PH-STAT

Moo K. Chung

University of Wisconsin-Madison
`mkchung@wisc.edu`

Abstract. The PH-STAT toolbox performs various persistent homology based topological data analysis in MATLAB. The code and manual are distributed in `https://github.com/laplcebeltrami/ISBI2023TDA/tree/main/PH-STAT`.

1 Simplical homology

A high dimensional object can be approximated by the point cloud data X consisting of p number of points. If we connect points of which distance satisfy a given criterion, the connected points start to recover the topology of the object. Hence, we can represent the underlying topology as a collection of the subsets of X that consists of nodes which are connected [2,5]. Given a point cloud data set X with a rule for connections, the topological space is a simplicial complex and its element is a simplex [12]. For point cloud data, the Delaunay triangulation is probably the most widely used method for connecting points. The Delaunay triangulation represents the collection of points in space as a graph whose face consists of triangles. Another way of connecting point cloud data is based on Rips complex often studied in persistent homology.

Homology is an algebraic formalism to associate a sequence of objects with a topological space [2]. In persistent homology, the algebraic formalism is usually built on top of objects that are hierarchically nested such as morse filtration, graph filtration and dendrograms. Formally homology usually refers to homology groups which are often built on top of a simplical complex for point cloud and network data [7].

The k-simplex σ is the convex hull of $v + 1$ independent points $v_0, \cdots, v_k$. A point is a 0-simplex, an edge is a 1-simplex, and a filled-in triangle is a 2-simplex. A *simplicial complex* is a finite collection of simplices such as points (0-simplex), lines (1-simplex), triangles (2-simplex) and higher dimensional counter parts [2]. A *k-skeleton* is a simplex complex of up to k simplices. Hence a graph is a 1-skeleton consisting of 0-simplices (nodes) and 1-simplices (edges). There are various simplicial complexes. The most often used simplicial complex in persistent homology is the Rips complex.

2 Rips complex

The Vietoris–Rips or Rips complex is the most often used simplicial complex in persistent homology. Let $X = \{x_0, \cdots, x_p\}$ be the set of n points in $\mathbb{R}^d$. The distance matrix between points in X is given by $w = (w_{ij})$ where w_{ij} is the distance between points x_i and x_j. Then the Rips complex $R_\epsilon(X)$ is defined as follows [1,6]. The Rips complex is a collection of simplicial complexes parameterized by ϵ. The complex $R_\epsilon(X)$ captures the topology of the point set X at a scale of ϵ or less.

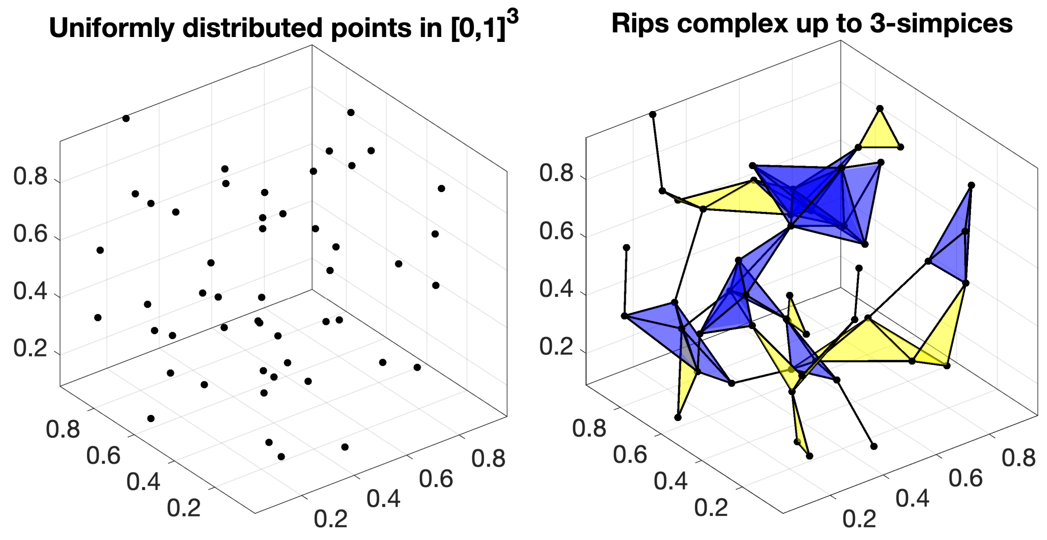


Fig. 1. Left: 50 randomly distributed points X in $[0,1]^3$. Right: Rips complex $R_{0.3}(X)$ within radius 0.3 containing 106 1-simplices, 75 2-simplices (yellow) and 22 3-simplices (blue).

