^U_{r,a,q}|\right| \\
&\leq |K^U_{r,a,q}| + |K^I_{r,a,q}| \\
&= |K^U_{r,*,q}| + |K^I_{r,a,q}|,
\end{aligned}$$

and since P is a locally finite set we have $K^I_{r,a,q}$ is finite and hence bounded for any fixed r.

Proof of (2). We will show it in the case where A is a cube with center x, that is A is on the form

$$A_n(x) = [-n/2, n/2]^N + x$$

As in Theorem 4.1 we need to show that

$$\sup_{0 \in A \in \mathcal{W}} \mathbb{E}\Big[\left|K_j(\mathcal{P}_A \cup \{0\}, s) \backslash K_j(\mathcal{P}_A, s)\right|^4\Big]$$

for $j = q, q+1$ is bounded.

Define

$$E_{r,n,x} \coloneqq \Big\{A_{\sqrt{r}}(\sqrt{r}z) \cap \mathcal{P}_A \neq \varnothing \text{ for all } \sqrt{r}z \in \sqrt{r}\mathbb{Z}^N \cap A_n(x)\Big\}$$

to denote the outcome that for any point $\sqrt{r}z \in \sqrt{r}\mathbb{Z}^N \cap A_n(x)$, the $\sqrt{r}$-case $A_{\sqrt{r}}(\sqrt{r}z)$ contains at least one point from $\mathcal{P}_A$. To show the moment condition, we need the following Lemma.

Lemma 5.7. *($E_{r,n,x}$ happens with high probability). There exists a $c_1 > 0$ such that for all $r > 1$we have*

$$\sup_{n \geq 0} \sup_{x \in \mathbb{R}^N} \mathbb{P}(E^c_{r,n,x}) \leq \exp(-c_1 r^{N/2})$$

Proof. We can write $E_{r,n,x}$ as

$$E_{r,n,x} = \bigcap_{\sqrt{z} \in \sqrt{r}\mathbb{Z}^N \cap A_n(x)} \{A_{\sqrt{r}}(\sqrt{r}z) \cap \mathcal{P} \neq \emptyset\}.$$

We can now notice that $|\sqrt{r}\mathbb{Z}^N \cap A_n(x)| \leq (\sqrt{r}+1)^N$ and hence we have

$$\mathbb{P}(E^c_{r,n,x}) \leq |\sqrt{r}\mathbb{Z}^N \cap A_n(x)|\, \mathbb{P}(A_{\sqrt{r}}(\sqrt{r}z) \cap \mathcal{P} = \emptyset) \leq C(r+1)^N \exp(-r^{N/2}),$$

where we bounded $\mathbb{P}(A_{\sqrt{r}}(\sqrt{r}z) \cap \mathcal{P} = \emptyset)$ by using $\mathcal{P}$ is a Poisson process. We can now notice that the right hand side is $\exp\big(-r^{N/2}(1+o(1))\big)$ and hence there exists a constant $c_1 > 0$ such that

$$\sup_{n\geq 0} \sup_{x \in Q_n} \mathbb{P}(E^c_{r,n,x}) \leq \exp(-c_1 r^{N/2}),$$

as desired. □

Lemma 5.8. *For any $k \geq N^2 + 9$ we have that*

$$E_{k/(4N),n,x} \subseteq \{R_{A,0} \leq k\},$$

Proof. We will start by showing for any $k \geq N^2 + 9$ that

(a) Under the event $E_{k/(4N),n,x}$, any cell with center in $A_{k/(4N)-(2N+2)\sqrt{k/(4N)}}(x)$ is contained in $A_{k/(4N)-2\sqrt{k/(4N)}}(x)$,

(b) Under the event $E_{k/(4N),n,x}(x)$, the intersection set between any cell with center outside $A_{k/(4N)+(N-1)\sqrt{k/(4N)}}(x)$ and with $A_{k/(4N)-2\sqrt{k/(4N)}}(x)$ is empty.

Proof of (a). Let C be a cell with center at $P \in A_{k/(4N)-(2N+2)\sqrt{k/(4N)}}(x)$ and consider an arbitrary point P' outside $A_{k/(4N)-2\sqrt{k/(4N)}}(x)$. Let P'' be the closest point on the boundary of $A_{k/(4N)-2\sqrt{k/(4N)}}(x)$. Then we have that $|P'-P| \geq |P'-P''| + \frac{(2N+2)\sqrt{k/(4N)}-2\sqrt{k/(4N)}}{2} = |P'-P''| + N\sqrt{k/(4N)}$. Under the event $E_{k/(4N),n,x}$, the distance from P' to one of the centers is at most $|P'-P''| + \sqrt{Nk/(4N)} < |P'-P|$ and thus $P' \notin C$.

