

# Matrix-free spectral relaxations of persistence rank invariants

Matt Piekenbrock

## Abstract

Using a duality result between persistence diagrams and persistence measures, we introduce a family of continuous relaxations of the persistent rank invariant for persistence modules indexed over the real line. Like the rank invariant, the relaxation satisfies inclusion-exclusion, is derived from simplicial boundary operators, and may be evaluated iteratively to recover persistence information. Unlike the rank invariant, the approximation enjoys a number of stability and continuity properties typically reserved for persistence diagrams, such as global Lipschitz continuity and differentiability over the positive semi-definite cone. Fundamental to the family we propose is their characterization as norms of spectral Löwner operators. These operators manifest as combinatorial Laplacians, which encode rich geometric information in their non-harmonic spectra, providing several avenues for geometric data analysis. As exemplary applications, we study its utility in performing common tasks in topological data analysis and machine learning, such as hyper-parameter optimization and shape classification.

## 1 Introduction

Persistent homology [1] is the most widely deployed tool in data analysis and learning applications within the topological data analysis (TDA) community. Persistence-related pipelines typically follow a well-established pattern: given input data set  $X$ , construct a filtration  $(K, f)$  from  $X$  such that useful topological or geometric information may be profitably extracted from its *persistence diagram* [2]. Historically, persistence diagrams were first used as shape descriptors due to their effectiveness in tackling the *homology inference problem* [3], as they demarcate topological features by pairings of simplices defined using homomorphisms between homology groups. The surprising and essential quality of persistence is that this pairing exists, is unique, and is stable under additive perturbations [10]. Indeed, persistence provides the bridge between the theory of homology and the application frontier: domains which operate on noisy measurement data like shape recognition, computer vision, and time series data have all found ways of exploiting the information contained in persistence diagrams.

Though theoretically sound, diagrams suffer from several practical issues: they are sensitive to strong outliers, far from injective, and are difficult both to compute *and* to compare. Indeed, it is difficult to do even basic statistical operations under the standard matching metrics, such as averaging [4]. Towards ameliorating many of these issues and facilitating the use of persistence in learning applications, many practitioners have developed means of vectorizing diagrams via maps to function spaces, such as Hilbert spaces [5]—popular examples include persistence images [6], and persistence landscapes [7], template functions [8].

These diagram-vectorizations have proven useful in learning purposes due to their stability and configurability [], they are fundamentally limited in that they all require diagrams to compute. Along a separate thread of research, Turner et al. propose the *persistent homology transform* (PHT) to handle the injectivity issue: rather than compute one diagram, compute many, each diagram representing a different perspective. The main result of [] is that this collection of diagrams is sufficient to reconstruct the data. This sufficiency result of the PHT is both an exciting and foundational development, as it sparks an entire inverse theory for persistence and in fact motivates the creation of a metric. However, it too relies on the computation of diagrams: in fact, it exacerbates the problem computationally, as computing diagrams in parameterized settings has proven to be non-trivial [].

We seek to shift the paradigm: rather than constructing diagrams first and vectorizing second, we perform both steps simultaneously. Our strategy is natural in that it extracts persistence information using the duality between diagrams and their rank functions []. However, unlike existing approaches, our proposed vectorization involves a spectral characterization that avoids the reduction algorithm [14] entirely, which has proven non-trivial to extend to the parameterized setting []. We are partially motivated by recent relationships established between the persistent Betti numbers and combinatorial Laplacian operators, and by a technical observation that suggests several computational advantages to working with the rank invariant directly (see section 2.1 for details).

## The Relaxation

So as to not leave the reader in suspense, we first outline the proposed relaxation informally. The rest of the paper discusses the theoretical and practical details. At a high level, the relaxation we propose constructs a vector-valued mapping, or *vectorization*, via:

$$(X, \mathcal{R}, \epsilon) \mapsto \mathbb{R}^{|\mathcal{A}|}$$

where  $X$  is the input a data set,  $\epsilon$  a relaxation parameter, and  $\mathcal{R}$  a *sieve* which sifts through a set  $\mathcal{A}$ , summarizing the topological and geometric behavior observed by  $(K, f_\alpha)$  for all  $\alpha \in \mathcal{A}$  in the process. The steps to produce this vectorization are as follows:

1. Let  $K$  denote a fixed simplicial complex constructed from the data set  $X$ . Select a continuously-parameterized family of filtrations:

$$(K, f_\alpha) = \{ f_\alpha : K \rightarrow \mathbb{R} \mid f_\alpha(\tau) \leq f_\alpha(\sigma) \ \forall \tau \subseteq \sigma \in K, \alpha \in \mathcal{A} \}$$

Exemplary choices of  $f_\alpha$  include filtrations geometrically realized from methods that themselves have parameters (“hyper-parameters”), such as density-filtered filtrations [] or Rips filtrations built over dynamic metric spaces [].

2. Choose a *sieve pattern*  $\mathcal{R} \subset \Delta_+$  that is decomposable to a disjoint set of rectangles:

$$\mathcal{R} = R_1 \cup R_2 \cup \dots \cup R_m = \{(i, j) \in \bar{\mathbb{R}}^2 \mid i < j\}$$

This choice is typically application-dependent as it requires a priori knowledge. In section 5 we give evidence random sampling may be sufficient when  $\mathcal{R}$  is unknown.

3. Choose a homology dimension  $p \geq 0$  and relaxation parameter  $\epsilon > 0$  representing how closely the relaxation should model the quantity:

$$\mu_p(\mathcal{R}_\alpha) \stackrel{\epsilon}{\approx} \{ \text{card}(\mathcal{R} \cap \text{dgm}(K, f_\alpha)) \mid \alpha \in \mathcal{A} \} \quad (1.1)$$

Larger choices of  $\epsilon$  yield smoother relaxations that depend more on the geometry of  $K$ , whereas smaller values of  $\epsilon$  more accurately approximate the topology of  $f_\alpha$ .