– The vertices of $R_\epsilon(X)$ are the points in X.
– If the distance w_{ij} is less than or equal to ϵ, then there is an edge connecting points x_i and x_j in $R_\epsilon(X)$.
– If the distance between any two points in $x_{i_0}, x_{i_1}, \ldots, x_{i_k}$ is less than or equal to ϵ, then there is a k-simplex in $R_\epsilon(X)$ whose vertices are $x_{i_0}, x_{i_1}, \ldots, x_{i_k}$.

In practice, the Rips complex is usually computed following the above definition iteratively adding simplices of increasing dimension. Given $p+1$ number of points, there are potentially up to $\binom{p+1}{k}$ k-simplices making the data representation extremely inefficient as the radius ϵ increases. Thus, we restrict simplices of dimension up to k in practice. Such a simplical complex is called the k-skeleton. It is implemented as `PH_rips.m`, which inputs the matrix `X` of size $p \times d$, dimension `k` and radius `e`. Then outputs the structured array `S` containing the collection of nodes, edges, faces up to k-simplices. For instance, the Rips complex up to 3-simplices in Figure 1 is created using

```
p=50; d=3;
X = rand(p, d);
S= PH_rips(X, 3, 0.3)
PH_rips_display(X,S);

S =

  4×1 cell array

    { 50×1 double}
```

```
{106×2 double}
{ 75×3 double}
{ 22×4 double}
```

The Rips complex is then displayed using `PH_rips_display.m` which inputs node coordinates `X` and simplical complex `S`.

3 Boundary matrix

Given a simplicial complex K, the boundary matrices B_k represent the boundary operators between the simplices of dimension k and $k-1$. Let C_k be the collection of k-simplices. Define the k-th boundary map

$$\partial_k : C_k \to C_{k-1}$$

as a linear map that sends each k-simplex σ to a linear combination of its $k-1$ faces

$$\partial_k \sigma = \sum_{\tau \in F_k(\sigma)} (-1)^{\mathrm{sgn}(\tau,\sigma)} \tau,$$

where $F_k(\sigma)$ is the set of $k-1$ faces of σ, and $\mathrm{sgn}(\tau,\sigma)$ is the sign of the permutation that sends the vertices of τ to the vertices of σ. This expression says that the boundary of a k-simplex σ is the sum of all its $(k-1)$-dimensional faces, with appropriate signs determined by the orientation of the faces. The signs alternate between positive and negative depending on the relative orientation of the faces, as determined by the permutation that maps the vertices of one face to the vertices of the other face. The k-th boundary map removes the filled-in interior of k-simplices. The vector spaces C_k, $C_{k-1}, C_{k-2}, \cdots$ are then sequentially nested by boundary operator ∂_k [2]:

$$\cdots \xrightarrow{\partial_{k+1}} C_k \xrightarrow{\partial_k} C_{k-1} \xrightarrow{\partial_{k-1}} C_{k-2} \xrightarrow{\partial_{k-2}} \cdots . \qquad (1)$$

Such nested structure is called the *chain complex.*

Consider a filled-in triangle $\sigma = [v_1, v_2, v_3] \in C_2$ with three vertices v_1, v_2, v_3 in Figure 2. The boundary map ∂_k applied to σ resulted in the collection of three edges that forms the boundary of σ:

$$\partial_2 \sigma = [v_1, v_2] + [v_2, v_3] + [v_3, v_1] \in C_1. \qquad (2)$$

If we give the direction or orientation to edges such that

$$[v_3, v_1] = -[v_1, v_3],$$

and use edge notation $e_{ij} = [v_i, v_j]$, we can write (2) as

$$\partial_2 \sigma = e_{12} + e_{23} + e_{31} = e_{12} + e_{23} - e_{13}.$$

The boundary map can be represented as a boundary matrix $\boldsymbol{\partial}_k$ with respect to a basis of the vector spaces C_k and C_{k-1}, where the rows of $\boldsymbol{\partial}_k$ correspond to the basis elements of C_k and the columns correspond to the basis elements of C_{k-1}. The (i, j) entry of $\boldsymbol{\partial}_k$ is given by the coefficient of the jth basis element in the linear combination of the $k-1$ faces of the ith basis element in C_k. The boundary matrix is the higher dimensional version of the