Proof of (b). Let C be a cell with center $P \in \mathbb{R}^N \backslash A_{k/(4N)+(N-1)\sqrt{k/(4N)}}(x)$ and let P' be any point inside $A_{k/(4N)-2\sqrt{k/(4N)}}(x)$ and let P'' be the closest point to the edge of $A_{k/(4N)-2\sqrt{k/(4N)}}$. Then we have that $|P'-P| \geq |P'-P''| + N\sqrt{k/(4N)}$. But under the event $E_{k/(4N),n,x}$, the distance from P' to one of the centers is at most $|P'-P''| + \sqrt{Nk/(4N)} < |P'-P|$ and thus P is not contained in $A_{k/(4N)-2\sqrt{k/(4N)}}$.

Now this gives that under the event $E_{k/(4N),n,x}$, a cell with center in $A_{k/(4N)-(2N+2)\sqrt{k/(4N)}}(x)$ contained in $A_{k/(4N)-2\sqrt{k/(4N)}}(x)$, and thus no cell with center outside $A_{k/(4N)+(N-1)\sqrt{k/(4N)}}(x)$ cannot change by the addition of $\{0\}$ to $\mathcal{P}_A$.

We can bound the $k/(4N)+(N-1)\sqrt{k/(4N)}$ box with a sphere of radius $\sqrt{N}\big(k/(4N)+(N-1)\sqrt{k/(4N)}\big)/2 = k/(4\sqrt{N})+\sqrt{k}N/4-\sqrt{k}/4$, hence we have

$$A_{k/(4N)+(N-1)\sqrt{k/(4N)}}(x) \subseteq B_{k/(4\sqrt{N})+\sqrt{k}N/4-\sqrt{k}/4}(x)$$

We now notice that for $k \geq N^2+9$

$$\begin{aligned}\frac{k}{4\sqrt{N}}+\frac{\sqrt{k}N}{4}-\frac{\sqrt{k}}{4} &\leq \frac{k}{4}+\frac{\sqrt{k}N}{4}\\ &\leq \frac{k}{4}+\frac{3\sqrt{k}\cdot\sqrt{k}}{4}\\ &= k\end{aligned}$$

and hence for any $k \geq N^2+9$ we have

$$E_{k/(4N),n,x} \subseteq \big\{R_{A,0} \leq \frac{k}{8\sqrt{N}}+\frac{N-1}{2}\cdot\sqrt{\frac{k}{4}}\big\} \subseteq \big\{R_{A,0}\leq k\big\},$$

as desired. □

We are now ready to prove Theorem 4.2.

Proof of Theorem 4.2. By using the above we have

$$\begin{aligned}\mathbb{E}\Big[\big|K_q(\mathcal{P}_A\cup\{0\},s)\backslash K_q(\mathcal{P}_A,s)\big|^4\Big] &\leq \mathbb{E}\Big[\big|\Xi^{(0)}_{A,0}(B_{R_0,A}(0))\big|^{4q+4}\Big]\\ &\leq \sum_{k\geq 0}\mathbb{E}\Big[\big|\Xi^{(0)}_{A,0}(B_{k+1}(0))\big|^{4q+4}\cdot\mathbf{1}_{\{R_{A,0}\in(k,k+1]\}}\Big]\\ &\leq \sum_{k\geq 0}\sqrt{\mathbb{E}\Big[\big|\Xi^{(0)}_{A,0}(B_{k+1}(0))\big|^{8q+8}}\cdot\sqrt{\mathbb{P}(R_{A,0}>k)},\end{aligned}$$

where in the first inequality we used that we can limit the number of q-simplicies by the number of points raised to $q+1$ and in the last inequality we used Cauchy-Schwarz. We will now bound the two expressions inside the sum separately. We know from Lemma 7 in [5] that there exists a constant $c_2>0$, so

$$\sqrt{\mathbb{E}\Big[\big|\Xi^{(0)}_{A,0}(B_{k+1}(0))\big|^{8q+4}\Big]} \leq (k+1)^{c_2}.$$