4. Associate with each unique *corner point*  $(i, j) \in \partial(\mathcal{R})$  an  $\alpha$ -parameterized combinatorial Laplacian operator  $\mathcal{L}_{ij}(K, f_\alpha)$ :

$$\mathcal{L}_{ij} : \mathcal{R} \rightarrow C_1(\mathcal{A}, \mathbb{R}_+)$$

Like step (2), the choice of Laplacian operator depends on the application—we discuss options in section (4.2).

5. For each corner point  $(i, j) \in \mathcal{R}$ , restrict and project  $\mathcal{L}$  onto the Krylov subspace:

$$T = \text{proj}_{\mathcal{K}} \mathcal{L}|_{\mathcal{K}}, \quad \mathcal{K}_n(\mathcal{L}, v) \triangleq \text{span}\{v, \mathcal{L}v, \mathcal{L}^2v, \dots, \mathcal{L}^{n-1}v\}$$

for some initially chosen vector  $v \in \text{span}(\mathbf{1})^\perp$ . As we will show in section 3, the eigenvalues of  $T$  form the basis of the  $\epsilon$ -approximation of (1.1).

6. (Optional) Depending on the amount of memory available, store the largest  $k \geq 0$  eigenvectors of  $T$  at each corner point  $(i, j) \in \partial(\mathcal{R})$  to accelerate the computation of (5) along similar  $\alpha \in \mathcal{A}$  (see section 4.1 for details).

The remaining steps of the relaxation depend on the application in mind. Below, we record two generic objectives and give several example applications that fit within the scope of each objective.

The duality between diagrams and rank functions suggests any application which exploits vectorized persistence information may benefit from our relaxation. In particular, any application which, either implicitly or explicitly, depends on *multiplicity* information may benefit.

#### Generic application #1: Vectorizing persistence information

Compute an  $\epsilon$ -approximation of  $\mu_p(\mathcal{R}_\alpha)$  over some family  $\mathcal{A}$  (in subcubic time per  $\alpha$ )

Such applications include, but are not limited to: characterizing swarm and flocking behavior with Betti curves [], topology-guided image denoising [], detecting bifurcations in dynamical systems [], generating metric invariants for shape classification and metric learning [], and so on.

The fact that our relaxation is differentiable enables applications which seek to perform a learning task with persistence information.

#### Generic application #2: Learning using persistence information

Compute a gradient  $\nabla f_\alpha(\mathcal{R}) \in \mathbb{R}^{|\mathcal{A}|}$  with which to optimize  $\mu_p(\mathcal{R}_\alpha)$  over  $\mathcal{A}$

Exemplary applications include filtration optimization, incorporating topological priors into loss functions, and....

**Contributions:** Our contributions are summarized as follows. We introduce an iterative  $\epsilon$ -approximation method for computing the *the persistent rank invariants*— $\mu_p^*$  and  $\beta_p^*$ —in *essentially*  $O(m)$  memory and  $O(mn)$  time, where  $m, n$  are the number of  $p+1, p$  simplices in the complex, respectively. The approximation is spectral-based and is particularly efficient when executed on perturbed inputs, which we generically refer to as the *parameterized setting*. When the accuracy parameter  $\epsilon$  is made small enough, both invariants are recovered exactly. In deriving the approximation, we obtain families of continuous rank invariants which are Lipschitz continuous, stable under perturbations, and differentiable on the positive semi-definite cone. Unlike existing dynamic persistence algorithms, our approach is quite simple, requiring no complicated data structures or maintenance procedures to implement. The proposed relaxation is also *matrix-free*, requiring only as much memory as is needed to enumerate simplices in the underlying complex  $K$ . Interestingly, our results also imply the existence of an efficient output-sensitive algorithm for computing persistence diagrams (via [8]) that requires as its only input matrix-vector product operator  $x \mapsto \partial x$ , which we consider to be of independent interest.

## Motivation: Duality between ranks and diagrams

In this section, we describe the background theory that motivated the relaxation and contributions listed above, beginning with description of the duality between persistence diagrams and the rank invariant.

The connection between the persistent homology (PH) groups and their corresponding persistent Betti numbers (PBNs) has long been studied from multiple perspectives [6, 7, 10, 24]. A discrete perspective was given by Cohen-Steiner et al. [10], who studied persistence through the lens of *multiplicities*: given a tame function  $f : \mathcal{X} \rightarrow \mathbb{R}$  over a topological space  $\mathcal{X}$ , its homological critical values  $\{a_i\}_{i=1}^n$ , and an interleaved sequence  $\{b_i\}_{i=0}^n$  satisfying  $b_{i-1} < a_i < b_i$  for all  $1 \leq i \leq n$ , the  $p$ -th persistence diagram  $\text{dgm}_p(f)$  over  $f$  defined by [10] is the multiset:

$$\text{dgm}_p(f) := \{ (a_i, a_j) : \mu_p^{i,j} \neq 0 \} \cup \Delta \quad (1.2)$$

where  $\Delta = \{ (x, x) : x \in \mathbb{R}, \mu_p^{x,x} = \infty \}$  denotes the diagonal, counted with infinite multiplicity, and  $\mu_p^{i,j}$  is the *multiplicity function*, defined for all  $0 \leq i < j \leq n+1$  as:

$$\mu_p^{i,j} = (\beta_p^{i,j-1} - \beta_p^{i,j}) - (\beta_p^{i-1,j-1} - \beta_p^{i-1,j}) \quad (1.3)$$

The inclusion-exclusion property conveyed by (1.3) suggests that diagrams completely characterize their PBNs, illuminating an intrinsic connection between diagrams and their Betti numbers. Indeed, the fundamental lemma of persistent homology [14] states that for every pair of indices  $0 \leq k \leq l \leq n+1$ :

$$\beta_p^{k,l} = \sum_{i \leq k} \sum_{j > l} \mu_p^{i,j} \quad (1.4)$$

The direct consequence of (1.4) is that if one is interested in computing any of the PBNs of some space  $\mathcal{X}$ , then it is sufficient to compute  $\text{dgm}_p(\mathcal{X})$  and read them off directly via (1.4). Conversely, combinations of Betti numbers recover portions of  $\text{dgm}_p(\mathcal{X})$  via (1.2) and (1.3).