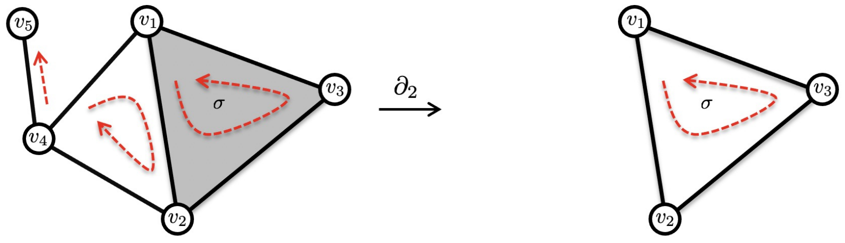


Fig. 2. A simplicial complex with 5 vertices and 2-simplex $\sigma = [v_1, v_2, v_3]$ with a filled-in face (colored gray). After boundary operation ∂_2, we are only left with 1-simplices $[v_1, v_2] + [v_2, v_3] + [v_3, v_1]$, which is the boundary of the filled in triangle. The complex has a single connected component ($\beta_0 = 1$) and a single 1-cycle. The dotted red arrows are the orientation of simplices.

incidence matrix in graphs [9,8,11] showing how $(k-1)$-dimensional simplices are forming k-dimensional simplex. The (i, j) entry of $\boldsymbol{\partial}_k$ is one if τ is a face of σ otherwise zero. The entry can be -1 depending on the orientation of τ. For the simplicial complex in Figure 2, the boundary matrices are given by

$$\boldsymbol{\partial}_2 = \begin{array}{c} \\ e_{12} \\ e_{23} \\ e_{31} \\ e_{24} \\ e_{41} \\ e_{45} \end{array} \begin{array}{c} \sigma \\ \begin{pmatrix} 1 \\ 1 \\ 1 \\ 0 \\ 0 \\ 0 \end{pmatrix} \end{array}$$

$$\boldsymbol{\partial}_1 = \begin{array}{c} \\ v_1 \\ v_2 \\ v_3 \\ v_4 \\ v_5 \end{array} \begin{array}{c} \begin{array}{cccccc} e_{12} & e_{23} & e_{31} & e_{24} & e_{41} & e_{45} \end{array} \\ \begin{pmatrix} -1 & 0 & 1 & 0 & 1 & 0 \\ 1 & -1 & 0 & -1 & 0 & 0 \\ 0 & 1 & -1 & 0 & 0 & 0 \\ 0 & 0 & 0 & 1 & -1 & -1 \\ 0 & 0 & 0 & 0 & 0 & 1 \end{pmatrix} \end{array}$$

$$\boldsymbol{\partial}_0 = \begin{array}{c} \\ 0 \end{array} \begin{array}{c} \begin{array}{ccccc} v_1 & v_2 & v_3 & v_4 & v_5 \end{array} \\ \begin{pmatrix} 0 & 0 & 0 & 0 & 0 \end{pmatrix} \end{array}.$$

In example in Figure 3-left, `PH_rips(X,3,0.5)` gives

```
>> S{1}

     1
     2
     3
     4
     5

>> S{2}
```

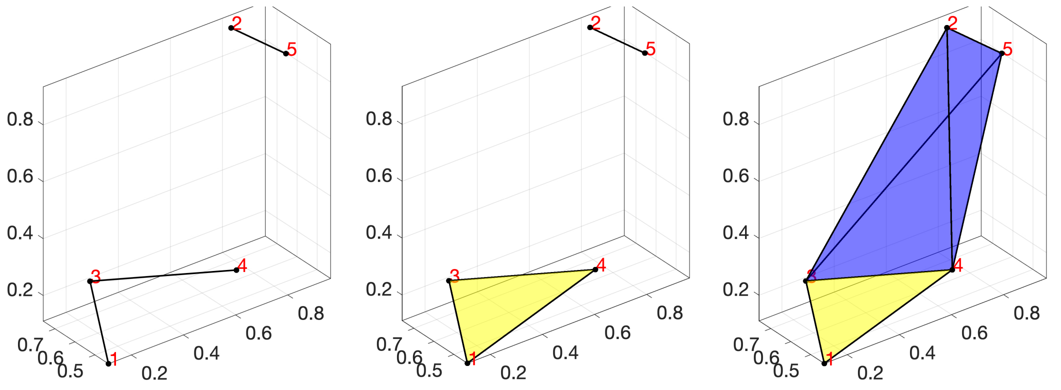


Fig. 3. Examples of boundary matrix computation. From the left to right, the radius is changed to 0.5, 0.6 and 1.0.

```
     1     3
     2     5
     3     4
```