Using Lemma 5.8, we have for $k \geq N^2 + 9$ that

$$\mathbb{P}(R_{A,0} \leq k) \geq \mathbb{P}\big(E_{k/(4N),n,x}\big)$$

and hence by using Lemma 5.7 we have there exists a $c_1 > 0$ such that

$$\mathbb{P}(R_{A,0} > k) \leq \mathbb{P}(E^c_{k/(4N),n,x}) \leq \exp(-c_1 \cdot (k/(4N))^{N/2}).$$

So we can now conclude there exists $c_1, c_2 > 0$ such that

$$\sup_{0 \in A \in \mathcal{W}} \mathbb{E}\Big[\big|K_q(\mathcal{P}_{A_n(x)} \cup \{0\}, s) \backslash K_q(\mathcal{P}_{A_n(x)}, s)\big|^4\Big] \leq \sum_{k=0}^{N^2+8} (k+1)^{c_2} \cdot \sqrt{\mathbb{P}(R_{A,0} > k)} + \sum_{k=N^2+9}^{\infty} (k+1)^{c_2} \cdot \sqrt{\exp\big(-c_1 \cdot (k/(4N))^{N/2}\big)},$$

where the last sum converges according to the quotient criterion, as desired. □

6 Simulations

In this section we will simulate Theorem 4.2 for different windows $W \subseteq \mathbb{R}^3$, and for different numbers of simulations, m. We do this because we want to show that asymptotic normality holds already for medium-sized windows. The simulations are made by simulating points from a homogeneous Poisson process in the window and finding Voronoi tessellations for these points. Next, the vertices, edges and faces are extracted from the Voronoi tessellations that were calculated and the corresponding persistence diagram is calculated. It appears that the simulation time depends linearly on the number of simulations and depends quadratically on the number of points.

We check for normality by creating a QQplot and a histogram. The histogram includes a normal distribution with mean 0 and empirical variance.

6.1 Persistent Betti tal

A typical persistence chart looks like Figure 4, so from this we choose to consider the first $(15, 25)$-persistent Betti number, $\beta_1^{15,25}$. I examine for normality by creating a QQplot and a

histogram of the standardized total persistence. The histogram includes a normal distribution with mean 0 and empirical variance.

In the following plots, the window is $W = [-200, 200]^3$, the number of points is $n = 500$ and the number of simulations, m, is given.

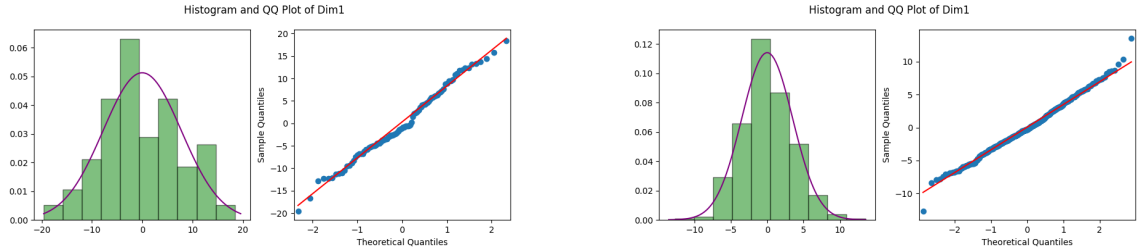

Figure 6: *Left plot is for* $m = 100$ *and right plot for* $m = 500$.

We see in Figure 6 that it looks just like a normal distribution even for a small m.

If we try to change the number of points to come from a Possison distribution with intensity $500/400^3$, and thus a mean of 500, instead of a fixed number of points, we would expect a larger spread. We see in Figure 7 that the empirical spread is about one and a half times as large here compared to before, when we had a fixed number of points. By making the window W larger, we would expect a larger spread as the points become more dispersed. In particular, we see that the empirical mean and spread for $W = [-200, 200]^3$ was $\mu = 1180$ and $s = 3.5$ respectively, while for $W = [-400, 400]^3$ it was $\mu = 295$ and $s = 1.3$.

