

PH-STAT

Moo K. Chung

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

Abstract. We introduce PH-STAT, a comprehensive Matlab toolbox designed for performing a wide range of statistical inferences on persistent homology. Persistent homology is a prominent tool in topological data analysis (TDA) that captures the underlying topological features of complex data sets. The toolbox aims to provide users with an accessible and user-friendly interface for analyzing and interpreting topological data. The package is distributed in <https://github.com/laplacebeltrami/PH-STAT>.

1 Introduction

PH-STAT (Statistical Inference on Persistent Homology) contains various statistical methods and algorithms for the analysis of persistent homology. The toolbox is designed to be compatible with a range of input data types, including point clouds, graphs, time series and functional data, graphs and networks, and simplicial complexes, allowing researchers from diverse fields to analyze their data using topological methods. The toolbox can be accessed and downloaded from the GitHub repository at <https://github.com/laplacebeltrami/PH-STAT>. The repository includes the source code, a detailed user manual, toy examples and example data sets for users to familiarize themselves with the toolbox's functionality. The user manual provides a comprehensive guide on how to run the toolbox, as well as an explanation of the underlying statistical methods and algorithms.

PH-STAT includes a rich set of statistical tools for analyzing and interpreting persistent homology, enabling users to gain valuable insights into their data. The toolbox provides visualization functions to help users understand and interpret the topological features of their data, as well as to create clear and informative plots for presentation and publication purposes. PH-STAT is an open-source project, encouraging users to contribute new features, algorithms, and improvements to the toolbox, fostering a collaborative and supportive community. PH-STAT is a versatile and powerful Matlab toolbox that facilitates the analysis of persistent homology in a user-friendly and accessible manner. By providing a comprehensive set of statistical tools, the toolbox enables researchers from various fields to harness the power of topological data analysis in their work.

2 Morse filtrations

In many applications, 1D functional data $f(t)$ is modeled as [49]

$$f(t) = \mu(t) + \epsilon(t), \quad t \in \mathbb{R}, \tag{1}$$

where μ is the unknown mean signal to be estimated and ϵ is noise. In the usual statistical parametric mapping framework [30,37,72], inference on the model (1) proceeds as follows. If

we denote an estimate of the signal by $\hat{\mu}$, the residual $f - \hat{\mu}$ gives an estimate of the noise. One then constructs a test statistic $T(t)$, corresponding to a given hypothesis about the signal. As a way to account for temporal correlation of the statistic $T(t)$, the global maximum of the test statistic over the search space \mathcal{M} is taken as the subsequent test statistic. Hence a great deal of the signal processing and statistical literature has been devoted to determining the distribution of $\sup_{t \in \mathbb{M}} T(t)$ using the random field theory [66,72], permutation tests [54] and the Hotelling–Weyl volume of tubes calculation [53]. The use of the mean signal is one way of performing data reduction, however, this may not necessarily be the best way to characterize complex multivariate imaging data. Thus instead of using the mean signal, we can use persistent homology, which pairs local critical values [24,26,76]. It is intuitive that local critical values of $\hat{\mu}$ approximately characterizes the shape of the continuous signal μ using only a finite number of scalar values. By pairing these local critical values in a nonlinear fashion and plotting them, one constructs the persistence diagram [22,24,51,74].

The function μ is called a *Morse function* if all critical values are distinct and non-degenerate, i.e., the Hessian does not vanish [50]. For a 1D Morse function $y = \mu(t)$, define sublevel set $R(y)$ as

$$R_y = \{t \in \mathbb{R} : \mu(t) \leq y\}.$$

As we increase height $y_1 \leq y_2 \leq y_3 \leq \dots$, the sublevel set gets bigger such that

$$R_{y_1} \subset R_{y_2} \subset R_{y_3} \subset \dots$$

The sequence of the sublevel sets form a *Morse filtration* with filtration values y_1, y_2, y_3, \dots . Let $\beta_0(R_y)$ be the 0-th Betti number of R_y , which counts the number of connected components in R_y . The number of connected components is the most often used topological invariant in applications [24]. $\beta_0(R_y)$ only changes its value as it passes through critical values (Figure 1). The birth and death of connected components in the Morse filtration is characterized by the pairing of local minimums and maximums. For 1D Morse functions, we do not have higher dimensional topological features beyond the connected components.

Let us denote the local minimums as g_1, \dots, g_m and the local maximums as h_1, \dots, h_n . Since the critical values of a Morse function are all distinct, we can combine all minimums and maximums and reorder them from the smallest to the largest: We further order all critical values together and let

$$g_1 = z_{(1)} < z_{(2)} < \dots < z_{(m+n)} = h_n,$$

where z_i is either h_i or g_i and $z_{(i)}$ denotes the i -th largest number in z_1, \dots, z_{m+n} . In a Morse function, g_1 is smaller than h_1 and g_m is smaller than h_n in the unbounded domain \mathbb{R} [12].

By keeping track of the birth and death of components, it is possible to compute topological invariants of sublevel sets such as the 0-th Betti number β_0 [24]. As we move y from $-\infty$ to ∞ , at a local minimum, the sublevel set adds a new component so that

$$\beta_0(R_{g_i - \epsilon}) = \beta_0(R_{g_i}) + 1$$

for sufficiently small ϵ . This process is called the *birth* of the component. The newly born component is identified with the local minimum g_i .

Similarly for at a local maximum, two components are merged as one so that

$$\beta_0(R_{h_i - \epsilon}) = \beta_0(R_{h_i}) - 1.$$

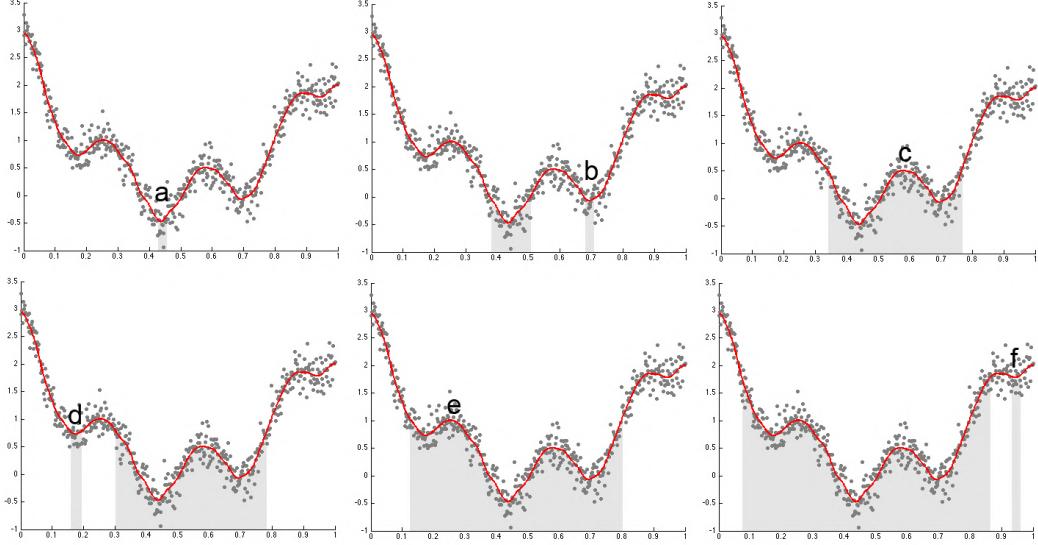


Fig. 1. The births and deaths of connected components in the sublevel sets in a Morse filtration [12]. We have local minimums $a < b < d < f$ and local maximums $c < e$. At $y = a$, we have a single connected component (gray area). As we increase the filtration value to $y = b$, we have the birth of a new component (second gray area). At the local maximum $y = c$, the two sublevel sets merge together to form a single component. This is viewed as the death of a component. The process continues till we exhaust all the critical values. Following the Elder rule, we pair birth to death: (b, c) and (d, e) . Other critical values are paired similarly. These paired points form the persistent diagram.

This process is called the *death* of the component. Since the number of connected components will only change if we pass through critical points and we can iteratively compute β_0 at each critical value as

$$\beta_0(R_{z_{(i+1)}}) = \beta_0(R_{z_{(i)}}) \pm 1.$$

The sign depends on if $z_{(i)}$ is maximum (-1) or minimum ($+1$). This is the basis of the Morse theory [50] that states that the topological characteristics of the sublevel set of Morse function are completely characterized by critical values.

To reduce the effect of low signal-to-noise ratio and to obtain smooth Morse function, either spatial or temporal smoothing have been often applied to brain imaging data before persistent homology is applied. In [15,45], Gaussian kernel smoothing was applied to 3D volumetric images. In [70], diffusion was applied to temporally smooth data.

Example 1. As an example, elder's rule is illustrated in Figure 1, where the gray dots are simulated with Gaussian noise with mean 0 and variance 0.2^2 as

$$f(x) = \mu(x) + N(0, 0.2^2)$$

with signal $\mu(t) = t + 7(t - 1/2)^2 + \cos(7\pi t)/2$. The signal μ is estimated using heat kernel smoothing [13] using degree $k = 100$ and kernel bandwidth $\sigma = 0.0001$ using `WFS_COS.m`.

Now we apply Morse filtration for filtration values y from $-\infty$ to ∞ . When we hit the first critical value $y = a$, the sublevel set consists of a single point. When we hit the minimum at $y = b$, we have the birth of a new component at b . When we hit the maximum at $y = c$, the two components identified by a and b are merged together to form a single component. When we pass through a maximum and merge two components, we pair the maximum with the higher of the two minimums of the two components [24]. Doing so we are pairing the birth of a component to its death. Obviously the paired extremes do not have to be adjacent to each other. If there is a boundary, the function value evaluated at the boundary is treated as a critical value. In the example, we need to pair (b, c) and (d, e) . Other critical values are paired similarly. The persistence diagram is then the scatter plot of these pairings computed using `PH_morse1D.m`. This is implemented as

```
t=[0:0.002:1]';
s= t + 7*(t - 0.5).^2 + cos(8*pi*t)/2;
e=normrnd(0,0.2,length(x),1);
Y=s+e;

k=100; sigma=0.0001;
[wfs, beta]=WFS_COS(Y,x,k,sigma);

pairs=PH_morse1D(x,wfs);
```

3 Simplicial 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 [33,32,25]. 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 [75]. 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 [25]. 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 simplicial complex for point cloud and network data [39].

The k -simplex σ is the convex hull of $v + 1$ independent points v_0, \dots, 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 [25]. 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.

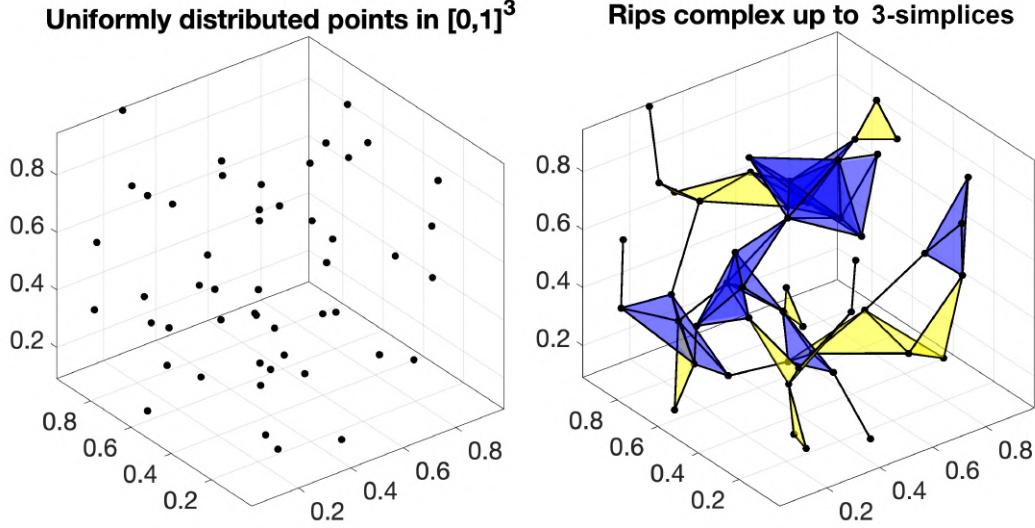


Fig. 2. Left: 50 randomly distributed points X in $[0,1]^3$. Right: Rips complex $R_{0.3}(X)$ within radius 0.3 containing 106 1-simplices, 75 2-simplices (yellow) and 22 3-simplices (blue).

3.1 Rips complex

The Vietoris–Rips or Rips complex is the most often used simplicial complex in persistent homology. Let $X = \{x_0, \dots, x_p\}$ 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 [24,34]. 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.

- 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 $x_{i_0}, x_{i_1}, \dots, x_{i_k}$ is less than or equal to ϵ , then there is a k -simplex in $R_\epsilon(X)$ whose vertices are $x_{i_0}, x_{i_1}, \dots, x_{i_k}$.