`PH_boundary.m` only use node set `S{1}` and edge set `S{2}` in building boundary matrix `B1` saving computer memory.

```
>> B{1}

    -1     0     0
     0    -1     0
     1     0    -1
     0     0     1
     0     1     0
```

The columns of boundary matrix `B{1}` is indexed with the edge set in `S{2}` such that the first column corresponds to edge `[1,3]`. Any other potential edges `[2,3]` that are not connected is simply ignored to save computer memory.

When we increase the filtration value and compute `PH_rips(X,3,0.6)`, a triangle is formed (yellow colored) and `S{3}` is created (Figure 3-middle).

```
>>  S{2}

     1     3
     1     4
     2     5
     3     4

>> S{3}

     1     3     4
```

Correspondingly, the boundary matrices change to

```
>> B{1}

    -1    -1     0     0
     0     0    -1     0
     1     0     0    -1
     0     1     0     1
     0     0     1     0

>> B{2}

     1
    -1
     0
     1
```

From the edge set `S{2}` that forms the row index for boundary matrix `B{2}`, we have `[ 1,3] - [1, 4] + [3,4]` that forms the triangle `[1,3,4]`.

When we increase the filtration value further and compute `PH_rips(X,3,1)`, a tetrahedron is formed (blue colored) and `S{4}` is created (Figure 3-right).

```
>> S{3}

     1     3     4
     2     3     4
     2     3     5
     2     4     5
     3     4     5

>>  S{4}

     2     3     4     5
```

Correspondingly, the boundary matrix `B{3}` is created

```
>>  B{3}

     0
    -1
     1
    -1
     1
```

The easiest way to check the computation is correct is looking at the sign of triangles in `-[2,3,4] + [2,3,5] - [2,4,5] + [3,4,5]`. Using the right hand thumb rule, which puts the orientation of triangle `[3,4,5]` toward the center of the tetrahedron, the orientation of all the triangles are toward the center of the tetrahedron. Thus, the signs are correctly assigned. Since computer algorithms are built inductively, the method should work correctly in higher dimensional simplices.

4 Homology group

The image of boundary map is defined as

$$\mathrm{im}\partial_{k+1} = \{\partial_{k+1}\sigma | \sigma \in C_{k+1}\} \subset C_k,$$

which is a collection of boundaries. The elements of the image of ∂_{k+1} are called k-boundaries, and they represent k-dimensional features that can be filled in by $(k+1)$-dimensional simplices. For instance, if we take the boundary ∂_2 of the triangle σ in Figure 2, we obtain a 1-cycle with edges e_{12}, e_{23}, e_{31}. The image of the boundary matrix B_{k+1} is the subspace spanned by its columns. The column space can be found by the Gaussian elimination or singular value decomposition.

The kernel of boundary map is defined as

$$\ker\partial_k = \{\sigma \in C_k | \partial_k\sigma = 0\},$$

which is a collection of cycles. The elements of the kernel of ∂_k are called cycles, since they form closed loops or cycles in the simplicial complex. The kernel of the boundary matrix B_k is spanned by eigenvectors v corresponding to zero eigenvalues of B_k.