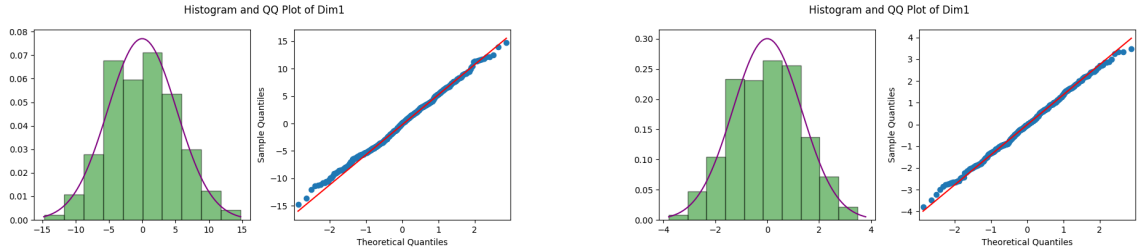

Figure 7: *Left plot is for* $m = 500$, *mean* 500 *and* $W = [-200, 200]^3$. *Right plot is for* $m = 500$, $n = 500$ *and* $W = [-400, 400]^3$.

Hypothesis test

We can define the following hypothesis we want to test

$$H_0 : \text{Fixed number of points}$$
$$H_A : \text{Poisson distribution.}$$

To test this hypothesis and find the strength of the test, we will generate data from the window $W = [-200, 200]^3$, $M = 500$ simulations and either $n = 200$ or intensity $200/400^3$, i.e. mean 200. We will repeat this 100 times and thus calculate a 95% confidence interval for the mean value based on asymptotic normality. The confidence interval became $[209.1; 214.1]$.

We will reject the null hypothesis if the empirical mean is outside the confidence interval. We can then calculate the power of this test by generating data from the alternative hypothesis and seeing how often the empirical mean is outside our confidence interval. We did this 100 times, and the empirical mean was outside the confidence interval 25 out of 100 times. So we have that the simulated power of the test is 0.25.

Similarly, we can make the opposite hypothesis, i.e.

$$H_0 : \text{Poisson distribution}$$
$$H_A : \text{Fixed number of points.}$$

and calculate the 95% confidence interval for the mean. The confidence interval was $[208.9; 216.2]$. Again, we generated data under the alternative hypothesis 100 times and an empirical mean value was in the confidence interval 1 out of 100 times. So we have the simulated power of the test is 0.01.

So we can conclude that our test is okay at detecting deviations from a fixed number of points, while less good at detecting deviations from the number of points following a Poisson distribution.

6.2 Total persistence

In [5] it is shown that there is a functional central limit theorem for the related M-bounded persistent Betti number. One can imagine that there is also a functional central limit theorem

for persistent Betti numbers. We will therefore try to examine the sum of all lifetimes, which we will call total persistence.

In the following plots, the window is $W = [-200, 200]^3$, the number of points is $n = 500$ and the number of simulations, m, is given.

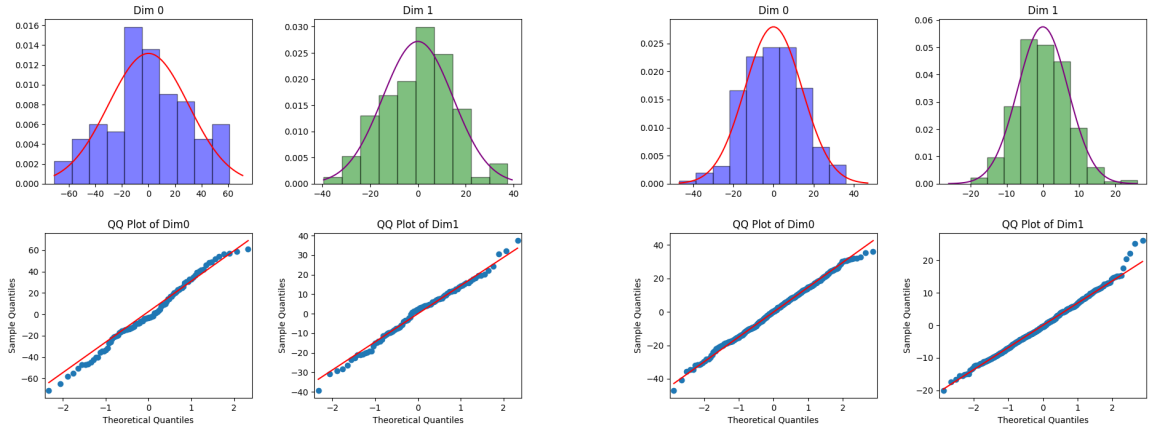

Figure 8: *Left plot is for* $m = 100$ *and right plot for* $m = 500$.

We also see in Figure 8 that for total persistence, it nicely resembles a normal distribution even for a small m.

As before, we try to change n to follow a Poisson distribution with mean 500 and try to change the window to $W = [-400, 400]^3$.