While a graph has at most 1-simplices, the Rips complex has at most k -simplices. 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 simplicial 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 2 is created using

```
p=50; d=3;
```

```

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

S =
4x1 cell array

{ 50x1 double}
{106x2 double}
{ 75x3 double}
{ 22x4 double}

```

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

3.2 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

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

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

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

where $F_k(\sigma)$ is the set of $k - 1$ faces of σ , and $\text{sgn}(\tau, \sigma)$ 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 $C_k, C_{k-1}, C_{k-2}, \dots$ are then sequentially nested by boundary operator ∂_k [25]:

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

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 3. The boundary map ∂_k applied to σ resulted in the collection of three edges that forms the boundary of σ :

$$\partial_2 \sigma = [v_1, v_2] + [v_2, v_3] + [v_3, v_1] \in C_1. \quad (3)$$

If we give the direction or orientation to edges such that

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

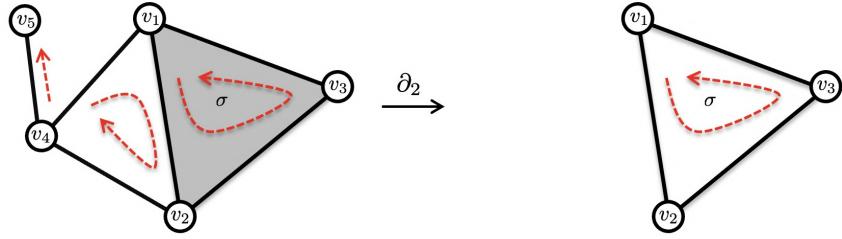


Fig. 3. A simplicial complex with 5 vertices and 2-simplex $\sigma = [v_1, v_2, v_3]$ with a filled-in face (colored gray). After boundary operation ∂_2 , we are only left with 1-simplices $[v_1, v_2] + [v_2, v_3] + [v_3, v_1]$, which is the boundary of the filled in triangle. The complex has a single connected component ($\beta_0 = 1$) and a single 1-cycle. The dotted red arrows are the orientation of simplices.

and use edge notation $e_{ij} = [v_i, v_j]$, we can write (3) 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 ∂_k with respect to a basis of the vector spaces C_k and C_{k-1} , where the rows of ∂_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 ∂_k is given by the coefficient of the j th basis element in the linear combination of the $k-1$ faces of the i th basis element in C_k . The boundary matrix is the higher dimensional version of the incidence matrix in graphs [41,40,60] showing how $(k-1)$ -dimensional simplices are forming k -dimensional simplex. The (i, j) entry of ∂_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 3, the boundary matrices are given by

$$\begin{aligned} \partial_2 &= \begin{pmatrix} \sigma \\ e_{12} & 1 \\ e_{23} & 1 \\ e_{31} & 1 \\ e_{24} & 0 \\ e_{41} & 0 \\ e_{45} & 0 \end{pmatrix} \\ \partial_1 &= \begin{pmatrix} v_1 & e_{12} & e_{23} & e_{31} & e_{24} & e_{41} & e_{45} \\ v_2 & -1 & 0 & 1 & 0 & 1 & 0 \\ v_3 & 1 & -1 & 0 & -1 & 0 & 0 \\ v_4 & 0 & 1 & -1 & 0 & 0 & 0 \\ v_5 & 0 & 0 & 0 & 1 & -1 & -1 \end{pmatrix} \\ \partial_0 &= \begin{pmatrix} v_1 & v_2 & v_3 & v_4 & v_5 \\ 0 & (0 & 0 & 0 & 0 & 0) \end{pmatrix}. \end{aligned}$$

In example in Figure 4-left, `PH_rips(X,3,0.5)` gives

>> S{1}

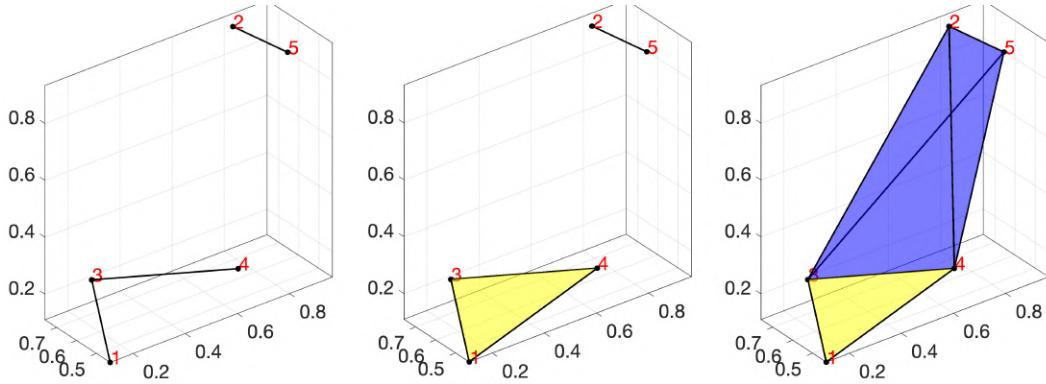


Fig. 4. Examples of boundary matrix computation. From the left to right, the radius is changed to 0.5, 0.6 and 1.0.

```

1
2
3
4
5

>> S{2}

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 4-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 4-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.

3.3 Homology group

The image of boundary map is defined as

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

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 3, 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

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

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.,

$$\partial_{k-1} \circ \partial_k = 0. \quad (4)$$

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 3 example and obtain

$$\begin{aligned} \partial_1\partial_2\sigma &= \partial_1e_{12} + \partial_1e_{23} + \partial_1e_{31} \\ &= v_2 - v_1 + v_3 - v_2 + v_1 - v_3 = 0. \end{aligned}$$

This property (4) implies that the image of ∂_k is contained in the kernel of ∂_{k-1} , i.e.,

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

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

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

which is called the k -th homology group. $H_k(K)$ is a vector space that captures the k th topological feature or cycles in K . The elements of the k -th homology group are often

∂_k		$S(\partial_k)$	
# of k -simplices	σ_j	$rank(\partial_k)$	$rank(ker\partial_k)$
1	0	1	0
0	1	...	1
.	.	.	0
.	.	0	1
.	.	0	0
1	1	0	0
0	1	0	...
0	0	1	0

Gaussian elimination \Rightarrow

# of $(k-1)$ -simplices	τ_i	$rank(\partial_k)$	$rank(ker\partial_k)$
1	0	1	0
0	1	...	1
.	.	.	0
.	.	0	1
.	.	0	0
1	1	0	0
0	1	0	...
0	0	1	0

\downarrow

# of k -simplices	σ_j	$rank(\partial_k^\top)$	$rank(ker\partial_k^\top)$
1	0	1	0
0	1	...	1
.	.	.	0
.	.	0	1
.	.	0	0
1	1	0	0
0	1	0	...
0	0	1	0

\uparrow

Fig. 5. The rank-nullity theorem for boundary matrix ∂_k , which states the dimension of the domain of ∂_k is the sum of the dimension of its image and the dimension of its kernel (nullity).

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 k th 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

$$\beta_k = rank(H_k) = rank(ker\partial_k) - rank(im\partial_{k+1}). \quad (5)$$

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 the boundary matrix ∂_k to the Smith normal form $S(\partial_k)$, which has a block diagonal matrix as a submatrix in the upper left, via Gaussian elimination [25]. In the Smith normal form, we have the rank-nullity theorem for boundary matrix ∂_k , which states the dimension of the domain of ∂_k is the sum of the dimension of its image and the dimension of its kernel (nullity) (Figure 5). In $S(\partial_k)$, the number of columns containing only zeros is $rank(ker\partial_k)$, the number of k -cycles while the number of columns containing one is $rank(\partial_k)$, the number of k -cycles that are boundaries. Thus

$$\beta_k = rank(ker\partial_k) - rank(\partial_{k+1}). \quad (6)$$

The computation starts with initial rank computation

$$rank\partial_0 = 0, \quad rank(ker\partial_0) = p.$$

Example 2. The boundary matrices ∂_k in Figure 3 is transformed to the Smith normal form $S(\partial_k)$ after Gaussian elimination as

$$S(\partial_1) = \begin{pmatrix} 1 & 0 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 & 0 \\ 0 & 0 & 1 & 0 & 0 \\ 0 & 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 0 & 0 \end{pmatrix}, \quad S(\partial_2) = \begin{pmatrix} 1 \\ 0 \\ 0 \\ 0 \\ 0 \end{pmatrix}.$$

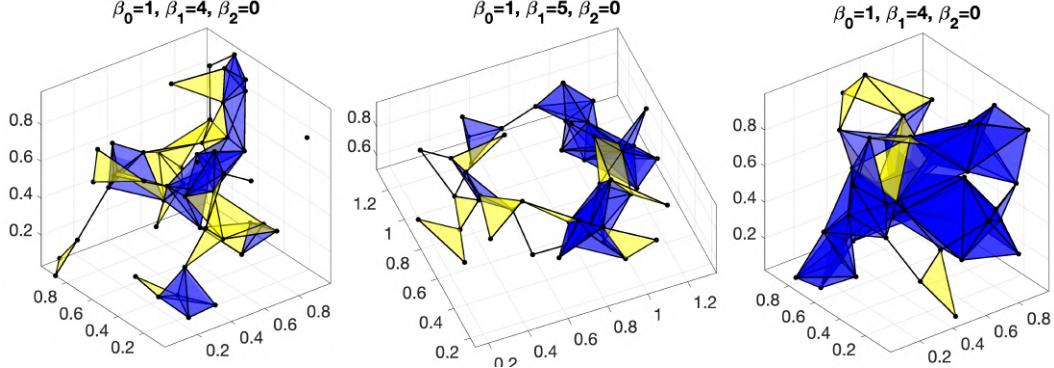


Fig. 6. Betti number computation on simplicial complex using `PH_betti.m` function.

From (6), the Betti number computation involves the rank computation of two boundary matrices. $\text{rank}(\partial_0) = 5$ is trivially the number of nodes in the simplicial complex. There are $\text{rank}(\ker \partial_1) = 2$ zero columns and $\text{rank}(\partial_1) = 4$ non-zero row columns. $\text{rank}(\partial_2) = 1$. Thus, we have

$$\begin{aligned}\beta_0 &= \text{rank}(\ker \partial_0) - \text{rank}(\partial_1) = 5 - 4 = 1, \\ \beta_1 &= \text{rank}(\ker \partial_1) - \text{rank}(\partial_2) = 2 - 1 = 1.\end{aligned}$$

Following the above worked out example, the Betti number computation is implemented in `PH_boundary_betti.m` which inputs the boundary matrices generated by `PH_boundary.m`. The function outputs β_1, β_2, \dots . The function computes the $(d-1)$ -th Betti number as

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

Figure 6 displays few examples of Betti number computation on Rips complexes. The rank computation in most computational packages is through the singular value decomposition (SVD).

3.4 Hodge Laplacian

The boundary matrix ∂_1 relates nodes to edges is commonly referred as an incidence matrix in graph theory. The boundary operator ∂_k can be represented and interpreted as as the higher dimensional version of *incidence matrix* [41,40,60]. The standard graph Laplacian can be computed using an incidence matrix as

$$\Delta_0 = \partial_1 \partial_1^\top,$$

which is also called the 0-th Hodge Laplacian [41]. In general, the k -th Hodge Laplacian is defined as

$$\Delta_k = \partial_{k+1} \partial_{k+1}^\top + \partial_k^\top \partial_k.$$

The boundary operation ∂_k depends on k -simplices.

The k -th Laplacian is a sparse $n_k \times n_k$ positive semi-definite symmetric matrix, where n_k is the number of k -simplices [29]. Then the k -th Betti number β_k is the dimension of $\ker\Delta_k$, which is given by computing the rank of Δ_k . The 0th Betti number, the number of connected component, is computed from Δ_0 while the 1st Betti number, the number of independent cycles, is computed from Δ_1 . In case of graphs, which is 1-skeletons consisting of only 0-simplices and 1-simplices, the boundary matrix $\partial_2 = 0$, thus the second term in the Hodge Laplacian Δ_1 vanishes and we have

$$\Delta_1 = \partial_1^\top \partial_1. \quad (7)$$

In brain network studies, brain networks are usually represented as graphs and thus (7) is more than sufficient unless we model higher order brain connectivity [2]. The Hodge Laplacian can be computed differently using the adjacency matrices as

$$\Delta_k = D - A^+ + (k+1)I_{n_k} + A^-,$$

where A^+ and A^- are the upper and lower adjacency matrices between the k -simplices. $D = \text{diag}(\deg(\sigma_1), \dots, \deg(\sigma_{n_k}))$ is the diagonal matrix consisting of the sum of node degrees of simplices σ_j [52].

The boundary matrices B as input, `PH_hodge.m` outputs the Hodge Laplacian as a cell array H . Then `PH_hodge_betti.m` computes the Betti numbers through the rank of kernel space of the Hodge Laplacians:

```
H=PH_hodge(B)
betti= PH_hodge_betti(H)
```

3.5 Rips filtrations

The Rips complex has the property that as the radius parameter value ϵ increases, the complex grows by adding new simplices. The simplices in the Rips complex at one radius parameter value are a subset of the simplices in the Rips complex at a larger radius parameter value. This nesting property is captured by the inclusion relation

$$\mathcal{R}\epsilon_0 \subset \mathcal{R}\epsilon_1 \subset \mathcal{R}\epsilon_2 \subset \dots$$

for $0 = \epsilon_0 \leq \epsilon_1 \leq \epsilon_2 \leq \dots$. This nested sequence of Rips complexes is called the *Rips filtration*, which is the main object of interest in persistent homology (Figure 7). The filtration values $\epsilon_0, \epsilon_1, \epsilon_2, \dots$ represent the different scales at which we are studying the topological structure of the point cloud. By increasing the filtration value ϵ , we are connecting more points, and therefore the size of the edge set, face set, and so on, increases.

The exponential growth in the number of simplices in the Rips complex as the number of vertices p increases can quickly become a computational bottleneck when working with large point clouds. For a fixed dimension k , the number of k -simplices in the Rips complex grows as $\mathcal{O}(p^k)$, which can make computations and memory usage impractical for large values of p . Furthermore, as the filtration value ϵ increases, the Rips complex becomes increasingly dense, with edges between every pair of vertices and filled triangles between every triple of vertices. Even for moderately sized point clouds, the Rips filtration can become very ineffective as a representation of the underlying data at higher filtration values. The complex becomes too dense to provide meaningful insights into the underlying topological structure of the data. To address these issues, various methods have been proposed to sparsify the Rips

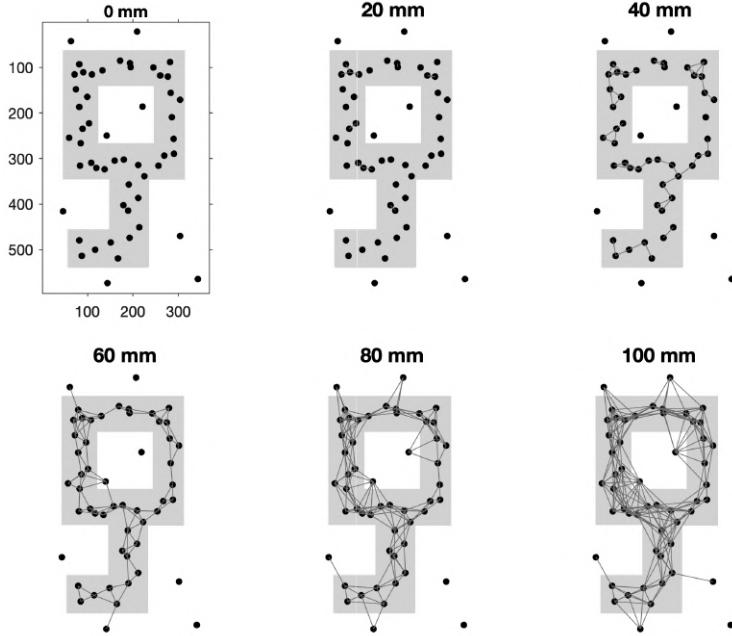


Fig. 7. Rips filtration on 1-skeleton of the point cloud data that was sampled along the underlying key shaped data. If two points are within the given radius, we connect them with an edge but do not form any other dimensional simplex. Such sparsity in Rips filtration can be more effective in practice. `PH_rips.m` limit the dimension of skeleton we build Rips filtrations and do not build every possible simplicial complexes.

complex. One such method is the graph filtration first proposed in [43,42], which constructs a filtration based on a weighted graph representation of the data. The graph filtration can be more effective than the Rips filtration especially when the topological features of interest are related to the graph structure of the data.

3.6 Persistent diagrams

As the radius ϵ increases, the Rips complex $R_\epsilon(X)$ grows and contains a higher-dimensional simplex that merges two lower-dimensional simplices representing the death of the two lower-dimensional features and the birth of a new higher-dimensional feature. The persistent diagram is a plot of the birth and death times of features. We start by computing the homology groups of each of the simplicial complexes in the filtration.

Let $H_k(K_i)$ denote the k^{th} homology group of the simplicial complex K_i . We then track the appearance and disappearance of each homology class across the different simplicial complexes in the filtration. The birth time of a homology class is defined as the smallest radius ϵ_b for which the class appears in the filtration, and the death time is the largest radius ϵ_d for which the class is present. We then plot each homology class as a point in the two-dimensional plane as $(\epsilon_b, \epsilon_d) \in \mathbb{R}^2$. The collection of all these points is the persistence diagram for k -th homology group.

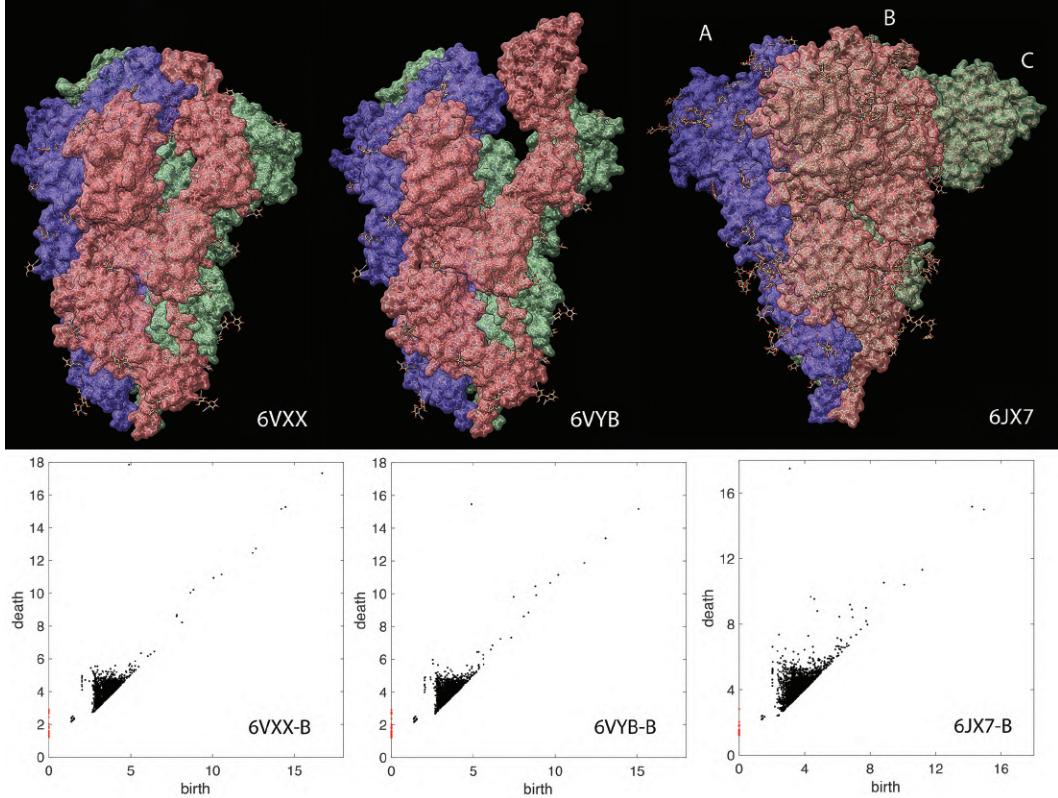


Fig. 8. Top: Spike proteins of the three different corona viruses. The spike proteins consist of three similarly shaped interwinding substructures identified as A (blue), B (red) and C (green) domains. Bottom: The persistent diagrams of spike proteins. The red dots are 0D homology and the black dots are 1D homology.

To track the birth and death times of homology classes, we need to identify when a new homology class is born or an existing homology class dies as the radius ϵ increases. We can do this by tracking the changes in the ranks of the boundary matrices. Specifically, a k -dimensional cycle is born when it appears as a new element in the kernel of ∂_k in a simplicial complex K_i that did not have it before, and it dies when it becomes a boundary in K_j for some $j > i$. Thus, we can compute the birth time ϵ_b of a k -dimensional homology class as the smallest radius for which it appears as a new element in the kernel of ∂_k . Similarly, we can compute the death time ϵ_d of the same class as the largest radius for which it is still a cycle in the simplicial complex K_j for some $j > i$. By tracking the changes in the ranks of boundary matrices, we can compute the birth and death times of homology classes and plot them in the persistence diagram for the k -th homology group. However, the computation is fairly demanding and not scale well.

Example 3. The example came from [18]. The atomic structure of spike proteins of corona virus can be determined through the cryogenic electron microscopy (cryo-EM) [7,69]. Figure

8-top displays a spike consists of three similarly shaped protein molecules with rotational symmetry often identified as A, B and C domains. The 6VXX and 6VYB are respectively the closed and open states of SARS-CoV-2 from human [69] while 6JX7 is feline coronavirus [73]. We used the atomic distances in building Rips filtrations in computing persistent diagrams. The persistent diagrams of both closed and open states are almost identical in smaller birth and death values below 6 Å (angstrom) (Figure 8-bottom). The major difference is in the scatter points with larger birth and death values. However, we need a quantitative measures for comparing the topology of closed and open states.

4 Graph filtrations

4.1 Graph filtration

The graph filtration has been the first type of filtrations applied in brain networks and it is now considered as the baseline filtrations in brain network data [43,42,44]. Euclidean distance is often used metric in building filtrations in persistent homology [25]. Most brain network studies also use the Euclidean distances for building graph filtrations [57,36,10,71,3,56]. Given weighted network $\mathcal{X} = (V, w)$ with edge weight $w = (w_{ij})$, the binary network $\mathcal{X}_\epsilon = (V, w_\epsilon)$ is a graph consisting of the node set V and the binary edge weights $w_\epsilon = (w_{\epsilon,ij})$ given by

$$w_{\epsilon,ij} = \begin{cases} 1 & \text{if } w_{ij} > \epsilon; \\ 0 & \text{otherwise.} \end{cases} \quad (8)$$

Note [42,44] defines the binary graphs by thresholding above such that $w_{\epsilon,ij} = 1$ if $w_{ij} < \epsilon$ which is consistent with the definition of the Rips filtration. However, in brain connectivity, higher value w_{ij} indicates stronger connectivity so we usually thresholds below [15].

Note w_ϵ is the adjacency matrix of \mathcal{X}_ϵ , which is a simplicial complex consisting of 0-simplices (nodes) and 1-simplices (edges) [32]. In the metric space $\mathcal{X} = (V, w)$, the Rips complex $\mathcal{R}_\epsilon(\mathcal{X})$ is a simplicial complex whose $(p - 1)$ -simplices correspond to unordered p -tuples of points that satisfy $w_{ij} \leq \epsilon$ in a pairwise fashion [32]. While the binary network \mathcal{X}_ϵ has at most 1-simplices, the Rips complex can have at most $(p - 1)$ -simplices. Thus, $\mathcal{X}_\epsilon \subset \mathcal{R}_\epsilon(\mathcal{X})$ and its compliment $\mathcal{X}_\epsilon^c \subset \mathcal{R}_\epsilon(\mathcal{X})$. Since a binary network is a special case of the Rips complex, we also have