The duality between diagrams and PBNs becomes more apparent from the measure-theoretic perspective of persistence modules indexed over the real line. By reinterpreting the multiplicity function  $\mu_p^*$  as a kind of integer-valued measure over rectangles in the plane, Chazal [7] demonstrated one may recover the diagram of a persistence module  $M$  over  $\mathbb{R}$  by constructing its corresponding *persistence measure*:

$$\mu_p(R; M) = \text{card} \left( \text{dgm}_p(M) \Big|_R \right) \quad \text{for all rectangles } R \subset \mathbb{R}^2 \quad (1.5)$$

Cerri et al. [6] incorporate this interpretation in studying the stability of PBNs in multidimensional persistence via what they define as *proper cornerpoints*. These are points  $x = (\hat{i}, \hat{j}) \in \Delta_+$  satisfying  $\mu_p(x) > 0$ , where:

$$\mu_p(x) := \min_{\delta > 0} \left( \beta_p^{i+\delta, j-\delta} - \beta_p^{i+\delta, j+\delta} \right) - \left( \beta_p^{i-\delta, j-\delta} - \beta_p^{i-\delta, j+\delta} \right) \quad (1.6)$$

One may compare (1.3) with (1.6). Towards understanding the stability of the rank invariant in the multidimensional persistence settings, Cerri et al. [6] use (1.6) to prove a representation theorem akin to (1.4), showing that PBNs of a scalar-valued filtering function can be completely described by a persistence diagram. One consequence of this theorem is that distances between diagrams induces distances between PBN functions. Since diagrams are stable, the former conveys stability in the latter: small changes in continuous filtering functions imply small changes in the corresponding persistent Betti numbers functions.

The duality exhibited by (1.5) and (1.6) suggests an alternative way of extracting persistence information than the diagram. For example, if one could replace the four PBN quantities in (1.6) with continuously-varying approximations, then the include-exclusion property of the rank invariant suggests the multiplicity  $\mu_p(x)$  could also be approximated in the same way.

## 1.1 Organization

The paper is organized as follows. Section 2 introduces the notation and background theory on which the rest of the paper depends, including an illuminating derivation of a relatively little-known expression of the persistent rank invariant  $\beta_p(K)$  in section 2.1, which is a key technical ingredient for much of the work presented here. Section 3 contains the main results: a family of spectral relaxations of the rank invariants, their properties and interpretations, and their connections to other areas of mathematics. Section 4 compares and contrasts the computational differences between the proposed relaxation and the traditional persistence computation. Section 5 demonstrates some of the prototypical use cases mentioned in section 1. Some technical details, illustrative examples, and pseudocode are relegated to the appendix for readability.

## 2 Notation & Background

An (abstract) *simplicial complex*  $K \subseteq \mathcal{P}(V)$  over a finite *ordered* set  $V = \{v_1, v_2, \dots, v_n\}$  is a collection of simplices  $\{\sigma : \sigma \in \mathcal{P}(V)\}$  such that  $\tau \subseteq \sigma \in K \implies \tau \in K$ . We denote with

$K^p = \{\sigma \in K : \dim(\sigma) = p\}$  the  $p$ -simplices of  $K$  and by  $K^{(p)} = \{\sigma \in K : \dim(\sigma) \leq p\}$  the  $p$ -skeleton of  $K$ . A *filtration*  $K_\bullet = \{K_i\}_{i \in I}$  of a simplicial complexes indexed by a totally ordered set  $I$  is a family of complexes such that  $i < j \in I \implies K_i \subseteq K_j$ .  $K_\bullet$  is called *simplexwise* if  $K_j \setminus K_i = \{\sigma_j\}$  whenever  $j$  is the immediate successor of  $i$  in  $I$ :

$$\emptyset = K_0 \subsetneq K_1 \subsetneq \cdots \subsetneq K_m = K_\bullet, \quad K_i = K_{i-1} \cup \{\sigma_i\} \quad (2.1)$$

Equivalently, we may at times define a filtration  $K_\bullet$  as a pair  $(K, f)$  where  $f : K \rightarrow I$  is a *filter function* over a totally ordered index set  $I$  satisfying  $f(\tau) \leq f(\sigma)$  whenever  $\tau \subseteq \sigma$ , for any  $\tau, \sigma \in K$ . Here, we consider two index sets:  $[n] = \{1, \dots, n\}$  and  $\mathbb{R}$ . Note that any filtration may be converted into a simplexwise filtration via *condensing*, *refining*, and *reindexing* maps [1]. For simplicity, but without loss of generality, we exclusively consider simplexwise filtrations and for brevity refer to them as filtrations.

Given a simplicial complex  $K \subseteq \mathcal{P}(V)$  and a strictly increasing subset  $\sigma = \{v_1, v_2, \dots, v_{p+1}\} \subseteq V$  satisfying  $v_1 < v_2 < \cdots < v_{p+1}$ , an *oriented  $p$ -simplex*  $[\sigma] = [v_1, v_2, \dots, v_{p+1}]$  is defined as:

$$[\sigma] = (-1)^{|\pi|} [v_{\pi(1)}, v_{\pi(2)}, \dots, v_{\pi(p+1)}] \quad (2.2)$$

where  $\pi$  is a permutation on  $[p+1]$  and  $|\pi|$  is the number of inversions of that permutation. The  $p$ -boundary  $\partial_p$  of an oriented  $p$ -simplex  $[\sigma] \in K$  is defined as the alternating sum of its oriented co-dimension 1 faces:

$$\partial_p[\sigma] := \sum_{i=1}^{p+1} (-1)^{i-1} [v_1, \dots, v_{i-1}, v_{i+1}, \dots, v_{p+1}] \quad (2.3)$$

Generalizing beyond simplices, given a field  $\mathbb{F}$ , an *oriented  $p$ -chain* is a formal  $\mathbb{F}$ -linear combination of oriented  $p$ -simplices of  $K$  whose boundary  $\partial_p[c]$  is defined linearly in terms of its constitutive simplices. The collection of chains under addition yields an  $\mathbb{F}$ -vector space  $C_p(K)$ , whose elements  $c \in \partial_p[c']$  are called *boundaries* unless  $\partial_p[c] = 0$ , in which case they are called *cycles*. Together, the collection of  $p$ -boundaries and  $p$ -cycles forms the groups  $B_p(K) = \text{Im } \partial_{p+1}$  and  $Z_p(K) = \text{Ker } \partial_p$ , respectively. Since  $\partial_p \circ \partial_{p+1} = 0$  for all  $p \geq 0$ , the quotient space  $H_p(K) = Z_p(K)/B_p(K)$  is a well-defined group called the  *$p$ -th homology group of  $K$*  with coefficients in  $\mathbb{F}$ . The dimension of the  $p$ -th homology group  $\beta_p(K) = \dim(H_p(K))$  of  $K$  is called the  *$p$ -th Betti number* of  $K$ .