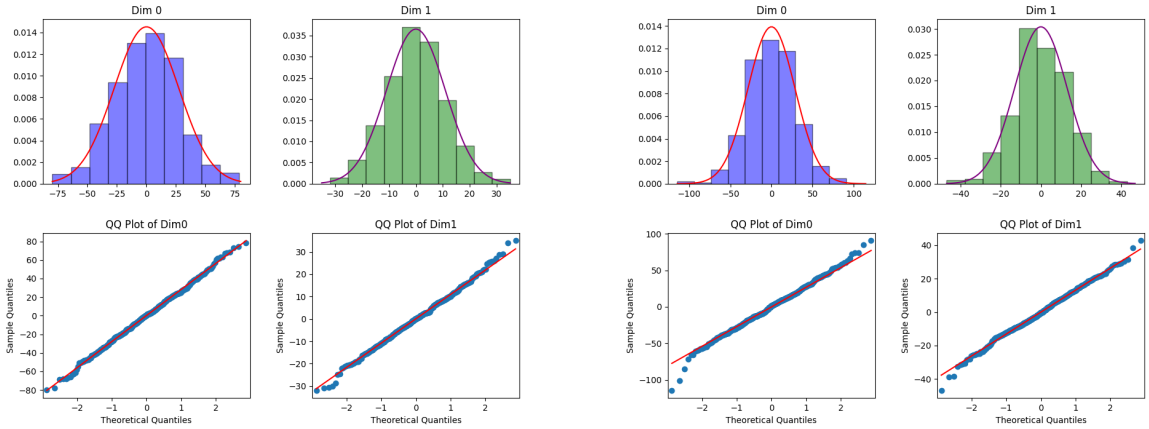

Figure 9: *Left plot is for* $m = 500$, *mean* 500 *and* $W = [-200, 200]^3$. *Right plot is for* $m = 500$, $n = 500$ *and* $W = [-400, 400]^3$.

In 9 we see the same trend as before, i.e. that the spread increases by about one and a half times when we let the number of points come from a Poisson distribution. When the window becomes larger, we again see the same trend, that the mean value and the spread become larger.

7 References

[1] Christophe A. N. Biscio, Nicolas Chenavier, Christian Hirsch, and Anne Marie Svane. Testing goodness of fit for point processes via topological data analysis. *Electronic Journal of Statistics*, 14(1):1024 – 1074, 2020.

[2] Alessandra Cipriani, Christian Hirsch, and Martina Vittorietti. Topology-based goodness-of-fit tests for sliced spatial data. *Computational Statistics & Data Analysis*, 179:107655, 2023.

[3] Yasuaki Hiraoka, Tomoyuki Shirai, and Khanh Duy Trinh. Limit theorems for persistence diagrams. *The Annals of Applied Probability*, 28(5):2740–2780, 2018.

[4] Christian Hirsch. Caput Statistics: Topological Data Analysis.

[5] Christian Hirsch, Johannes Krebs, and Claudia Redenbach. Persistent homology based goodness-of-fit tests for spatial tessellations. *arXiv preprint*, 2022.

[6] Nina Otter, Mason A Porter, Ulrike Tillmann, Peter Grindrod, and Heather A Harrington. A roadmap for the computation of persistent homology. *EPJ Data Science*, 6(1), aug 2017.

[7] Mathew D. Penrose and Joseph E. Yukich. Central limit theorems for some graphs in computational geometry. *The Annals of Applied Probability*, 11(4):1005 – 1041, 2001.

[8] Claudia Redenbach and André Liebscher. *Random Tessellations and their Application to the Modelling of Cellular Materials*, pages 73–93. Springer International Publishing, Cham, 2015.

8 Appendix

Code used to make the simulations ued in Section 6.