The boundary map satisfy the property that the composition of any two consecutive boundary maps is the zero map, i.e.,

$$\partial_{k-1} \circ \partial_k = 0. \tag{3}$$

This reflect the fact that the boundary of a boundary is always empty. We can apply the boundary operation ∂_1 further to $\partial_2\sigma$ in Figure 2 example and obtain

$$\begin{aligned}\partial_1\partial_2\sigma &= \partial_1 e_{12} + \partial_1 e_{23} + \partial_1 e_{31}\\ &= v_2 - v_1 + v_3 - v_2 + v_1 - v_3 = 0.\end{aligned}$$

This property (3) implies that the image of ∂_k is contained in the kernel of ∂_{k-1}, i.e.,

$$\mathrm{im}\partial_{k+1} \subset \ker\partial_k.$$

Further, the boundaries $\mathrm{im}\partial_{k+1}$ form subgroups of the cycles $\ker\partial_k$. We can partition $\ker\partial_k$ into cycles that differ from each other by boundaries through the quotient space

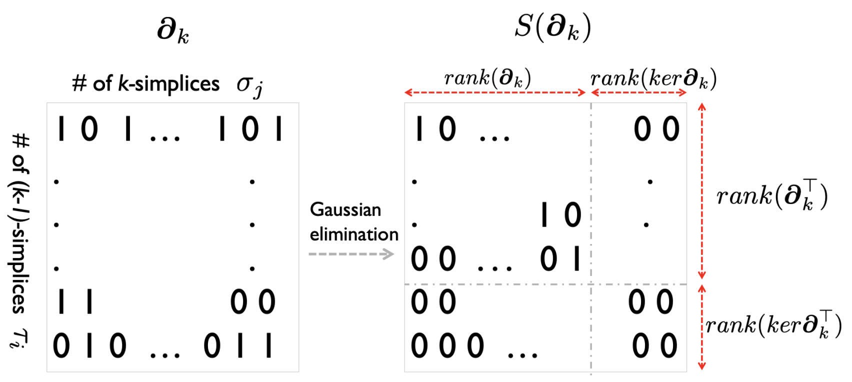
$$H_k(K) = \ker(\partial_k)/\mathrm{im}(\partial_{k+1}),$$

which is called the k-th homology group. $H_k(K)$ is a vector space that captures the kth topological feature or cycles in K. The elements of the k-th homology group are often referred to as k-dimensional cycles or k-cycles. Intuitively, it measures the presence of k-dimensional loops in the simplicial complex.

The rank of $H_k(K)$ is the kth Betti number of K, which is an algebraic invariant that captures the topological features of the complex K. Although we put direction in the boundary matrices by adding sign, the Betti number computation will be invariant of how we orient simplices. The k-th Betti number β_k is then computed as

$$\beta_k = rank(H_k) = rank(\ker\partial_k) - rank(\mathrm{im}\partial_{k+1}). \tag{4}$$

The 0-th Betti number is the number of connected components while the 1-st Betti number is the number of cycles. The Betti numbers β_k are usually algebraically computed by reducing



$\boldsymbol{\partial}_k$
$S(\boldsymbol{\partial}_k)$
# of k-simplices σ_j
# of (k-1)-simplices τ_i
Gaussian elimination
$rank(\boldsymbol{\partial}_k)$
$rank(ker\boldsymbol{\partial}_k)$
$rank(\boldsymbol{\partial}_k^\top)$
$rank(ker\boldsymbol{\partial}_k^\top)$

Fig. 4. The rank-nullity theorem for boundary matrix $\boldsymbol{\partial}_k$, which states the dimension of the domain of $\boldsymbol{\partial}_k$ is the sum of the dimension of its image and the dimension of its kernel (nullity).

the boundary matrix $\boldsymbol{\partial}_k$ to the Smith normal form $\mathcal{S}(\boldsymbol{\partial}_k)$, which has a block diagonal matrix as a submatrix in the upper left, via Gaussian elimination [2]. In the Smith normal form, we have the rank-nullity theorem for boundary matrix $\boldsymbol{\partial}_k$, which states the dimension of the domain of $\boldsymbol{\partial}_k$ is the sum of the dimension of its image and the dimension of its kernel (nullity) (Figure 4). In $\mathcal{S}(\boldsymbol{\partial}_k)$, the number of columns containing only zeros is $rank(\ker\boldsymbol{\partial}_k)$, the number of k-cycles while the number of columns containing one is $rank(\boldsymbol{\partial}_k)$, the number of k-cycles that are boundaries. Thus

$$\beta_k = rank(ker\boldsymbol{\partial}_k) - rank(\boldsymbol{\partial}_{k+1}). \tag{5}$$