Let  $K_\bullet = \{K_i\}_{i \in [N]}$  denote a filtration of size  $|K_\bullet| = N$ , and let  $\Delta_+^N = \{(i, j) : 0 \leq i \leq j \leq N\}$  denote the set of filtration index pairs. For every such pair  $(i, j) \in \Delta_+^N$ , the sequence of inclusions  $K_i \subsetneq K_{i+1} \subsetneq \cdots \subsetneq K_j$  induce linear transformations  $h_p^{i,j} : H_p(K_i) \rightarrow H_p(K_j)$  at the level of homology:

$$0 = H_p(K_0) \rightarrow \cdots \rightarrow H_p(K_i) \xrightarrow{h_p^{i,j}} H_p(K_j) \rightarrow \cdots \rightarrow H_p(K_N) = H_p(K_\bullet) \quad (2.4)$$

When  $\mathbb{F}$  is a field, this sequence of homology groups uniquely decomposes  $K_\bullet$  into a pairing of simplices  $(\sigma_i, \sigma_j)$  demarcating the evolution of homology classes [24]:  $\sigma_i$  marks the creation of a homology class,  $\sigma_j$  marks its destruction, and the difference  $|i - j|$  records the lifetime

of the class, called its *persistence*. The  $p$ -th persistent homology groups are the images of these transformations and the  $p$ -th persistent Betti numbers are their dimensions:

$$H_p^{i,j} = \begin{cases} H(K_i) & i = j \\ \text{Im } h_p^{i,j} & i < j \end{cases}, \quad \beta_p^{i,j} = \begin{cases} \beta_p(K_i) & i = j \\ \dim(H_p^{i,j}) & i < j \end{cases} \quad (2.5)$$

For a fixed  $p \geq 0$ , the collection of persistent pairs  $(i, j)$  together with unpaired simplices  $(l, \infty)$  form a summary representation  $\text{dgm}_p(K_\bullet)$  called the  $p$ -th *persistence diagram* of  $K_\bullet$ . Conceptually,  $\beta_p^{i,j}$  counts the number of persistent pairs lying inside the box  $(-\infty, i] \times (j, \infty)$  (see Figure 1)—the number of persistent homology groups born at or before  $i$  that died sometime after  $j$ .

## 2.1 Motivating Derivation

To motivate this effort, we begin by deriving a lesser known expression of the persistent Betti number, beginning with a bit of notation. Let  $B_p(K_\bullet) \subseteq Z_p(K_\bullet) \subseteq C_p(K_\bullet)$  denote the  $p$ -th boundary, cycle, and chain groups of a given filtration  $K_\bullet$ , respectively, and let  $\partial_p : C_p(K_\bullet) \rightarrow C_{p-1}(K_\bullet)$  denote the  $p$ -th boundary operator. We use  $\partial$  to denote the  $N \times N$  filtration boundary matrix with respect to an ordered basis  $(\sigma_i)_{1 \leq i \leq N}$  induced by  $K_\bullet$ , given by:

$$\partial[i, j] = \begin{cases} (-1)^{s_{ij}} & \sigma_i \in \partial[\sigma_j], \text{ where } s_{ij} = \text{sgn}([\sigma_i], \partial[\sigma_j]) \\ 0 & \text{otherwise} \end{cases} \quad (2.6)$$

Since  $\partial_p$  is completely characterized by the oriented  $p$  and  $p - 1$  simplices of  $K_\bullet$ , one may obtain a matrix representative of  $\partial_p$  by setting all columns corresponding to simplices of dimension  $q \neq p$  to 0. With a small abuse in notation, we will freely use both  $\partial$  and  $\partial_p$  to refer to both the boundary operators themselves and their matrix representatives.

Given a filtration  $K_\bullet$  of size  $N = |K_\bullet|$ , its  $p$ -th persistent Betti number  $\beta_p^{i,j}$  at index  $(i, j) \in \Delta_+^N$ , where  $\Delta_+^N := \{(i, j) \in [N] \times [N] : i < j\}$ , is defined as the dimension of the quotient group  $H_p^{i,j} = Z_p(K_i)/B_p(K_j)$ :

$$\begin{aligned} \beta_p^{i,j} &= \dim(Z_p(K_i)/B_p(K_j)) \\ &= \dim(Z_p(K_i)/(Z_p(K_i) \cap B_p(K_j))) \\ &= \dim(Z_p(K_i)) - \dim(Z_p(K_i) \cap B_p(K_j)) \end{aligned} \quad (2.7)$$

By definition, the boundary and cycle groups  $B_p(K_j)$  and  $Z_p(K_i)$  are subspaces of the operators  $\partial_p(K_i)$  and  $\partial_{p+1}(K_j)$ , yielding:

$$\beta_p^{i,j} = \dim(\text{Ker}(\partial_p(K_i))) - \dim(\text{Ker}(\partial_p(K_i)) \cap \text{Im}(\partial_{p+1}(K_j))) \quad (2.8)$$

Now, consider computing  $\beta_p^{i,j}$  via (2.8) from matrix representatives  $\partial_p \in \mathbb{F}^{n \times m}$ . Since the nullity of an operator may be reduced to a rank computation, the complexity of first term may be reduced to the complexity of computing the rank of a (sparse)  $n \times m$  matrix. In contrast, the second term—the persistence part—typically requires finding a basis in intersection of the two subspaces via either column reductions or projection-based techniques. In general, direct methods that accomplish this require  $\Omega(N^3)$  time and  $\Omega(N^2)$  memory.