Computing tesselation

```
import numpy as np
from scipy.stats import gamma
from scipy.spatial.distance import pdist, euclidean
from scipy.spatial import Delaunay, ConvexHull
from tess import Container
import multiprocessing as mp
import seaborn as sns
import pyvoro
import time



start_time = time.time()



n = 500

import numpy as np
minv = np.array([-200, -200, -200])
maxv = np.array([200, 200, 200])
def containment(cell, minv = minv, maxv = maxv):
    return np.all(minv< np.min(np.vstack(cell.vertices()),
    axis = 0)) and
    np.all(maxv> np.max(np.vstack(cell.vertices()), axis = 0))

npool = 10
```

```
nsims = 10
pois = [np.random.rand(npoi, 3) * (maxv - minv) + minv
for _ in range(nsims * npool)]
print('sim done')

import pickle
import gc

def face_verts(pt):
    """Face verticces
    # Arguments
        pt: cell centers
    # Result
        Face verticces
    """
    return [np.array(c.vertices())[f]
            for c in Container(pt,
                limits = [minv, maxv],
                               radii = None,
                               periodic = True)
            for f in c.face_vertices()
            if containment(c)]

path = './sim_data/tess/'

wsize = 1
for pts, lab in zip([pois],
                    ['pois']):
    print(lab)
    def dump_vor(i):
```

```
        pickle.dump(pyvoro.compute_voronoi(pts[i],
        list(np.array(list(zip(*[minv, maxv])))),
        100, radii = [], periodic = [True] * 3),
        open('{}{}{}.p'.format(path, lab, i), 'wb'))
    [dump_vor(i) for i in range(len(pts))]

end_time = time.time()

execution_time = round(end_time - start_time)

execution_time_in_minutes = round(execution_time / 60)

print("Execution time:", execution_time_in_minutes, "minutes")
```

Extraction of corners, edges and faces

```
from scipy.cluster.hierarchy import single, fcluster
import pyvoro
from scipy.spatial import distance
import pandas as pd
import numpy as np
import time

start_time = time.time()

minv = np.array([-200, -200, -200])
maxv = np.array([200, 200, 200])
def containment(cell, minv = minv, maxv = maxv):
    return np.all(minv< np.min(np.vstack(cell['vertices']),
    axis = 0)) and
```

```
    np.all(maxv> np.max(np.vstack(cell['vertices']),
    axis = 0))

def cluster_vertices(tes, eps = 1e-5):
    """Clustering of vertices
    # Arguments
        tes: tessellation
        eps: threshold for clustering
    # Result
        clustered vertices together with list describing the
        association of the unclustered vertices to the clusters
    """

    #collect vertices from cells -- take into account periodic bc
    tes = [c for c in tes if containment(c)]
    verts_unc = [np.array(vert)
                 for cell in tes
                 for vert in cell['vertices']
                 if containment(cell)]

    cl_map = fcluster(single(distance.pdist(
    verts_unc, 'euclidean')),
    eps,
    criterion = 'distance') - 1

    verts_clust = [[p] + list(v)
                   for p, v in zip(cl_map,
                                   verts_unc)]

    return pd.DataFrame(verts_clust).groupby(0).
```

```
    mean().values, cl_map

#compute offsets for each cell to global vertex id
compute_offset = lambda tes: [0] +
list(np.cumsum([len(cell['vertices'])
                                for cell in tes]))


def clustering(tes):
    """List of cavities, faces and edges given by ids of
    clustered vertices
    # Arguments
        tes: tessellation
    # Result
        List of cavities, faces and edges given by ids of
        clustered vertices
    """
    tes = [c for c in tes if containment(c)]
    offset = compute_offset(tes)
    _, cl_map = cluster_vertices(tes)

    #unclustered cavities and their vertices
    cavs_unc = [cl_map[range(l, h)]
                for l, h in zip(offset,
                                np.roll(offset, -1))][:-1]

    #unclustered faces and their vertices
    faces_unc = [np.roll(cl_map[np.array(face['vertices']) + o],
    -np.argmin(cl_map[np.array(face['vertices']) + o]))
                            for cell, o in zip(tes,
```

```
                                        offset)
                            for face in cell['faces']
                            if containment(cell)]

#order faces
faces_unc = [list( np.roll( facc[::-1],
-np.argmin(facc[::-1]))
                    if facc[1] > facc[-1]
                    else facc)
                for facc in faces_unc]


faces_unc = [list( np.roll( facc[::-1],
-np.argmin(facc[::-1]))
                    if facc[1] > facc[-1]
                    else facc)
                for facc in faces_unc if
                len(np.unique(facc)) > 2]



#unclustered edges
edges_unc = [(cl_map[np.array(vert) + o]).tolist()
                            for cell, o in zip(tes,
                                               offset)
                            for face in cell['faces']
                            for vert in list(zip(face['vertices'],
                                 np.roll(face['vertices'], 1)))
                            if containment(cell)]
return [[list(x) for x in  set([tuple(x) for x in
```

```
    [sorted(cfe, key = kf)
    for cfe in cfes])]
    for cfes, kf in  zip([cavs_unc, faces_unc, edges_unc],
    [lambda x: 0, lambda x: 0, None])]


from multiprocessing import Pool
from scipy.spatial import KDTree, distance

import warnings
import pickle
import pandas as pd
import numpy as np


path = './sim_data/tess/'
path_cu = './sim_data/clust/'

labs = ['pois']

nsample  = 500


clustered_vor = []
verts_vor = []

for lab in labs:
    print(lab)
    with warnings.catch_warnings():
        warnings.simplefilter("ignore")
```

```
        def clv(i):
            if i%1==0:
                print(i)
    tess = pickle.load(open('{}{}{}.p'.format(path, lab, i),
    'rb'))
    pickle.dump(clustering(tess), open('{}clu{}{}.p'.format(
    path_cu, lab, i), 'wb'))
    print('clustering done')
    pickle.dump(cluster_vertices(tess)[0],
    open('{}clv{}{}.p'.format(path_cu, lab, i), 'wb'))
    del tess
        [clv(i) for i in range(nsample)]

end_time = time.time()

execution_time = round(end_time - start_time)

execution_time_in_minutes = round(execution_time / 60)

print("Execution time:", execution_time_in_minutes, "minutes")
```