$$\mathcal{X}_{\epsilon_0} \supset \mathcal{X}_{\epsilon_1} \supset \mathcal{X}_{\epsilon_2} \supset \dots$$

and equivalently

$$\mathcal{X}_{\epsilon_0}^c \subset \mathcal{X}_{\epsilon_1}^c \subset \mathcal{X}_{\epsilon_2}^c \subset \dots$$

for $0 = \epsilon_0 \leq \epsilon_1 \leq \epsilon_2 \dots$. The sequence of such nested multiscale graphs is defined as the *graph filtration* [42,44]. Figure 9 illustrates a graph filtration in a 4-nodes example while Figure 10 shows the graph filtration on structural covariates on maltreated children on 116 parcellated brain regions.

Note that \mathcal{X}_0 is the complete weighted graph while \mathcal{X}_∞ is the node set V . By increasing the threshold value, we are thresholding at higher connectivity so more edges are removed. Given a weighted graph, there are infinitely many different filtrations. This makes the comparisons between two different graph filtrations difficult. For a graph with p nodes, the

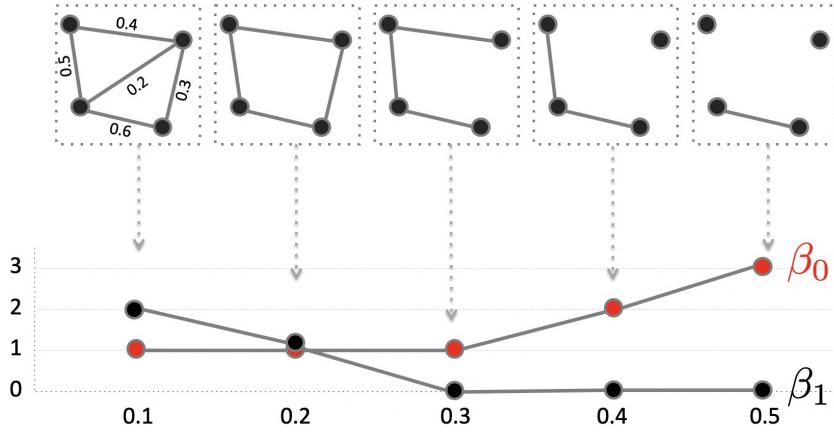


Fig. 9. Schematic of graph filtration and Betti curves. We sort the edge weights in an increasing order. We threshold the graph at filtration values and obtain binary graphs. The thresholding is performed sequentially by increasing the filtration values. The 0-th Betti number β_0 , which counts the number of connected components, and the first Betti number β_1 , which counts the number of cycles, is then plotted over the filtration. The Betti plots curves monotone in graph filtrations.

maximum number of edges is $(p^2 - p)/2$, which is obtained in a complete graph. If we order the edge weights in the increasing order, we have the sorted edge weights:

$$0 = w_{(0)} < \min_{j,k} w_{jk} = w_{(1)} < w_{(2)} < \cdots < w_{(q)} = \max_{j,k} w_{jk},$$

where $q \leq (p^2 - p)/2$. The subscript (\cdot) denotes the order statistic. For all $\lambda < w_{(1)}$, $\mathcal{X}_\lambda = \mathcal{X}_0$ is the complete graph of V . For all $w_{(r)} \leq \lambda < w_{(r+1)}$ ($r = 1, \dots, q-1$), $\mathcal{X}_\lambda = \mathcal{X}_{w_{(r)}}$. For all $w_{(q)} \leq \lambda$, $\mathcal{X}_\lambda = \mathcal{X}_{\rho_{(q)}} = V$, the vertex set. Hence, the filtration given by

$$\mathcal{X}_0 \supset \mathcal{X}_{w_{(1)}} \supset \mathcal{X}_{w_{(2)}} \supset \cdots \supset \mathcal{X}_{w_{(q)}}$$

is *maximal* in a sense that we cannot have any additional filtration \mathcal{X}_ϵ that is not one of the above filtrations. Thus, graph filtrations are usually given at edge weights [15].

The condition of having unique edge weights is not restrictive in practice. Assuming edge weights to follow some continuous distribution, the probability of any two edges being equal is zero. For discrete distribution, it may be possible to have identical edge weights. Then simply add Gaussian noise or add extremely small increasing sequence of numbers to q number of edges.

4.2 Monotone Betti curves

The graph filtration can be quantified using monotonic function f satisfying

$$f(\mathcal{X}_{\epsilon_0}) \geq f(\mathcal{X}_{\epsilon_1}) \geq f(\mathcal{X}_{\epsilon_2}) \geq \cdots \quad (9)$$

or

$$f(\mathcal{X}_{\epsilon_0}) \leq f(\mathcal{X}_{\epsilon_1}) \leq f(\mathcal{X}_{\epsilon_2}) \leq \cdots \quad (10)$$

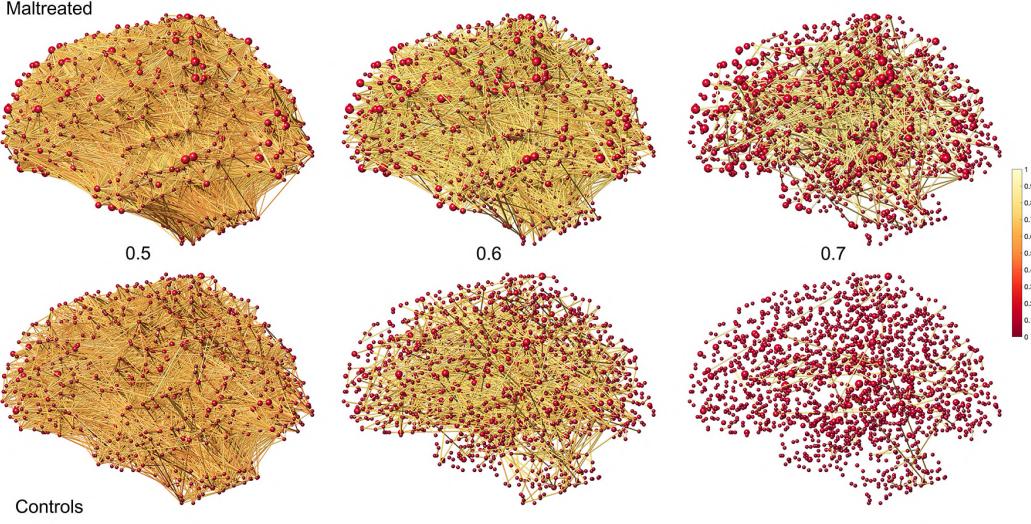


Fig. 10. Graph filtrations of maltreated children vs. normal control subjects on FA-values [15]. The Pearson correlation is used as filtration values at 0.5, 0.6 and 0.7. maltreated subjects show much higher correlation of FA-values indicating more homogeneous and less varied structural covariate relationship.

The number of connected components (zeroth Betti number β_0) and the number of cycles (first Betti number β_1) satisfy the monotonicity (Figures 9 and 11). The size of the largest cluster also satisfies a similar but opposite relation of monotonic increase. There are numerous monotone graph theory features [15,20].

For graphs, β_1 can be computed easily as a function of β_0 . Note that the Euler characteristic χ can be computed in two different ways

$$\begin{aligned}\chi &= \beta_0 - \beta_1 + \beta_2 - \dots \\ &= \#nodes - \#edges + \#faces - \dots,\end{aligned}$$

where $\#nodes$, $\#edges$, $\#faces$ are the number of nodes, edges and faces. However, graphs do not have filled faces and Betti numbers higher than β_0 and β_1 can be ignored. Thus, a graph with p nodes and q edges is given by [1]

$$\chi = \beta_0 - \beta_1 = p - q.$$

Thus,

$$\beta_1 = p - q - \beta_0.$$

In a graph, Betti numbers β_0 and β_1 are monotone over filtration on edge weights [16,17]. When we do filtration on the maximal filtration in (9), edges are deleted one at a time. Since an edge has only two end points, the deletion of an edge disconnect the graph into at most two. Thus, the number of connected components (β_0) always increases and the increase is at most by one. Note p is fixed over the filtration but q is decreasing by one while β_0 increases at most by one. Hence, β_1 always decreases and the decrease is at most by one.

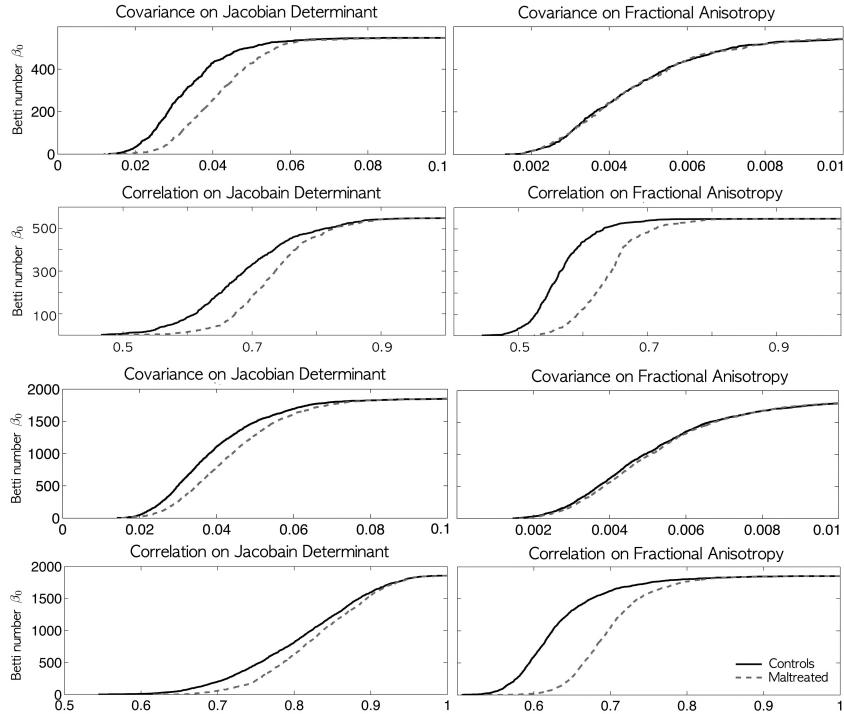


Fig. 11. The Betti curves on the covariance correlation matrices for Jacobian determinant (left column) and fractional anisotropy (right column) on 548 (top two rows) and 1856 (bottom two rows) nodes [15]. Unlike the covariance, the correlation seems to shows huge group separation between normal and maltreated children visually. However, in all 7 cases except top right (548 nodes covariance for FA), statistically significant differences were detected using the rank-sum test on the areas under the Betti-plots (p -value < 0.001). The shapes of Betti-plots are consistent between the studies with different node sizes indicating the robustness of the proposed method over changing number of nodes.

Further, the length of the largest cycles, as measured by the number of nodes, also decreases monotonically (Figure 12).

Identifying connected components in a network is important to understand in decomposing the network into disjoint subnetworks. The number of connected components (0-th Betti number) of a graph is a topological invariant that measures the number of structurally independent or disjoint subnetworks. There are many available existing algorithms, which are not related to persistent homology, for computing the number of connected components including the Dulmage-Mendelsohn decomposition [58], which has been widely used for decomposing sparse matrices into block triangular forms in speeding up matrix operations.

In graph filtrations, the number of cycles increase or decreases as the filtration value increases. The pattern of monotone increase or decrease can visually show how the topology of the graph changes over filtration values. The overall pattern of *Betti curves* can be used as a summary measure of quantifying how the graph changes over increasing edge weights

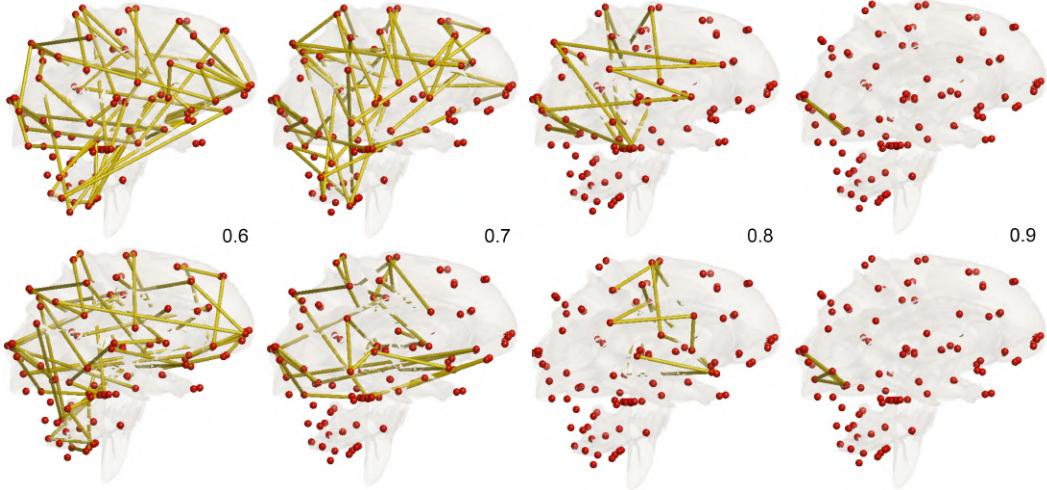


Fig. 12. The largest cycle at given correlation thresholds on rs-fMRI. Two representative subjects in HCP were used [16]. As the threshold increases, the length of cycles decreases monotonically.

[14] (Figure 9). The Betti curves are related to barcodes. The Betti number is equal to the number of bars in the barcodes at the specific filtration value.

Figure 14 displays an example of graph filtration constructed using random scatter points in a cube. Given scatter points X , the pairwise distance matrix is computed as $w = pdist2(X, X)$. The maximum distance is given by $\maxw = \max(w(:))$. Betti curves are computed using `PH_betti.m`, which inputs the pairwise distance w and the range of filtration values $[0:0.05:\maxw]$. The function outputs β_0 and β_1 values as structured arrays `beta.zero` and `beta.one`. We display them using `PH_betti_display.m`.