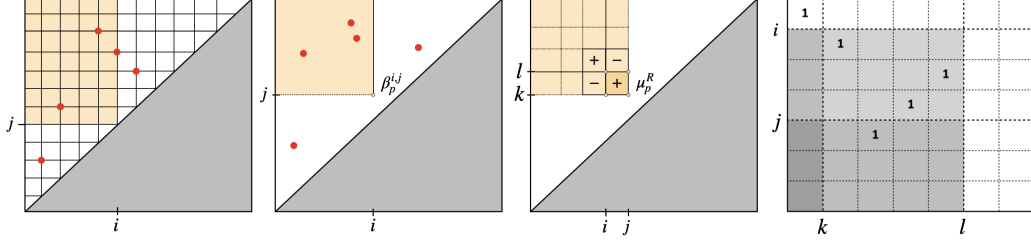


Figure 1: From left to right,  $\beta_p^{i,j}$  counts the number of points (3) in upper left-corner of  $\text{dgm}_p(K_\bullet)$ , where  $i, j \in \Delta_+^N$ ; the same  $\beta_p^{i,j}$  with  $i, j \in \Delta_+$ ; the additivity of  $\beta_p^*$  implies  $\mu_p^R$  over a box  $R = [i, j] \times [k, l]$  is given as the sum of four PBNs; generalization of 2.9—in this case  $\mu_p^R = 4 - 1 - 0 + 0 = 3$  counts pivot entries in the reduced matrix  $R = \partial V$ .

To illustrate an alternative approach, we will require a key property of persistence. The structure theorem from [24] shows that 1-parameter persistence modules can be decomposed in an *essentially unique* way into indecomposables. One consequence of this result is the Pairing Uniqueness Lemma [15], which asserts that if  $R = \partial V$  decomposes the boundary matrix  $\partial$  to a *reduced* matrix  $R \in \mathbb{R}^{m \times n}$  using left-to-right column operations, then:

$$R[i, j] \neq 0 \Leftrightarrow \text{rank}(R^{i,j}) - \text{rank}(R^{i+1,j}) + \text{rank}(R^{i+1,j-1}) - \text{rank}(R^{i,j-1}) \neq 0 \quad (2.9)$$

where  $R^{i,j}$  denotes the lower-left submatrix defined by the first  $j$  columns and the last  $m-i+1$  rows (rows  $i$  through  $m$ , inclusive). In other words, the existence of non-zero “pivot” entries in  $R$  may be inferred entirely from the ranks of certain submatrices of  $R$ . As we will use this fact frequently in this paper, we record it formally with a lemma.

**Lemma 1.** *Given filtration  $K_\bullet$  of size  $N = |K|$ , let  $R = \partial V$  denote the decomposition of the filtered boundary matrix  $\partial \in \mathbb{R}^{N \times N}$  given in equation (2.6). Then, for any pair  $(i, j)$  satisfying  $1 \leq i < j \leq N$ , we have:*

$$\text{rank}(R^{i,j}) = \text{rank}(\partial^{i,j}) \quad (2.10)$$

*Equivalently, all lower-left submatrices of  $\partial$  have the same rank as their corresponding submatrices in  $R$ .*

An explicit proof of this can be found in [12], though it was also noted in passing by Edelsbrunner [15]. It can be shown by combining the Pairing Uniqueness Lemma with the fact that  $R$  is obtained from  $\partial$  via left-to-right column operations, which preserve the ranks of every such submatrix. Lemma 1 is remarkable in that although  $R$  is not unique, its non-zero pivots are, and these pivots *define* the persistence diagram. This seems like a minor observation at first, however it is far more general, as recently noted by Bauer et al. [3]:

**Proposition 1** (Bauer et al. [3]). *Any persistence algorithm which preserves the ranks of the submatrices  $\partial^{i,j}(K_\bullet)$  for all  $i, j \in [N]$  is a valid persistence algorithm.*

A lesser-known fact that exploits Lemma 1—also pointed out in [12]—is that (2.10) enables the PBN to be written as a sum of ranks of submatrices of  $\partial_p$  and  $\partial_{p+1}$ .



**Proposition 2** (Dey & Wang [12]). *Given a fixed  $p \geq 0$ , a filtration  $K_\bullet$  of size  $N = |K_\bullet|$ , and any pair  $(i, j) \in \Delta_+^N$ , the persistent Betti number  $\beta_p^{i,j}(K_\bullet)$  at  $(i, j)$  is given by:*

$$\beta_p^{i,j}(K_\bullet) = |K_i^p| - \text{rank}(\partial_p^{1,i}) - \text{rank}(\partial_{p+1}^{1,j}) + \text{rank}(\partial_{p+1}^{i+1,j}) \quad (2.11)$$

For completeness, we give our own detailed proof of Proposition 2 in the appendix. By combining Proposition 2 with (1.3), we recover a submatrix-rank-based  $p$ -th multiplicity function  $\mu_p^R(\cdot)$ , which to the authors knowledge was first pointed out by Chen & Kerber [8]:

**Proposition 3** (Chen & Kerber [8]). *Given a fixed  $p \geq 0$ , a filtration  $K_\bullet = \{K_i\}_{i \in [N]}$  of size  $N = |K|$ , and a  $R = [i, j] \times [k, l]$  whose indices  $(i, j, k, l)$  satisfy  $0 \leq i < j \leq k < l \leq N$ , the  $p$ -th multiplicity  $\mu_p^R$  of  $K_\bullet$  is given by:*

$$\mu_p^R(K_\bullet) = \text{rank}(\partial_{p+1}^{j+1,k}) - \text{rank}(\partial_{p+1}^{i+1,k}) - \text{rank}(\partial_{p+1}^{j+1,l}) + \text{rank}(\partial_{p+1}^{i+1,l}) \quad (2.12)$$

For more geometric intuition of this proposition, see Figure 1. Computationally, the relevance of Propositions 2 and 3 is they imply the complexity of obtaining either  $\beta_p^{i,j}(K_\bullet)$  or  $\mu_p^R(K_\bullet)$  may be reduced to the complexity of computing the rank of a set of submatrices of  $\partial$ , a fact that actually motivated the rank-based persistence algorithm from Chen et al [8].