Computing the persistence diagram

```
import warnings
from scipy.spatial.qhull import QhullError
import gudhi
from sklearn.decomposition import PCA
from multiprocessing import Pool
import pickle
```

```
from scipy.cluster.hierarchy import single, fcluster
from scipy.spatial import distance
import pandas as pd
import numpy as np
import time

start_time = time.time()

def cluster_vertices(tes,
                     eps = 1e-5):
    """Clustering of vertices
    # Arguments
        tes: tessellation
        eps: threshold for clustering
    # Result
        clustered vertices together with list describing the
        association of the unclustered vertices to the clusters
    """

    #collect vertices from cells -- take into account periodic bc
    verts_unc = [np.array(vert)%1
                 for cell in tes
                 for vert in cell['vertices']]

    cl_map = fcluster(single(distance.pdist(
    verts_unc, 'euclidean')),
                  eps,
                  criterion = 'distance') - 1

    verts_clust = [[p] + list(v)
```

```
                    for p, v in zip(cl_map, verts_unc)]

    return pd.DataFrame(verts_clust).groupby(0).mean().values,
    cl_map

#compute offsets for each cell to global vertex id
compute_offset = lambda tes: [0] +
list(np.cumsum([len(cell['vertices'])
                                for cell in tes]))


def clustering(tes):
    """List of cavities, faces and edges given by ids of
    clustered vertices
    # Arguments
        tes: tessellation
    # Result
        List of cavities, faces and edges given by ids of
        clustered vertices
    """
    offset = compute_offset(tes)
    _, cl_map = cluster_vertices(tes)

    #unclustered cavities and their vertices
    cavs_unc = [cl_map[range(l, h)]
                for l, h in zip(offset,
                                np.roll(offset, -1))][:-1]

    #unclustered faces and their vertices
    faces_unc = [np.roll(cl_map[np.array(face['vertices']) + o],
```

```
-np.argmin(cl_map[np.array(face['vertices']) + o]))
                        for cell, o in zip(tes,
                                           offset)
                        for face in cell['faces']]

#order faces
faces_unc = [list( np.roll( facc[::-1],
-np.argmin(facc[::-1]))
                   if facc[1] > facc[-1]
                   else facc)
             for facc in faces_unc]


faces_unc = [list( np.roll( facc[::-1],
-np.argmin(facc[::-1]))
                   if facc[1] > facc[-1]
                   else facc)
        for facc in faces_unc if len(np.unique(facc)) > 2]

#unclustered edges
edges_unc = [(cl_map[np.array(vert) + o]).tolist()
                        for cell, o in zip(tes,
                                           offset)
                        for face in cell['faces']
                        for vert in list(zip(face['vertices'],
                             np.roll(face['vertices'], 1)))]

return [np.unique([sorted(cfe,
                          key = kf)
                   for cfe in cfes],
```

```
                        axis  = a) for a, cfes, kf
                        in zip([None, None, 0],
                        [cavs_unc, faces_unc, edges_unc],
                        [lambda x: 0, lambda x: 0, None])]

import cechmate as cm
from itertools import combinations

def filt(cavs,
         faces,
         edges,
         cav_verts,
         fac_verts,
         edg_verts):
    """Compute filtration times for cavities
    # Arguments
        cavs: cavities
        faces: faces
        edges: edges
        cav_verts: coords of cavities
        fac_verts: coords of cavities
        edg_verts: coords of cavities
    # Result
        Filtration times for cavities
    """

    #pca to account for lower dimensions
    ch = cm.Alpha(maxdim  = 2,
                  verbose = False)
    pca = PCA(2)
```

```
#filtration times for cavities and faces