```
p=50; d=3;
X = rand(p, d);
w = pdist2(X, X);
maxw = max(w(:));

thresholds=[0:0.05:maxw];
beta = PH_betti(w, thresholds);
PH_betti_display(beta,thresholds)
```

4.3 Rips filtration vs. graph filtration

Persistent homology does not scale well with increased data size (Figure 13). The computational complexity of persistent homology grows rapidly with the number of simplices [67]. With p number of nodes, the size of the k -skeleton grows as p^{k+1} . Homology calculations are often done by Gaussian elimination, and if there are N simplices, it takes $\mathcal{O}(N^3)$ time to perform. In \mathbb{R}^d , the computational complexity is $\mathcal{O}(p^{3k+3})$ [63]. Thus, the computation of Rips complex is exponentially costly. It can easily become infeasible when one tries to use brain networks at the voxel level resolution. Thus, there have been many attempts

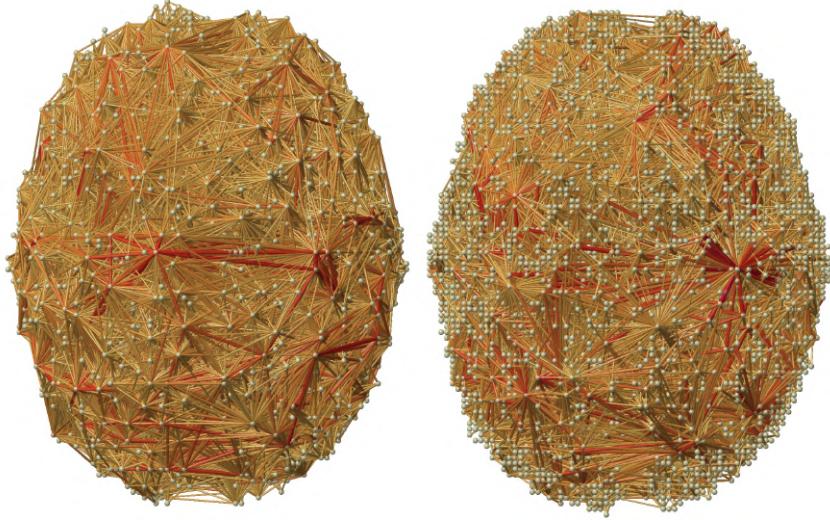


Fig. 13. rs-fMRI correlation network of two subjects from HCP with more than 25000 nodes. Identifying cycles and computing the number of cycles can be computationally demanding in this type of dense correlation network since persistent homology computations are not very scalable.

in computing Rips complex approximately but fast for large-scale data such as alpha filtration based on Delaunay triangulation with $\mathcal{O}(n^2)$ for $k = 3$ and sparse Rips filtration with $\mathcal{O}(n)$ simplices and $\mathcal{O}(n \log n)$ runtime [19,55,61]. However, all of these filtrations are all approximation to Rips filtration. To remedy the computational bottleneck caused by Rips filtrations, *graph filtration* was introduced particularly for network data [42,44].

The graph filtration is a special case of Rips filtration restricted to 1-simplices. If the Rips filtration up to 2-simplices is given by `PH_rips(X, 2, e)`, the graph filtration is given by `PH_rips(X, 1, maxw-e)`. In Figure 14 displays the comparison of two filtrations for randomly generated 50 nodes in a cube. In the both filtrations, β_0 -curves are monotone. However, β_1 -curve for the Rips filtration is not monotone. Further, the range of changes in β_1 is very narrow. In some randomly generated points, we can have multiple peaks in β_1 making the β_1 -curve somewhat unstable. On the other hand, the β_1 -curve for the graph filtration is monotone and gradually changing over the whole range of filtration values. This will give consistent to the β_1 -curve that is required for increasing statistical power in the group level inference.

4.4 Graph filtration in trees

Binary trees have been a popular data structure to analyze using persistent homology in recent years [4,47]. Trees and graphs are 1-skeletons, which are Rips complexes consisting of only nodes and edges. However, trees do not have 1-cycles and can be quantified using up to 0-cycles only, i.e., connected components, and higher order topological features can be simply ignored. However, [31] used somewhat inefficient filtrations in the 2D plane that increase the radius of circles from the root node or points along the circles. Such filtrations will produce persistent diagrams (PD) that spread points in 2D plane. Further, it may create 1-cycles.

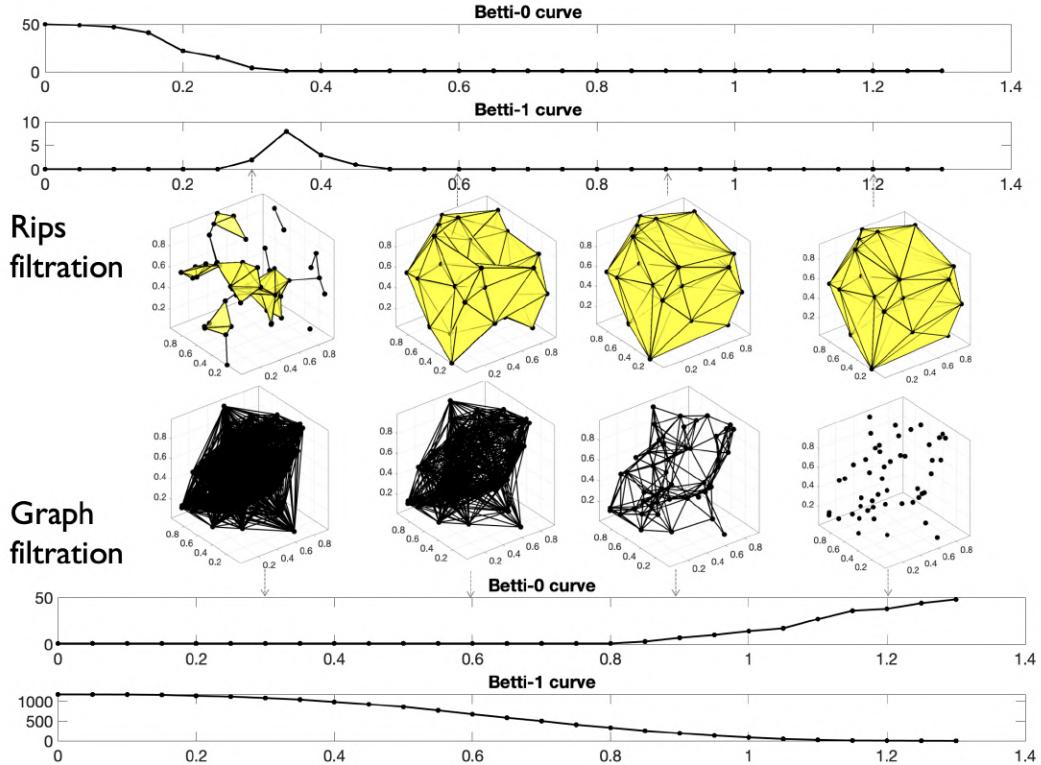


Fig. 14. The comparison between the Rips and graph filtrations.

Such PD are difficult to analyze since scatter points do not correspond across different PD. For 1-skeleton, the *graph filtration* offers more efficient alternative [17,64].

Consider a tree $\mathcal{T} = (V, w)$ with node set $V = \{1, 2, \dots, p\}$ and weighted adjacency matrix w . If we have binary tree with binary adjacency matrix, we add edge weights by taking the distance between nodes i and j as the edge weights w_{ij} and build a weighted tree with $w = (w_{ij})$. For a tree T with $p \geq 2$ nodes and unique $p - 1$ positive edge weights $w_{(1)} < w_{(2)} < \dots < w_{(p-1)}$. Threshold \mathcal{T} at ϵ and define the binary tree $\mathcal{T}_\epsilon = (V, w_\epsilon)$ with edge weights $w_\epsilon = (w_{\epsilon,ij})$, $w_{\epsilon,ij} = 1$ if $w_{ij} > \epsilon$ and 0 otherwise. Then we have graph filtration on trees

$$\mathcal{T}_{w(0)} \supset \mathcal{T}_{w(1)} \supset \mathcal{T}_{w(2)} \supset \dots \supset \mathcal{T}_{w(p-1)}. \quad (11)$$

Since all the edge weights are above filtration value $w_{(0)} = 0$, all the nodes are connected, i.e., $\beta_0(w_{(0)}) = 1$. Since no edge weight is above the threshold $w_{(p-1)}$, $\beta_0(w_{(p-1)}) = p$. Each time we threshold, the tree splits into two and the number of components increases exactly by one in the tree [17]. Thus, we have

$$\beta_0(\mathcal{T}_{w(1)}) = 2, \beta_0(\mathcal{T}_{w(2)}) = 3, \dots, \beta_0(\mathcal{T}_{w(p-1)}) = p.$$

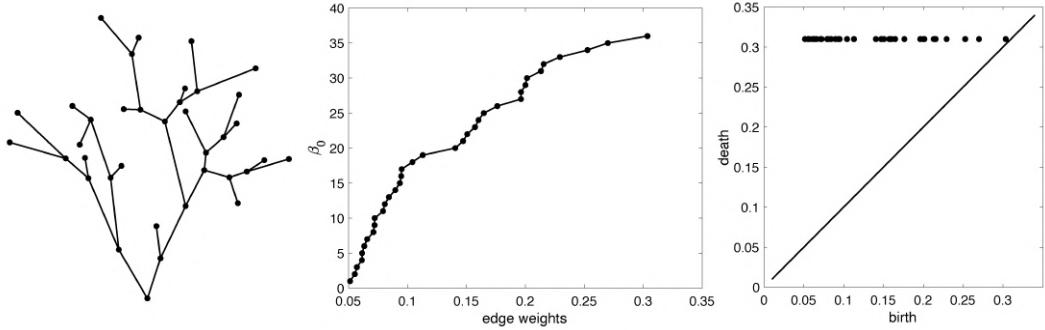


Fig. 15. Left: binary tree used in [31]. Middle: β_0 -curve over graph filtration. Edge weights of the tree is used as the filtration values. Right: The points in the persistent diagram all lined up at $y = 0.31$, which is arbitrarily picked to be larger than the maximum edge weight 0.3034.

Thus, the coordinates for the 0-th Betti curve is given by

$$(0, 1), (w_{(1)}, 2), \dots, (w_{(2)}, 3), (w_{(p-1)}, p), (\infty, p).$$

All the 0-cycles (connected components) never die once they are born over graph filtration. For convenience, we simply let the death value of 0-cycles at some fixed number $c > w_{(q-1)}$. Then PD of the graph filtration is simply

$$(w_{(1)}, c), (w_{(2)}, c), \dots, (w_{(q-1)}, c)$$

forming 1D scatter points along the horizontal line $y = c$ making various operations and analysis on PD significantly simplified [64]. Figure 15 illustrates the graph filtration and corresponding 1D scatter points in PD on the binary tree used in [31]. A different graph filtration is also possible by making the edge weight to be the shortest distance from the root node. However, they should carry the identical topological information.

For a general graph, it is not possible to analytically determine the coordinates for its Betti curves. The best we can do is to compute the number of connected components β_0 numerically using the single linkage dendrogram method (SLD) [44], the Dulmage-Mendelsohn decomposition [58,11] or through the Gaussian elimination [62,9,26].

4.5 Birth-death decomposition

Unlike the Rips complex, there are no higher dimensional topological features beyond the 0D and 1D topology in graph filtration. The 0D and 1D persistent diagrams (b_i, d_i) tabulate the life-time of 0-cycles (connected components) and 1-cycles (loops) that are born at the filtration value b_i and die at value d_i . The 0th Betti number $\beta_0(w_{(i)})$ counts the number of 0-cycles at filtration value $w_{(i)}$ and shown to be non-decreasing over filtration (Figure 16) [16]: $\beta_0(w_{(i)}) \leq \beta_0(w_{(i+1)})$. On the other hand the 1st Betti number $\beta_1(w_{(i)})$ counts the number of independent loops and shown to be non-increasing over filtration [16]: $\beta_1(w_{(i)}) \geq \beta_1(w_{(i+1)})$.

During the graph filtration, when new components is born, they never dies. Thus, 0D persistent diagrams are completely characterized by birth values b_i only. Loops are viewed

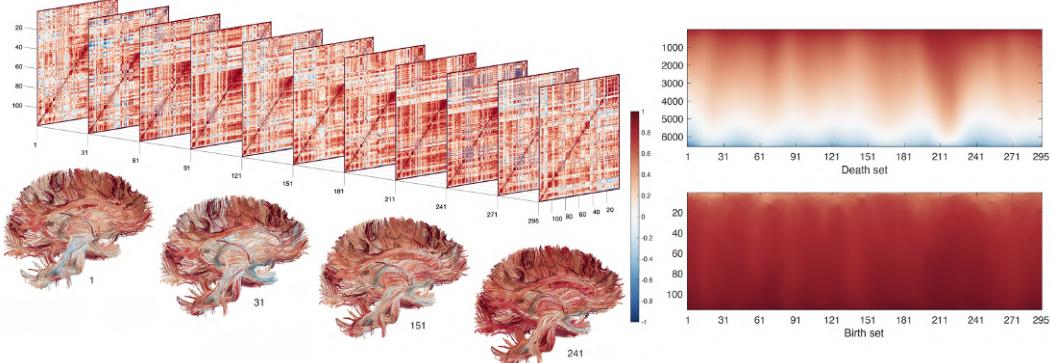


Fig. 16. The birth-death decomposition partitions the edge set into the birth and death edge sets. The birth set forms the maximum spanning tree (MST) and contains edges that create connected components (0D topology). The death set contains edges not belong to maximum spanning tree (MST) and destroys loops (1D topology).

as already born at $-\infty$. Thus, 1D persistent diagrams are completely characterized by death values d_i only. We can show that the edge weight set W can be partitioned into 0D birth values and 1D death values [65]:

Theorem 1 (Birth-death decomposition). *The edge weight set $W = \{w_{(1)}, \dots, w_{(q)}\}$ has the unique decomposition*

$$W = W_b \cup W_d, \quad W_b \cap W_d = \emptyset \quad (12)$$

where birth set $W_b = \{b_{(1)}, b_{(2)}, \dots, b_{(q_0)}\}$ is the collection of 0D sorted birth values and death set $W_d = \{d_{(1)}, d_{(2)}, \dots, d_{(q_1)}\}$ is the collection of 1D sorted death values with $q_0 = p - 1$ and $q_1 = (p - 1)(p - 2)/2$. Further W_b forms the 0D persistent diagram while W_d forms the 1D persistent diagram.

In a complete graph with p nodes, there are $q = p(p - 1)/2$ unique edge weights. There are $q_0 = p - 1$ number of edges that produces 0-cycles. This is equivalent to the number of edges in the maximum spanning tree of the graph. Thus, $q_1 = q - q_0 = \frac{(p-1)(p-2)}{2}$ number of edges destroy loops. The 0D persistent diagram is given by $\{(b_{(1)}, \infty), \dots, (b_{(q_0)}, \infty)\}$. Ignoring ∞ , W_b is the 0D persistent diagram. The 1D persistent diagram of the graph filtration is given by $\{(-\infty, d_{(1)}), \dots, (-\infty, d_{(q_1)})\}$. Ignoring $-\infty$, W_d is the 1D persistent diagram. We can show that the birth set is the maximum spanning tree (MST) (Figure 16) [64].

Numerical implementation. The identification of W_b is based on the modification to Kruskal's or Prim's algorithm and identify the MST [44]. Then W_d is identified as W/W_b . Figure 16 displays how the birth and death sets change over time for a single subject used in the study. Given edge weight matrix W as input, the Matlab function `WS_decompose.m` outputs the birth set W_b and the death set W_d .

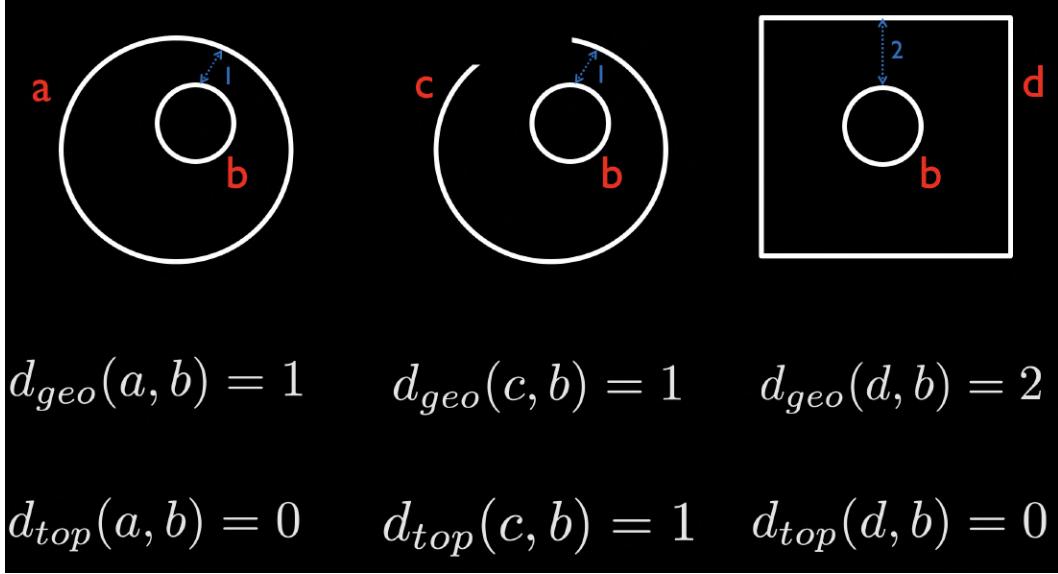


Fig. 17. Comparison between geometric distance d_{geo} and topological distance d_{top} . We used the shortest distance between objects as the geometric distance. The left and middle objects are topologically different while the left and right objects are topologically equivalent. The geometric distance cannot discriminate topologically different objects (left and middle) and produces false negatives. The geometric distance incorrectly discriminates topologically equivalent objects (left and right) and produces false positives.

5 Topological Inference

The main difference between the geometric and topological distance is if the distance can discriminate the presence of topological difference and not able to discriminate the presence of topological indifference (Figure 17). This can be achieved using the Wasserstein distance between persistent diagrams.

5.1 Wasserstein distance

Given two probability distributions $X \sim f_1$ and $Y \sim f_2$, the r -Wasserstein distance D_W , which is the probabilistic version of the optimal transport, is defined as

$$\mathcal{D}(f_1, f_2) = \left(\inf \mathbb{E} |X - Y|^r \right)^{1/r},$$

where the infimum is taken over every possible joint distributions of X and Y . The Wasserstein distance is the optimal expected cost of transporting points generated from f_1 to those generated from f_2 [8]. There are numerous distances and similarity measures defined between probability distributions such as the Kullback-Leibler (KL) divergence and the mutual information [38]. While the Wasserstein distance is a metric satisfying positive definiteness,

symmetry, and triangle inequality, the KL-divergence and the mutual information are not metric. Although they are easy to compute, the biggest limitation of the KL-divergence and the mutual information is that the two probability distributions has to be defined on the same sample space. If the two distributions do not have the same support, it may be difficult to even define them. If f_1 is discrete while f_2 is continuous, it is difficult to define them. On the other hand, the Wasserstein distance can be computed for any arbitrary distributions that may not have the common sample space making it extremely versatile.

The Wasserstein distance is often used distance for measuring the discrepancy between persistent diagrams. Consider persistent diagrams P_1 and P_2 given by

$$P_1 : x_1 = (b_1^1, d_1^1), \dots, x_q = (b_q^1, d_q^1), \quad P_2 : y_1 = (b_1^2, d_1^2), \dots, y_q = (b_q^2, d_q^2).$$

Their empirical distributions are given in terms of Dirac-delta functions

$$f_1(x) = \frac{1}{q} \sum_{i=1}^q \delta(x - x_i), \quad f_2(y) = \frac{1}{q} \sum_{i=1}^q \delta(y - y_i).$$

Then we can show that the r -Wasserstein distance on persistent diagrams is given by

$$\mathcal{D}(P_1, P_2) = \inf_{\psi: P_1 \rightarrow P_2} \left(\sum_{x \in P_1} \|x - \psi(x)\|^r \right)^{1/r} \quad (13)$$

over every possible bijection ψ , which is permutation, between P_1 and P_2 [68,8,5]. Optimization (13) is the standard assignment problem, which is usually solved by Hungarian algorithm in $\mathcal{O}(q^3)$ [27]. However, for graph filtration, the distance can be computed *exactly* in $\mathcal{O}(q \log q)$ by simply matching the order statistics on the birth or death values [59,64,65]. Note the 0D persistent diagram for graph filtration is just 1D scatter points of birth values while the 1D persistent diagram for graph filtration is just 1D scatter points of death values. Thus, the Wasserstein distance can be simplified as follows.

Theorem 2. [64] *The r -Wasserstein distance between the 0D persistent diagrams for graph filtration is given by*

$$\mathcal{D}_0(P_1, P_2) = \left[\sum_{i=1}^{q_0} (b_{(i)}^1 - b_{(i)}^2)^r \right]^{1/r},$$

where $b_{(i)}^j$ is the i -th smallest birth values in persistent diagram P_j . The 2-Wasserstein distance between the 1D persistent diagrams for graph filtration is given by

$$\mathcal{D}_1(P_1, P_2) = \left[\sum_{i=1}^{q_1} (d_{(i)}^1 - d_{(i)}^2)^r \right]^{1/r},$$

where $d_{(i)}^j$ is the i -th smallest death values in persistent diagram P_j .

In this study, we will simply use the combined 0D and 1D topological distance $\mathcal{D} = \mathcal{D}_0 + \mathcal{D}_1$ for inference and clustering. For collection of graphs `con_i` and `con_j`, the pairwise Wasserstein distance between graphs is computed as