Compared to the classical reduction methods suggested in [14, 24], the primary advantage of the rank-based expressions from (2.11)-(2.12) is that they only depend on the ranks of *unfactored* boundary (sub)matrices. Thus, if we restrict the field  $\mathbb{F}$  to  $\mathbb{R}$ , we may use iterative Krylov and subspace acceleration methods for their computation instead of reduction-like procedures. Moreover, the many known algebraic properties of the rank function, such as permutation invariance, suggest there may be means of simplifying these rank-based expressions even further. We dedicate the rest of the paper to exploring these questions.

**Remark 1.** The notation used thus far employed integer indices  $(i, j) \in \Delta_+^N$  to describe persistent quantities over a filtration  $K_\bullet = (K, f)$  of size  $|K| = N$ , which is equivalent to indexing  $K$  with a filter function  $f : K \rightarrow I$  defined on the index set  $I = [N]$ . It is more common in practice to define persistence of a persistent-pair  $(\sigma_i, \sigma_j) \in \text{dgm}(K_\bullet)$  via  $f(\sigma_j) - f(\sigma_i)$ , rather than as  $j - i$ , especially when  $f$  is geometric in nature. In this setting, each pair  $(\sigma_i, \sigma_j) \in \text{dgm}(K_\bullet)$  is typically represented as the point  $(\hat{i}, \hat{j})$  where  $\hat{i} = f(\sigma_i)$  and  $\hat{j} = f(\sigma_j)$ . Again exploiting the inclusion-exclusion property known for real-valued persistence modules [7], we from now on will use the notation  $(i, j)$  to denote pairs  $(i, j) \in \Delta_+$  from the upper-half plane  $\Delta_+ = \{(x, y) \in \mathbb{R}^2 : y > x\}$ .

### 3 Spectral relaxation and its implications

The lesser known rank invariant expressions from (2.11) and (2.12) can profitably benefit from the many of invariances of the rank function itself. One such property is permutation invariance: given any  $A \in \mathbb{R}^{n \times n}$ , it is easy to see that  $\text{rank}(A) = \text{rank}(P^T A P)$  for any permutation matrix  $P$ . To see the utility of this fact, consider a filtration  $(K_\bullet, f)$  equipped with a *parameterized* filter function  $f : \mathcal{A} \times K \rightarrow \mathbb{R}$  with respect to some set  $\mathcal{A}$ . In typical dynamic persistence settings (e.g. [11]), the filtration boundary matrix  $\partial(K_\bullet)$  must maintain a simplexwise total order to preserve the filtered poset structure of  $K_\bullet$ . However,

the permutation invariance of the rank function suggests  $\partial$  need not maintain this order—as long as each constitutive term in e.g. (2.12) has the same non-zero pattern as its filtration-ordered counterpart, their ranks will be identical.

To convey the computational utility permutation invariance enables in parameterized settings, consider the following reformulation of the boundary operator  $\partial_p$ .

**Definition 1** (Parameterized  $\partial_p$ ). Let  $(K, f_\alpha)$  denote a simplicial complex of size  $|K| = N$  filtered by a parameterized filter  $f_\alpha : K \times \mathcal{A} \rightarrow \mathbb{R}$ . Fix an arbitrary linear extension  $(K, \preceq^*)$  of the face poset. Define the  $\mathcal{A}$ -parameterized boundary matrix  $\partial_p(\alpha)$  as the  $N \times N$  matrix ordered by  $\preceq^*$  for all  $\alpha \in \mathcal{A}$  whose entries  $(k, l)$  satisfy:

$$\partial_p(\alpha)[k, l] = \begin{cases} s_{kl} \cdot f_\alpha(\sigma_k) \cdot f_\alpha(\sigma_l) & \text{if } \sigma_k \in \partial_p(\sigma_l) \\ 0 & \text{otherwise} \end{cases} \quad (3.1)$$

where  $s_{kl} = \text{sgn}([\sigma_k], \partial[\sigma_l])$  is the sign of the oriented face  $[\sigma_k]$  in  $\partial[\sigma_l]$ .

Observe that (1) the non-zero entries of the boundary operator from definition 3.1 vary continuously in  $f_\alpha$  and (2)  $\partial_p(\alpha)$  can be decoupled into a product of diagonal matrices  $D_*(f_\alpha)$  via:

$$\partial_p(\alpha) = D_p(f_\alpha) \circ \partial_p \circ D_{p+1}(f_\alpha) \quad (3.2)$$

where the entries of  $D_p(f_\alpha)$  depend on  $f_\alpha(\tau)$  for  $\tau \in K^p$ . We refer to the fixed inner matrix  $\partial_p \in \{-1, 0, 1\}^{n \times m}$  as the *sign pattern matrix*. Now, observe that exploiting permutation invariance and the decoupling from 3.2 we have:

$$\text{rank}(\partial_p^{i,j}(K_\bullet)) = \text{rank}(\partial_p^{i,j}(\alpha)) \triangleq \text{rank}(D_p(\bar{S}_i \circ f_\alpha) \circ \partial_p(K) \circ D_{p+1}(S_j \circ f_\alpha)) \quad (3.3)$$

where  $\bar{S}, S : \mathbb{R} \rightarrow \{0, 1\}$  are up and down *step functions*, respectively, given by:

$$\bar{S}_i(x) = \begin{cases} 1 & \text{if } x > i \\ 0 & \text{otherwise} \end{cases}, \quad S_j(x) = \begin{cases} 1 & \text{if } x \leq j \\ 0 & \text{otherwise} \end{cases}$$

Note in (3.3) we write  $K_\bullet$  and  $K$  to make the distinction that the boundary operator of the former is explicitly filtered in the traditional simplexwise manner, whereas the latter is ordered arbitrarily. Since these operators have the same rank, we may replace the constitutive boundary terms in (2.12) and (2.11) appropriately, thus obtaining expressions for the rank invariants using parameterized boundary operators of the form given in definition 1.