cf_times = [[[[sorted([cav[z] for z in x]), y]
              for  x,y in ch.build(cv)
                  if len(x) == 4]
              for cav, cv in zip(cavs, cav_verts)],
[[[sorted([face[z] for z in x]), y]
               for x, y in ch.build(pca.fit_transform(fv))
              if len(x) == 3]
          for face, fv in zip(faces, fac_verts)]]

#replace by maximal filtration time per cell/face and
add lower dimensions.
cf_times_max = [[ max(y
                      for [x, y] in cf_time)
                  if len(cf_time) > 0 else 999
                  for cf_time in cf_timess]
                  for cf_timess in cf_times]

#add lower dimensional structures to ensure it is
a simplicial complex
return [[[list(z), cf_time_max]
         for cf_time, cf_time_max in zip(cf_timess,
         cf_timess_max)
         for x, _ in cf_time
         for z in [x] + [list(v)
                         for u in [list(combinations(x, i))
                                           for i in r]
                         for v in u]]
```

```
            for cf_timess, cf_timess_max, r in zip(cf_times,
            cf_times_max,
            [range(2, 4), range(2, 3)])] \
            + [[[edge, distance.euclidean(ev[0], ev[1])/2]
                for edge, ev in zip(edges, edg_verts)]]      \
            + [[[[x], 0]
                for x in range(np.max(edges) + 1)]]


def update(*tes):
    """Update filtration times in lower dimensional structures
    # Arguments
        tes: tessellation
    # Result
        3 arrays containing the filtration times in cavities,
        faces and edges, respectively
    """
    fc, ff, fe, fv = filt(*tes)

    #cavities
    dim1c, dim2c, dim3c = [[a for a in fc
                            if len(a[0]) == i + 2]
                           for i in range(3)]

    #merge faces with cavities
    dim1f, dim2f = [[a for a in ff
                     if len(a[0]) == i + 2]
                    for i in range(2)]
    dim2fc = dim2c + dim2f
    dim2fc = [[[int(z) for z in b[:-1]], b[-1]]
```

```
                for b in pd.DataFrame([a[0] + [a[1]]
                for a in dim2fc]).groupby(
                list(range(3))).min().reset_index().values]

    #merge edges with rest
    dim1efc = fe + dim1f + dim1c
    dim1efc = [[[int(z) for z in b[:-1]], b[-1]]
                for b in pd.DataFrame([a[0] + [a[1]]
                for a in dim1efc]).groupby(
                list(range(2))).min().reset_index().values]

    return fv + dim1efc + dim2fc + dim3c

dgms = []

path_cu = './sim_data/clust/'

labs = ['pois']

nsample = 500

for lab in labs[0:1]:
    with warnings.catch_warnings():
        warnings.simplefilter("ignore")
        print(lab)

        #Create diagram
        def dgms_fun(i):
            if i%1==0:
                print(i)
```

```
            cv = pickle.load( open('{}clu{}{}.p'.format(
            path_cu,  lab, i), 'rb'))
            vw = np.array(pickle.load( open('{}clv{}{}.p'.format(
            path_cu, lab, i), 'rb')))

            vv =  ([[[x for x in vw[f]]
                           for f in cvv]
                           for cvv in cv])

            upd = update([], *(cv[1:]), [], *(vv[1:]))
            st = gudhi.SimplexTree()
            [st.insert(*splx) for splx in upd]
            pd.DataFrame(([[x[0]] +list(x[1])
                          for x in st.persistence(
                          homology_coeff_field=2)
                          if x[1][1]!= np.inf])).to_csv(
                          '{}dgm{}{}.csv'.format(
                          path_cu,  lab, i),
                          header=None, index=False)

        [dgms_fun(i) for i in range(nsample)]

end_time = time.time()

execution_time = round(end_time - start_time)

execution_time_in_minutes = round(execution_time / 60)

print("Execution time:", execution_time_in_minutes, "minutes")
```