```
lossMtx = WS_pdist2(con_i, con_j)
```

struct with fields:

```
D0: [10×10 double]
D1: [10×10 double]
D01: [10×10 double]
```

Each entry of `lossMtx` stores 0D, 1D and combined topological distances.

5.2 Topological inference

There are a few studies that used the Wasserstein distance [48,73]. The existing methods are mainly applied to geometric data without topological consideration. It is not obvious how to apply the method to perform statistical inference for a population study. We will present a new statistical inference procedure for testing the topological inference of two groups, the usual setting in brain network studies. Consider a collection of graphs $\mathcal{X}_1, \dots, \mathcal{X}_n$ that are grouped into two groups C_1 and C_2 such that

$$C_1 \cup C_2 = \{\mathcal{X}_1, \dots, \mathcal{X}_n\}, \quad C_1 \cap C_2 = \emptyset.$$

We assume there are n_i graphs in C_i and $n_1 + n_2 = n$. In the usual statistical inference, we are interested in testing the null hypothesis of the equivalence of topological summary \mathcal{T} :

$$H_0 : \mathcal{T}(C_1) = \mathcal{T}(C_2).$$

Under the null, there are $\binom{n}{n_1}$ number of permutations to permute n graphs into two groups, which is an extremely large number and most computing systems including MATLAB/R cannot compute them exactly if the sample size is larger than 50 in each group. If $n_1 = n_2$, the total number of permutations is given asymptotically by Stirling's formula [28]

$$\binom{n}{n_1} \sim \frac{4^{n_1}}{\sqrt{\pi n_1}}.$$

The number of permutations *exponentially* increases as the sample size increases, and thus it is impractical to generate every possible permutation. In practice, up to hundreds of thousands of random permutations are generated using the uniform distribution on the permutation group with probability $1/\binom{n}{n_1}$. The computational bottleneck in the permutation test is mainly caused by the need to recompute the test statistic for each permutation. This usually cause a serious computational bottleneck when we have to recompute the test statistic for large samples for more than million permutations. We propose a more scalable approach.

Define the within-group distance \mathcal{L}_W as

$$2\mathcal{L}_W = \sum_{\mathcal{X}_i, \mathcal{X}_j \in C_1} \mathcal{D}(\mathcal{X}_i, \mathcal{X}_j) + \sum_{\mathcal{X}_i, \mathcal{X}_j \in C_2} \mathcal{D}(\mathcal{X}_i, \mathcal{X}_j).$$

The within-group distance corresponds to the sum of all the pairwise distances in the block diagonal matrices in Figure 18. The average within-group distance is then given by

$$\bar{\mathcal{L}}_W = \frac{\mathcal{L}_W}{n_1(n_1 - 1) + n_2(n_2 - 1)}.$$

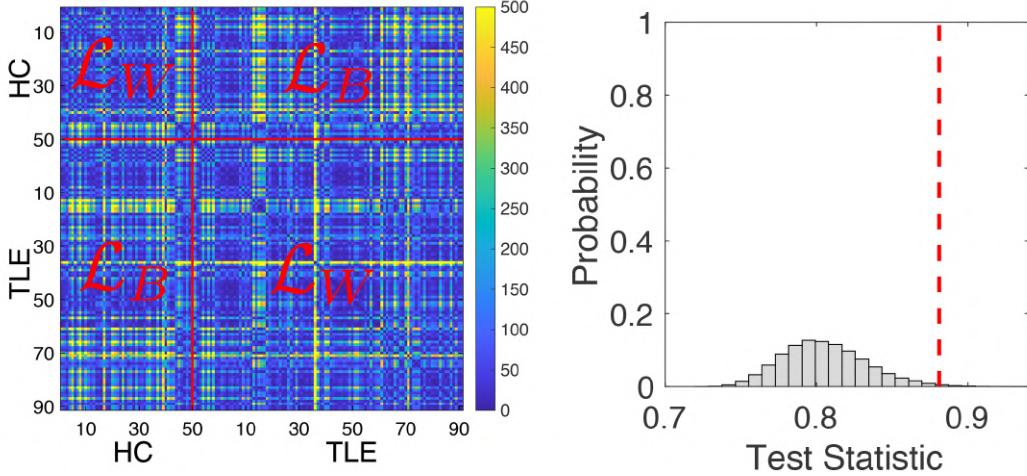


Fig. 18. Pairwise Wasserstein distance between 50 healthy controls (HC) and 101 temporal lobe epilepsy (TLE) patients. There are subtle pattern difference in the off-diagonal patterns (between group distances \mathcal{L}_B) compared to diagonal patterns (within group distances \mathcal{L}_W). The permutation test with 100 million permutations was used to determine the statistical significance using the ratio statistic. The red line is the observed ratio. The histogram is the empirical null distribution obtained from the permutation test.

The between-group distance \mathcal{L}_B is defined as

$$2\mathcal{L}_B = \sum_{\mathcal{X}_i \in C_1} \sum_{\mathcal{X}_j \in C_2} \mathcal{D}(\mathcal{X}_i, \mathcal{X}_j) + \sum_{\mathcal{X}_i \in C_2} \sum_{\mathcal{X}_j \in C_1} \mathcal{D}(\mathcal{X}_i, \mathcal{X}_j).$$

The between-group distance corresponds to the off-diagonal block matrices in Figure 18. The average between-group distance is then given by

$$\bar{\mathcal{L}}_B = \frac{\mathcal{L}_B}{n_1 n_2}.$$

Note that the sum of within-group and between-group distance is the sum of all the pairwise distances in Figure 18:

$$2\mathcal{L}_W + 2\mathcal{L}_B = \sum_{i=1}^n \sum_{j=1}^n \mathcal{D}(\mathcal{X}_i, \mathcal{X}_j).$$

When we permute the group labels, the total sum of all the pairwise distances do not change and fixed. If the group difference is large, the between-group distance \mathcal{L}_B will be large and the within-group distance \mathcal{L}_W will be small. Thus, to measure the disparity between groups as the ratio [64]

$$\phi_{\mathcal{L}} = \frac{\mathcal{L}_B}{\mathcal{L}_W}.$$

The ratio statistic is related to the elbow method in clustering and behaves like traditional F -statistic, which is the ratio of squared variability of model fits. If $\phi_{\mathcal{L}}$ is large, the groups

differ significantly in network topology. If $\phi_{\mathcal{L}}$ is small, it is likely that there is no group differences.

Since the ratio is always positive, its probability distribution cannot be Gaussian. Since the distributions of the ratio $\phi_{\mathcal{L}}$ is unknown, the permutation test can be used to determine the empirical distributions. Figure 18-right displays the empirical distribution of $\phi_{\mathcal{L}}$. The p -value is the area of the right tail thresholded by the observed ratio $\phi_{\mathcal{L}}$ (dotted red line) in the empirical distribution. Since we only compute the pairwise distances only once and only shuffle each entry over permutations. This is equivalent to rearranging rows and columns of entries corresponding to the permutations in Figure 18. The simple rearranging of rows and columns of entries and sum them in the block-wise fashion should be faster than the usual two-sample t test which has to be recomputed for each permutation.

To speed up the permutation further, we adapted the transposition test, the online version of permutation test [21]. In the transposition test, we only need to work out how \mathcal{L}_B and \mathcal{L}_W changes over a transposition, a permutation that only swaps one entry from each group. When we transpose k -th and l -th graphs between the groups (denoted as τ_{kl}), all the k -th and i -th rows and columns will be swapped. The within-group distance after the transposition τ_{kl} is given by

$$\tau_{kl}(\mathcal{L}_W) = \mathcal{L}_W + \Delta_W,$$

where Δ_W is the terms in the k -th and i -th rows and columns that are required to swapped. We only need to swap up to $\mathcal{O}(2n)$ entries while the standard permutation test that requires the computation over $\mathcal{O}(n^2)$ entries. Similarly we have incremental changes

$$\tau_{kl}(\mathcal{L}_B) = \mathcal{L}_B + \Delta_B.$$

The ratio statistic over the transposition is then sequentially updated over random transpositions. To further accelerate the convergence and avoid potential bias, we introduce one permutation to the sequence of 1000 consecutive transpositions.

The observed ratio statistic is computed using `WS_ratio.m`, which inputs the distance matrix `lossMtx`, sample size in each group. The whole procedure for performing the transposition test is implemented as `WS_transpositions.m` and takes less than one second in a desktop computer for million permutations. The function inputs the distance matrix `lossMtx`, sample size in each group, number of transpositions and the number of permutations that are interjected into transpositions. Figure 19 displays the convergence plot of the transposition test.

5.3 Topological clustering

We validate the proposed topological distances in simulations with the ground truth in a clustering setting. The Wasserstein distance was previously used for clustering for *geometric objects* without topology in $\mathcal{O}(q^3)$ [48,73]. The proposed topological method builds the Wasserstein distances on persistent diagrams in $\mathcal{O}(q \log q)$ making our method scalable. Consider a collection of graphs $\mathcal{X}_1, \dots, \mathcal{X}_n$ that will be clustered into k clusters $C = (C_1, \dots, C_k)$. Let $\mu_j = \mathbb{E}C_j$ be the topological mean of C_j computing using the Wasserstein distance. Let $\mu = (\mu_1, \dots, \mu_k)$ be the cluster mean vector. The within-cluster Wasserstein distance is given by

$$l_W(C; \mu) = \sum_{j=1}^k \sum_{X \in C_j} \mathcal{D}(X, \mu_j) = \sum_{j=1}^k |C_j| \mathbb{V}C_j$$

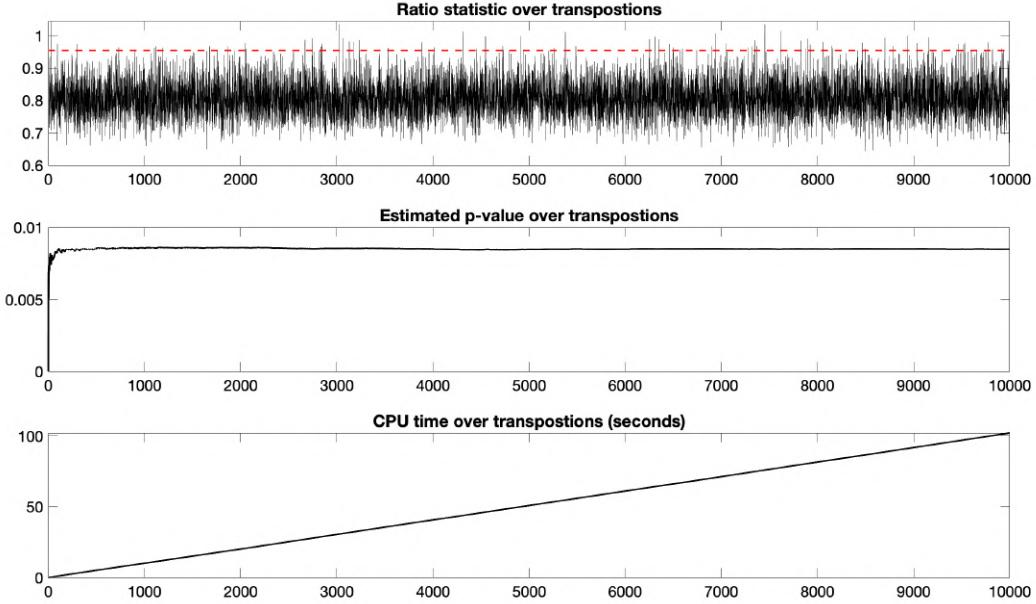


Fig. 19. The plot of ratio statistic $\phi_{\mathcal{L}}$ (top) over 100 million transpositions in testing the topological difference between HC and TLE. The plot is only shown at every 10000 transposition. The redline is the observed ratio static 0.9541. The estimated p -value (middle) converges to 0.0086 after 100 million transpositions. The CPU time (bottom) is linear and takes 102 seconds for 100 million transpositions.

with the topological variance ∇C_j of cluster C_j . The within-cluster Wasserstein distance generalizes the within-group distance defined on two groups to k number of groups (or clusters). When $k = 2$, we have $l_W(C; \mu) = 2\mathcal{L}_W$.

The topological clustering through the Wasserstein distance is then performed by minimizing $l_W(C)$ over every possible C . The Wasserstein graph clustering algorithm can be implemented as the two-step optimization often used in variational inferences [6]. The algorithm follows the proof below.

Theorem 3. *Topological clustering with the Wasserstein distance converges locally.*

Proof. 1) Expectation step: Assume C is estimated from the previous iteration. In the current iteration, the cluster mean μ corresponding to C is updated as $\mu_j \leftarrow \mathbb{E}C_j$ for each j . The cluster mean gives the lowest bound on distance $l_W(C; \nu)$ for any $\nu = (\nu_1, \dots, \nu_k)$:

$$l_W(C; \mu) = \sum_{j=1}^k \sum_{X \in C_j} \mathcal{D}(X, \mu_j) \leq \sum_{j=1}^k \sum_{X \in C_j} \mathcal{D}(X, \nu_j) = l_W(C; \nu). \quad (14)$$

2) We check if the cluster mean μ is changed from the previous iteration. If not, the algorithm simply stops. Thus we can force $l_W(C; \nu)$ to be strictly decreasing over each iteration. 3) Minimization step: The clusters are updated from C to $C' = (C'_{J_1}, \dots, C'_{J_k})$ by reassigning

each graph \mathcal{X}_i to the closest cluster C_{J_i} satisfying $J_i = \arg \min_j \mathcal{D}(\mathcal{X}_i, \mu_j)$. Subsequently, we have

$$l_W(C'; \mu) = \sum_{J_i=1}^k \sum_{X \in C'_{J_i}} \mathcal{D}(X, \mu_{J_i}) \leq \sum_{j=1}^k \sum_{X \in C_j} \mathcal{D}(X, \mu_j) = l_W(C; \mu). \quad (15)$$

From (14) and (15), $l_W(C; \mu)$ strictly decreases over iterations. Any bounded strictly decreasing sequence converges.

Just like k -means clustering that converges only to local minimum, there is no guarantee the Wasserstein graph clustering converges to the global minimum [35]. This is remedied by repeating the algorithm multiple times with different random seeds and identifying the cluster that gives the minimum over all possible seeds.

Let y_i be the true cluster label for the i -th data. Let \hat{y}_i be the estimate of y_i we determined from Wasserstein graph clustering. Let $y = (y_1, \dots, y_n)$ and $\hat{y} = (\hat{y}_1, \dots, \hat{y}_n)$. In clustering, there is no direct association between true clustering labels and predicted cluster labels. Given k clusters C_1, \dots, C_k , its permutation $\pi(C_1), \dots, \pi(C_k)$ is also a valid cluster for $\pi \in \mathbb{S}_k$, the permutation group of order k . There are $k!$ possible permutations in \mathbb{S}_k [21]. The clustering accuracy $A(y, \hat{y})$ is then given by

$$A(\hat{y}, y) = \frac{1}{n} \max_{\pi \in \mathbb{S}_k} \sum_{i=1}^n \mathbf{1}(\pi(\hat{y}) = y).$$

This a modification to an assignment problem and can be solved using the Hungarian algorithm in $\mathcal{O}(k^3)$ run time [27]. In Matlab, it can be solved using `confusionmat.m`, which tabulates misclustering errors between the true cluster labels and predicted cluster labels. Let $F(\hat{y}, y) = (f_{ij})$ be the confusion matrix of size $k \times k$ tabulating the correct number of clustering in each cluster. The diagonal entries show the correct number of clustering while the off-diagonal entries show the incorrect number of clusters. To compute the clustering accuracy, we need to sum the diagonal entries. Under the permutation of cluster labels, we can get different confusion matrices. For large k , it is prohibitive expensive to search for all permutations. Thus we need to maximize the sum of diagonals of the confusion matrix under permutation:

$$\frac{1}{n} \max_{Q \in \mathbb{S}_k} \text{tr}(QC) = \frac{1}{n} \max_{Q \in \mathbb{S}_k} \sum_{i,j} q_{ij} f_{ij}, \quad (16)$$

where $Q = (q_{ij})$ is the permutation matrix consisting of entries 0 and 1 such that there is exactly single 1 in each row and each column. This is a linear sum assignment problem (LSAP), a special case of linear assignment problem [23,46]. The clustering accuracy is computed using