The computation starts with initial rank computation

$$rank\boldsymbol{\partial}_0 = 0, \quad rank(ker\boldsymbol{\partial}_0) = p.$$

Example 1. The boundary matrices $\boldsymbol{\partial}_k$ in Figure 2 is transformed to the Smith normal form $\mathcal{S}(\boldsymbol{\partial}_k)$ after Gaussian elimination as

$$\mathcal{S}(\boldsymbol{\partial}_1) = \begin{pmatrix} 1\,0\,0\,0\,0\,0 \\ 0\,1\,0\,0\,0\,0 \\ 0\,0\,1\,0\,0\,0 \\ 0\,0\,0\,1\,0\,0 \\ 0\,0\,0\,0\,0\,0 \end{pmatrix}, \quad \mathcal{S}(\boldsymbol{\partial}_2) = \begin{pmatrix} 1 \\ 0 \\ 0 \\ 0 \\ 0 \\ 0 \end{pmatrix}.$$

From (5), the Betti number computation involves the rank computation of two boundary matrices. $rank(\boldsymbol{\partial}_0) = 5$ is trivially the number of nodes in the simplicial complex. There are $rank(\ker\boldsymbol{\partial}_1) = 2$ zero columns and $rank(\boldsymbol{\partial}_1) = 4$ non-zero row columns. $rank(\boldsymbol{\partial}_2) = 1$. Thus, we have

$$\beta_0 = rank(\ker\boldsymbol{\partial}_0) - rank(\boldsymbol{\partial}_1) = 5 - 4 = 1,$$
$$\beta_1 = rank(\ker\boldsymbol{\partial}_1) - rank(\boldsymbol{\partial}_2) = 2 - 1 = 1.$$

Following the above worked out example, the Betti number computation is implemented as `betti = PH_betti.m` which inputs the boundary matrices generated by `PH_boundary.m`. The function outputs $\beta_1, \beta_2, \cdots$. The function computes the $(d-1)$-th Betti number as

```
betti(d)=  rank(null(B{d-1})) - rank(B{d}).
```

The rank computation in most computational packages are through the singular value decomposition (SVD).

5 Persistent diagram

[Explain how to draw persistent diagrams. Then write `PH_PD.m` that draws the persistent diagram. The code should be consistently written within PH-STAT. Easiest way is to modify `PH_rips.m` to `PH_rips_update.m` with inputs the Rips complex `S` computed at `e` and incremental change `de`. Then update the Rips complex `S` at `e + de`.]

References

1. Edelsbrunner, H., Harer, J.: Persistent homology - a survey. Contemporary Mathematics **453**, 257–282 (2008)
2. Edelsbrunner, H., Harer, J.: Computational topology: An introduction. American Mathematical Society (2010)
3. Friedman, J.: Computing betti numbers via combinatorial laplacians. Algorithmica **21**(4), 331–346 (1998)
4. Ghrist, R.: Barcodes: The persistent topology of data. Bulletin of the American Mathematical Society **45**, 61–75 (2008)
5. Hart, J.: Computational topology for shape modeling. In: Proceedings of the International Conference on Shape Modeling and Applications. pp. 36–43 (1999)
6. Hatcher, A.: Algebraic topology. Cambridge University Press (2002)
7. Lee, H., Chung, M.K., K.H., Lee, D.: Hole detection in metabolic connectivity of Alzheimer's disease using k-Laplacian. MICCAI, Lecture Notes in Computer Science **8675**, 297–304 (2014)
8. Lee, H., Chung, M., Choi, H., K., H., Ha, S., Kim, Y., Lee, D.: Harmonic holes as the submodules of brain network and network dissimilarity. International Workshop on Computational Topology in Image Context, Lecture Notes in Computer Science pp. 110–122 (2019)
9. Lee, H., Chung, M., Kang, H., Choi, H., Kim, Y., Lee, D.: Abnormal hole detection in brain connectivity by kernel density of persistence diagram and Hodge Laplacian. In: IEEE International Symposium on Biomedical Imaging (ISBI). pp. 20–23 (2018)
10. Muhammad, A., Egerstedt, M.: Control using higher order laplacians in network topologies. In: Proc. of 17th International Symposium on Mathematical Theory of Networks and Systems. pp. 1024–1038 (2006)
11. Schaub, M., Benson, A., Horn, P., Lippner, G., Jadbabaie, A.: Random walks on simplicial complexes and the normalized hodge laplacian. arXiv preprint arXiv:1807.05044 (2018)
12. Zomorodian, A.: Topology for computing. Cambridge University Press, Cambridge (2009)