We summarize the observations above with a proposition. To simplify the notation, we write  $A^x = A^{*,x}$  to denote the submatrix including all rows of  $A$  and all columns of  $A$  up to  $x$ , and we write  $q = p + 1$ . Without loss in generality, we also assume the orientation of the simplices induced by  $(K, \preceq^*)$  is inherited from the order on the vertex set  $V$ .

**Proposition 4.** Let  $\delta > 0$  denote the number from (1.6) satisfying  $i + \delta < j - \delta$ . Given  $(K, f_\alpha)$  and any rectangle  $R = [i, j] \times [k, l] \subset \Delta_+$  satisfying  $i < j \leq k < l$ , the  $\mathcal{A}$ -parameterized invariants  $\beta_p^{i,j} : \mathcal{A} \times K \rightarrow \mathbb{N}$  and  $\mu_p^R : \mathcal{A} \times K \rightarrow \mathbb{N}$  functions given by:

$$\beta_p^{i,j}(\alpha) = |K_i^p(\alpha)| - \text{rank}(\partial_p^i(\alpha)) - \text{rank}(\partial_q^j(\alpha)) + \text{rank}(\partial_q^{i+\delta,j}(\alpha)) \quad (3.4)$$

$$\mu_p^R(\alpha) = \text{rank}(\partial_q^{j+\delta,k}(\alpha)) - \text{rank}(\partial_q^{i+\delta,k}(\alpha)) - \text{rank}(\partial_q^{j+\delta,l}(\alpha)) + \text{rank}(\partial_q^{i+\delta,l}(\alpha)) \quad (3.5)$$

yield the correct quantities  $\mu_p(R) = \text{card}(\text{dgm}_p(f_\alpha)|_R)$  and  $\beta_p^{i,j} = \dim(H_p^{i,j}(K))$  from (1.5) and (2.7), respectively, for all  $\alpha \in \mathcal{A}$ .

We give proofs the equivalence between these two expressions in the appendix.

### 3.1 Spectral rank relaxation

Extending from the ideas outlined in the previous section, we now turn towards ways of substituting the discontinuous rank function with a continuous approximation, beginning with a spectral characterization. Given a matrix  $X \in \mathbb{R}^{n \times m}$  and its singular value decomposition (SVD)  $X = U\Sigma V^T$ , define the *rank* of  $X$  as the composition:

$$\text{rank}(X) = \sum_{i=1}^n \text{sgn}_+(\sigma_i(X)), \quad \text{sgn}_+(x) = \begin{cases} 1 & \text{if } x > 0 \\ 0 & \text{otherwise} \end{cases} \quad (3.6)$$

where  $\Sigma = \text{diag}(\{\sigma_1, \sigma_2, \dots, \sigma_n\})$  are the singular values of  $X$  and  $\text{sgn}_+ : \mathbb{R} \rightarrow \{0, 1\}$  is the one-sided sign function. Like Betti numbers, the unstable nature of the rank function is due to its inherently combinatorial properties. In particular, the discontinuity in (3.6) manifests due to the one-sided sign function.

We follow the framework presented by Bi et al. [5] for constructing a smooth rank function approximations via relaxations of the  $\text{sgn}_+$  function. Building off the seminal work of Chen & Mangasarian [20], their approach is to approximate the  $\text{sgn}_+$  function in (3.6) by integrating smoothed variations  $\hat{\delta}$  of the Dirac delta function  $\delta$ :

$$(\forall z \geq 0, \epsilon > 0) \quad \phi(x, \epsilon) := \int_{-\infty}^x \hat{\delta}(z, \epsilon) dz, \quad \hat{\delta}(z, \epsilon) = \nu(1/\epsilon) \cdot p(z \nu(1/\epsilon)) \quad (3.7)$$

where  $p : \mathbb{R}_+ \rightarrow \mathbb{R}_+$  is continuous density function and  $\nu : \mathbb{R}_+ \rightarrow \mathbb{R}_+$  is continuous increasing function satisfying  $\nu(0) = 0$  and  $\nu(\epsilon) > 0$ . In contrast to the  $\text{sgn}_+$  function, if  $p$  is continuous on  $\mathbb{R}_+$  then  $\phi(\cdot, \epsilon)$  is continuously differentiable on  $\mathbb{R}_+$ , and if  $p$  is bounded above on  $\mathbb{R}_+$ , then  $\phi(\cdot, \epsilon)$  is globally Lipschitz continuous on  $\mathbb{R}_+$ . Moreover, note that  $\phi(0, \epsilon) = 0$  and:

$$(\forall x \geq 0) \quad \lim_{\epsilon \rightarrow 0^+} \phi(x, \epsilon) = \text{sgn}_+(x) \quad (3.8)$$

for any choice of  $(p, \nu)$  satisfying the assumptions above. Varying the relaxation parameter  $\epsilon > 0$  in (3.7) yields an  $\epsilon$ -parameterized family of continuous  $\text{sgn}_+$  relaxations  $\phi : \mathbb{R}_+ \times \mathbb{R}_{++} \rightarrow \mathbb{R}_+$ , where  $\epsilon > 0$  controls the accuracy of the relaxation.

Many of the properties of (3.7) extend naturally to the rank function when the sign function is substituted appropriately in (3.6). In particular, if  $X \in \mathbb{R}^{n \times m}$  admits a singular value decomposition  $X = U\Sigma V^T$  and is paired with a scalar-valued  $\phi$  continuously differentiable at every  $\sigma \in \Sigma$ , then for any  $\epsilon > 0$  the corresponding *Löwner operator*  $\Phi_\epsilon$  given by:

$$\begin{aligned} \Phi_\epsilon(X) &= U \text{diag}(\phi(\sigma_1, \epsilon), \phi(\sigma_2, \epsilon), \dots, \phi(\sigma_n, \epsilon)) V^T \\ &= \sum_{i=1}^n \phi(\sigma_i, \epsilon) u u^T \end{aligned} \quad (3.9)$$

is a continuously differentiable operator in  $\mathbb{R}^{n \times m}$ . In fact, if  $\phi$  is twice-differentiable at each  $\sigma_i(X)$  for all  $i = 1, \dots, n$ , then (3.9) is also twice continuously differentiable at  $X$  [13]. Before summarizing the additional properties this operator has and its relationship with the rank function, we give a definition:

**Definition 2** (Spectral Rank Approximation). Given  $X \in \mathbb{R}^{n \times m}$  with singular value decomposition  $X = U\Sigma V^T$  where  $\Sigma = \text{diag}(\sigma_1, \sigma_2, \dots, \sigma_n)$  and a fixed  $\epsilon > 0$ , any choice of  $\phi : \mathbb{R}_+ \times \mathbb{R}_{++}$  satisfying (3.7) defines a *spectral rank approximation*  $\|\Phi_\epsilon(X)\|_*$  of  $X$  via:

$$\|\Phi_\epsilon(X)\|_* = \sum_{i=1}^n \phi(\sigma_i, \epsilon) \quad (3.10)$$

Apart from serving as a smooth approximation of the rank function, this quantity turns out to have a variety of attractive properties related to monotonicity and differentiability. A few such properties are recorded below.