```
function [accuracy C]=cluster_accuracy(ytrue,ypred)
```

where `ytrue` is the true cluster labels and `ypred` is the predicted cluster labels. `accuracy` is the clustering accuracy and `C` is the confusion matrix [23].

Example 4. We replace the Euclidean distance (L_2 -norm) in k -means clustering with the topological distance \mathcal{D} and compared the performance with the traditional k -means clustering and hierarchical clustering [42]. We generated 4 circular patterns of identical topology

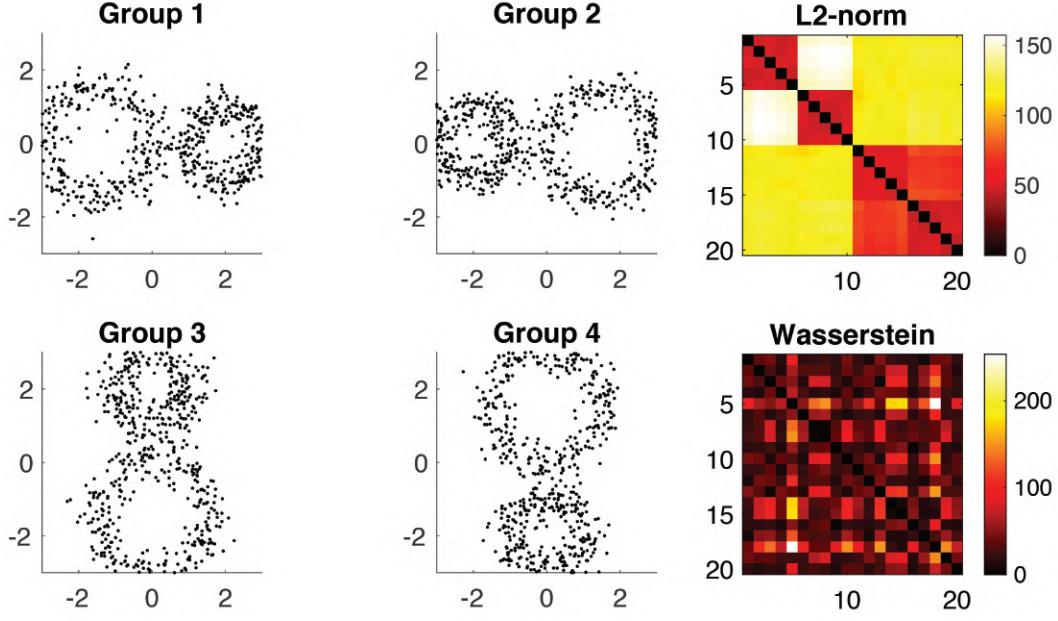


Fig. 20. Simulation study on topological equivalence. The correct clustering method should *not* be able to cluster them since they are all topologically equivalent. Right: the pairwise Euclidean distance (L_2 -norm) is used in k -means and hierarchical clustering. The Wasserstein distance is used in topological clustering.

(Figure 20) and different topology (Figure 21). Along the circles, we uniformly sampled 60 nodes and added Gaussian noise $N(0, 0.3^2)$ on the coordinates. We generated 5 random networks per group. The Euclidean distance (L_2 -norm) between randomly generated points are used to build connectivity matrices for k -means and hierarchical clustering. Figures 20 and 21 shows the superposition of nodes from 20 networks. For k -means and Wasserstein graph clustering, the average result of 100 random seeds are reported.

We tested for false positives when there is no topology difference in Figure 20, where all the groups are simply obtained from Group 1 by rotations. All the groups are topologically equivalent and thus we should not detect any topological difference. Any detected signals are all false positives. The k -means had 0.90 ± 0.15 while the hierarchical clustering had perfect 1.00 accuracy. Existing clustering methods based on Euclidean distance are reporting significant false positives and should not be used in topological clustering task had the accuracy. On the other hand, the Wasserstein graph clustering had low 0.53 ± 0.08 accuracy. We conclude that Wasserstein graph clustering are not reporting topological false positive like k -means and hierarchical clusterings.

We also tested for false negatives when there is topology difference in Figure 21, where all the groups have different number of cycles. All the groups are topologically different and thus we should detect topological differences. The k -means clustering achieved 0.83 ± 0.16 accuracy. The hierarchical clustering is reporting perfect 1.00 accuracy. On the other hand, the topological clustering achieved respectable 0.98 ± 0.09 accuracy. It is extremely difficult

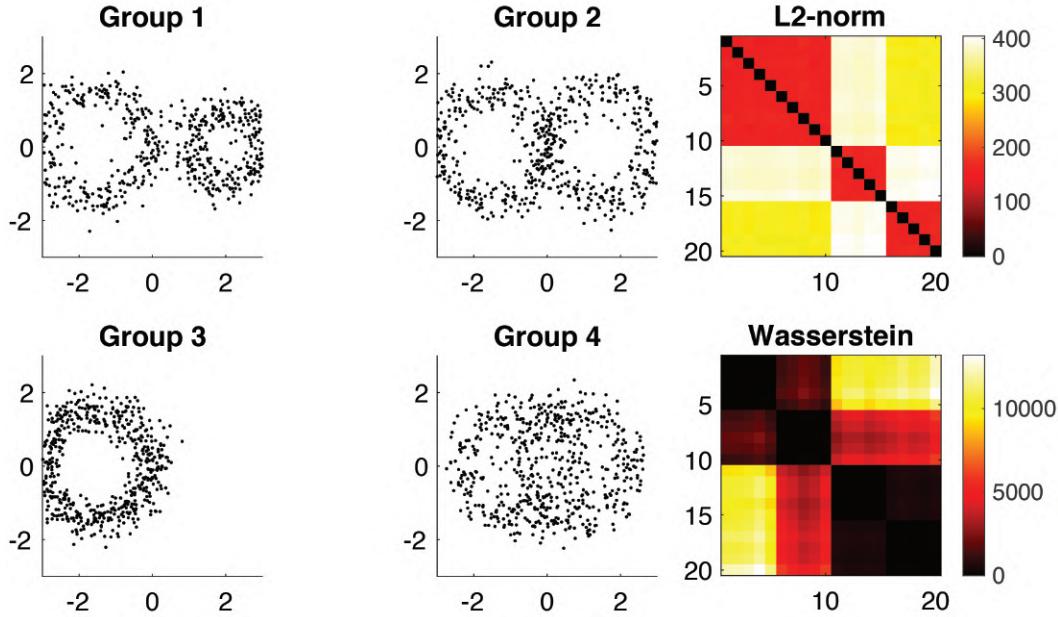


Fig. 21. Simulation study on topological difference. The correct clustering method should be able to cluster them since they are all topologically different. Right: the pairwise Euclidean distance (L_2 -norm) is used in k -means and hierarchical clustering. The Wasserstein distance is used in topological clustering.

to separate purely topological signals from geometric signals. Thus, when there is topological difference, it is expected to have geometric signal. Thus, all the methods are expected to perform well.

Existing clustering methods based on geometric distances will likely to produce significant amount of false positives and not suitable for topological learning tasks. On the other hand, the proposed Wasserstein distance performed extremely well in both cases and not likely to report false positives or false negatives. The clusterings are performed using

```
acc_WS = WS_cluster(G)
acc_K = kmeans_cluster(G)
acc_H = hierarchical_cluster(G)
```

Acknowledgement

The project is supported by NIH R01 EB028753 and NSF MDS-2010778. We also like to thank Hyekyung Lee of Seoul National University and Tananun Songdechakrakruw of University of Wisconsin-Madison for the contribution of some of functions.

References

1. Adler, R., Bobrowski, O., Borman, M., Subag, E., Weinberger, S.: Persistent homology for random fields and complexes. In: Borrowing strength: theory powering applications—a Festschrift for Lawrence D. Brown, pp. 124–143. Institute of Mathematical Statistics (2010)
2. Anand, D., Chung, M.: Hodge-Laplacian of brain networks and its application to modeling cycles. *IEEE Transactions on Medical Imaging*, in press, arXiv preprint arXiv:2110.14599 (2023)
3. Anirudh, R., Thiagarajan, J., Kim, I., Polonik, W.: Autism spectrum disorder classification using graph kernels on multidimensional time series. arXiv preprint arXiv:1611.09897 (2016)
4. Bendich, P., Marron, J., Miller, E., Pieloch, A., Skwerer, S.: Persistent homology analysis of brain artery trees. *The annals of applied statistics* **10**, 198 (2016)
5. Berwald, J., Gottlieb, J., Munch, E.: Computing wasserstein distance for persistence diagrams on a quantum computer. arXiv:1809.06433 (2018)
6. Bishop, C.: Pattern recognition and machine learning. Springer (2006)
7. Cai, Y., Zhang, J., Xiao, T., Peng, H., Sterling, S., Walsh, R., Rawson, S., Rits-Volloch, S., Chen, B.: Distinct conformational states of SARS-CoV-2 spike protein. *Science* **369**, 1586–1592 (2020)
8. Canas, G., Rosasco, L.: Learning probability measures with respect to optimal transport metrics. arXiv preprint arXiv:1209.1077 (2012)
9. Carlsson, G., Memoli, F.: Persistent clustering and a theorem of J. Kleinberg. arXiv preprint arXiv:0808.2241 (2008)
10. Cassidy, B., Rae, C., Solo, V.: Brain activity: conditional dissimilarity and persistent homology. In: IEEE 12th International Symposium on Biomedical Imaging (ISBI). pp. 1356–1359 (2015)
11. Chung, M., Adluru, N., Dalton, K., Alexander, A., Davidson, R.: Scalable brain network construction on white matter fibers. In: Proc. of SPIE. vol. 7962, p. 79624G (2011)
12. Chung, M., Bubenik, P., Kim, P.: Persistence diagrams of cortical surface data. Proceedings of the 21st International Conference on Information Processing in Medical Imaging (IPMI), Lecture Notes in Computer Science (LNCS) **5636**, 386–397 (2009)
13. Chung, M., Dalton, K., Shen, L., Evans, A., Davidson, R.: Weighted Fourier representation and its application to quantifying the amount of gray matter. *IEEE Transactions on Medical Imaging* **26**, 566–581 (2007)
14. Chung, M., Hanson, J., Lee, H., Adluru, N., Alexander, A.L., Davidson, R., Pollak, S.: Persistent homological sparse network approach to detecting white matter abnormality in maltreated children: MRI and DTI multimodal study. MICCAI, Lecture Notes in Computer Science (LNCS) **8149**, 300–307 (2013)
15. Chung, M., Hanson, J., Ye, J., Davidson, R., Pollak, S.: Persistent homology in sparse regression and its application to brain morphometry. *IEEE Transactions on Medical Imaging* **34**, 1928–1939 (2015)
16. Chung, M., Huang, S.G., Gritsenko, A., Shen, L., Lee, H.: Statistical inference on the number of cycles in brain networks. In: 2019 IEEE 16th International Symposium on Biomedical Imaging (ISBI 2019). pp. 113–116. IEEE (2019)
17. Chung, M., Lee, H., DiChristofano, A., Ombao, H., Solo, V.: Exact topological inference of the resting-state brain networks in twins. *Network Neuroscience* **3**, 674–694 (2019)
18. Chung, M., Ombao, H.: Lattice paths for persistent diagrams. In: Interpretability of Machine Intelligence in Medical Image Computing, and Topological Data Analysis and Its Applications for Medical Data, LNCS 12929, pp. 77–86 (2021)
19. Chung, M., Smith, A., Shiu, G.: Reviews: Topological distances and losses for brain networks. arXiv e-prints pp. arXiv-2102.08623 (2020), <https://arxiv.org/pdf/2102.08623.pdf>

20. Chung, M., Vilalta-Gil, V., Lee, H., Rathouz, P., Lahey, B., Zald, D.: Exact topological inference for paired brain networks via persistent homology. *Information Processing in Medical Imaging (IPMI)*, Lecture Notes in Computer Science (LNCS) **10265**, 299–310 (2017)
21. Chung, M., Xie, L., Huang, S.G., Wang, Y., Yan, J., Shen, L.: Rapid acceleration of the permutation test via transpositions **11848**, 42–53 (2019)
22. Cohen-Steiner, D., Edelsbrunner, H., Harer, J.: Stability of persistence diagrams. *Discrete and Computational Geometry* **37**, 103–120 (2007)
23. Duff, I., Koster, J.: On algorithms for permuting large entries to the diagonal of a sparse matrix. *SIAM Journal on Matrix Analysis and Applications* **22**(4), 973–996 (2001)
24. Edelsbrunner, H., Harer, J.: Persistent homology - a survey. *Contemporary Mathematics* **453**, 257–282 (2008)
25. Edelsbrunner, H., Harer, J.: Computational topology: An introduction. American Mathematical Society (2010)
26. Edelsbrunner, H., Letscher, D., Zomorodian, A.: Topological persistence and simplification. *Discrete and Computational Geometry* **28**, 511–533 (2002)
27. Edmonds, J., Karp, R.: Theoretical improvements in algorithmic efficiency for network flow problems. *Journal of the ACM (JACM)* **19**, 248–264 (1972)
28. Feller, W.: An introduction to probability theory and its applications, vol. 2. John Wiley & Sons (2008)
29. Friedman, J.: Computing betti numbers via combinatorial laplacians. *Algorithmica* **21**(4), 331–346 (1998)
30. Friston., K.: A short history of statistical parametric mapping in functional neuroimaging. *Tech. Rep.* Technical report, Wellcome Department of Imaging Neuroscience, ION, UCL., London, UK. (2002)
31. Garside, K., Gjoka, A., Henderson, R., Johnson, H., Makarenko, I.: Event history and topological data analysis. arXiv preprint arXiv:2012.08810 (2020)
32. Ghrist, R.: Barcodes: The persistent topology of data. *Bulletin of the American Mathematical Society* **45**, 61–75 (2008)
33. Hart, J.: Computational topology for shape modeling. In: *Proceedings of the International Conference on Shape Modeling and Applications*. pp. 36–43 (1999)
34. Hatcher, A.: Algebraic topology. Cambridge University Press (2002)
35. Huang, S.G., Samdin, S.T., Ting, C., Ombao, H., Chung, M.: Statistical model for dynamically-changing correlation matrices with application to brain connectivity. *Journal of Neuroscience Methods* **331**, 108480 (2020)
36. Khalid, A., Kim, B., Chung, M., Ye, J., Jeon, D.: Tracing the evolution of multi-scale functional networks in a mouse model of depression using persistent brain network homology. *NeuroImage* **101**, 351–363 (2014)
37. Kiebel, S., Poline, J.P., Friston, K., Holmes, A., Worsley, K.: Robust smoothness estimation in statistical parametric maps using standardized residuals from the general linear model. *NeuroImage* **10**, 756–766 (1999)
38. Kullback, S., Leibler, R.: On information and sufficiency. *The Annals of Mathematical Statistics* **22**, 79–86 (1951)
39. 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)
40. 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)

41. 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)
42. Lee, H., Chung, M., Kang, H., Kim, B.N., Lee, D.: Computing the shape of brain networks using graph filtration and Gromov-Hausdorff metric. MICCAI, Lecture Notes in Computer Science **6892**, 302–309 (2011)
43. Lee, H., Chung, M., Kang, H., Kim, B.N., Lee, D.: Discriminative persistent homology of brain networks. In: IEEE International Symposium on Biomedical Imaging (ISBI). pp. 841–844 (2011)
44. Lee, H., Kang, H., Chung, M., Kim, B.N., Lee, D.: Persistent brain network homology from the perspective of dendrogram. IEEE Transactions on Medical Imaging **31**, 2267–2277 (2012)
45. Lee, H., Kang, H., Chung, M., Lim, S., Kim, B.N., Lee, D.: Integrated multimodal network approach to PET and MRI based on multidimensional persistent homology. Human Brain Mapping **38**, 1387–1402 (2017)
46. Lee, M., Xiong, Y., Yu, G., Li, G.Y.: Deep neural networks for linear sum assignment problems. IEEE Wireless Communications Letters **7**, 962–965 (2018)
47. Li, Y., Wang, D., Ascoli, G., Mitra, P., Wang, Y.: Metrics for comparing neuronal tree shapes based on persistent homology. PloS one **12**(8), e0182184 (2017)
48. Mi, L., Zhang, W., Gu, X., Wang, Y.: Variational wasserstein clustering. In: Proceedings of the European Conference on Computer Vision (ECCV). pp. 322–337 (2018)
49. Miller, M., Banerjee, A., Christensen, G., Joshi, S., Khaneja, N., Grenander, U., Matejic, L.: Statistical methods in computational anatomy. Statistical Methods in Medical Research **6**, 267–299 (1997)
50. Milnor, J.: Morse Theory. Princeton University Press (1973)
51. Morozov, D.: Homological Illusions of Persistence and Stability. Ph.D. thesis, Duke University (2008)
52. 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)
53. Naiman, D.: volumes for tubular neighborhoods of spherical polyhedra and statistical inference. Ann. Statist. **18**, 685–716 (1990)
54. Nichols, T., Hayasaka, S.: Controlling the familywise error rate in functional neuroimaging: a comparative review. Stat Methods Med. Res. **12**, 419–446 (2003)
55. Otter, N., Porter, M., Tillmann, U., Grindrod, P., Harrington, H.: A roadmap for the computation of persistent homology. EPJ Data Science **6**(1), 17 (2017)
56. Palande, S., Jose, V., Zielinski, B., Anderson, J., Fletcher, P., Wang, B.: Revisiting abnormalities in brain network architecture underlying autism using topology-inspired statistical inference (2017)
57. Petri, G., Expert, P., Turkheimer, F., Carhart-Harris, R., Nutt, D., Hellyer, P., Vaccarino, F.: Homological scaffolds of brain functional networks. Journal of The Royal Society Interface **11**, 20140873 (2014)
58. Pothen, A., Fan, C.: Computing the block triangular form of a sparse matrix. ACM Transactions on Mathematical Software (TOMS) **16**, 324 (1990)
59. Rabin, J., Peyré, G., Delon, J., Bernot, M.: Wasserstein barycenter and its application to texture mixing. In: International Conference on Scale Space and Variational Methods in Computer Vision. pp. 435–446. Springer (2011)
60. 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)
61. Sheehy, D.: Linear-size approximations to the Vietoris–Rips filtration. Discrete & Computational Geometry **49**, 778–796 (2013)

62. de Silva, V., Ghrist, R.: Homological sensor networks. *Notic Amer Math Soc* **54**, 10–17 (2007)
63. Solo, V., Poline, J., Lindquist, M., Simpson, S., Bowman, D., C.M., Cassidy, B.: Connectivity in fMRI: a review and preview. *IEEE Transactions on Medical Imaging* p. in press (2018)
64. Songdechakraiut, T., Chung, M.: Topological learning for brain networks. *Annals of Applied Statistics* **17**, 403–433 (2023)
65. Songdechakraiut, T., Shen, L., Chung, M.: Topological learning and its application to multi-modal brain network integration. *Medical Image Computing and Computer Assisted Intervention (MICCAI)* **12902**, 166–176 (2021)
66. Taylor, J., Worsley, K.: Random fields of multivariate test statistics, with applications to shape analysis. *Annals of Statistics* **36**, 1–27 (2008)
67. Topaz, C., Ziegelmeier, L., Halverson, T.: Topological data analysis of biological aggregation models. *PLoS One* p. e0126383 (2015)
68. Vallender, S.: Calculation of the Wasserstein distance between probability distributions on the line. *Theory of Probability & Its Applications* **18**, 784–786 (1974)
69. Walls, A., Park, Y.J., Tortorici, M., Wall, A., McGuire, A., Veesler, D.: Structure, function, and antigenicity of the SARS-CoV-2 spike glycoprotein. *Cell* **181**, 281–292 (2020)
70. Wang, Y., Ombao, H., Chung, M.: Topological data analysis of single-trial electroencephalographic signals. *Annals of Applied Statistics* **12**, 1506–1534 (2018)
71. Wong, E., Palande, S., Wang, B., Zielinski, B., Anderson, J., Fletcher, P.: Kernel partial least squares regression for relating functional brain network topology to clinical measures of behavior. In: *IEEE International Symposium on Biomedical Imaging (ISBI)*. pp. 1303–1306 (2016)
72. Worsley, K., Marrett, S., Neelin, P., Vandal, A., Friston, K., Evans, A.: A unified statistical approach for determining significant signals in images of cerebral activation. *Human Brain Mapping* **4**, 58–73 (1996)
73. Yang, Z., Wen, J., Davatzikos, C.: Smile-GANs: Semi-supervised clustering via GANs for dissecting brain disease heterogeneity from medical images. *arXiv preprint arXiv*, 2006.15255 (2020)
74. Zomorodian, A.: Computing and Comprehending Topology: Persistence and Hierarchical Morse Complexes. Ph.D. Thesis, University of Illinois, Urbana-Champaign (2001)
75. Zomorodian, A.: *Topology for computing*. Cambridge University Press, Cambridge (2009)
76. Zomorodian, A., Carlsson, G.: Computing persistent homology. *Discrete and Computational Geometry* **33**, 249–274 (2005)