**Proposition 5** (Bi et al. [5]). *The operator  $\Phi_\epsilon : \mathbb{R}^{n \times m} \rightarrow \mathbb{R}^{n \times m}$  defined by (3.9) satisfies:*

1. *For any  $\epsilon \leq \epsilon'$ ,  $\|\Phi_\epsilon(X)\|_* \geq \|\Phi_{\epsilon'}(X)\|_*$  for all  $X \in \mathbb{R}^{n \times m}$ .*
2. *For any given  $X \in \mathbb{R}^{n \times m}$  with rank  $r = \text{rank}(X)$ , if  $\epsilon$  satisfies  $0 < \epsilon \leq \sigma_r/r$ , then:*

$$0 \leq r - \|\Phi_\epsilon(X)\|_* \leq c$$

*where  $c$  is a positive constant that depends only on  $r$ ,  $\nu$ , and  $p$ .*

3.  *$\|\Phi_\epsilon(X)\|_*$  is globally Lipschitz continuous and semismooth<sup>1</sup> on  $\mathbb{R}^{n \times m}$ .*

Note that  $\|\Phi_\epsilon(X)\|_*$  is not necessarily differentiable on  $\mathbb{R}^{n \times m}$  due to Proposition 2.2(e) in [5], but it is differentiable on the positive semi-definite cone  $\mathbb{S}_+^n$ . As we will discuss in section 3.2, this is sufficient for our purposes here.

To adapt these relaxations to the counting functions defined in equations (1.4) and (2.12), we need to modify the expression of the boundary chains (2.3) to vary continuously in  $\mathcal{A}$ . Fortunately, this requires a simple augmentation to the step function from the one in definition 1. In particular, we swap out the step functions  $S : \mathbb{R} \rightarrow \{0, 1\}$  from (3.1) with *smoothstep* functions  $\mathcal{S} : \mathbb{R} \rightarrow [0, 1]$ . These are clamped “S-curve” functions which interpolate the discontinuous step portion of  $S$  along a fixed interval  $(a, a + \omega)$ , which are given by:

$$\mathcal{S}_a^\omega(x) = \begin{cases} 0 & x \leq a \\ S_n(\omega^{-1}((a + \omega) - x)) & a < x < a + \omega \\ 1 & a + \omega \leq x \end{cases} \quad (3.11)$$

where  $S_n : [0, 1] \rightarrow [0, 1]$  is a generic  $n$ -th order polynomial whose coefficients are fixed such that  $S_n(0) = 0$  and  $S_n(1) = 1$ . Due to their efficiency in evaluation and simple configuration, such functions have found applications computer graphics and machine learning

---

<sup>1</sup>The notion of semismoothness here refers to the existence certain directional derivatives in the limit as  $\epsilon \rightarrow 0^+$ , see [5, 4] for more details.

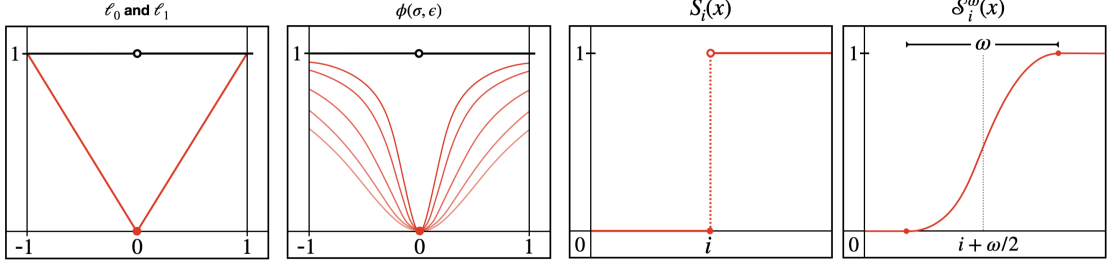


Figure 2: From left to right—the  $\ell_1$  norm (red) forms a convex envelope over the  $\ell_0$  (black) pseudo-norm on the interval  $[-1, 1]$ ;  $\tilde{\phi}(\cdot, \epsilon)$  at various values of  $\epsilon$ , with  $p(x) = 2x(x^2 + 1)^{-2}$  and  $\nu(\epsilon) = \sqrt{\epsilon}$  (red) and at  $\epsilon = 0$  (black); the step function  $S_i(x)$  from definition 1; the smoothstep relaxation  $\mathcal{S}_i^\omega$  from (3.11).

applications []. Our motivation to use them here is based on the observations shown in Figure 2: by substituting  $\mathcal{S}_n^\omega$  for the step functions in (3.1) and adding a small translation, we may replace the step functions in the parameterized boundary matrix with entries varies continuously in  $\mathcal{A}$  and retain the correct sign. By composing with (2.12) and substituting  $\text{rank}(\cdot)$  with  $\Phi(\cdot, \epsilon)$  for some  $\epsilon > 0$ , we obtain  $\epsilon$ -approximate, continuously-varying multiplicity function.

**Definition 3** (Spectral Rank). Let  $K$  denote a simplicial complex and  $f : K \times \mathcal{A} \rightarrow \mathbb{R}$  a parameterized family of functions which vary continuously in  $\mathcal{A}$ . For some fixed  $R = [i, j] \times [k, l] \subset \Delta_+$  and parameters  $(p, \epsilon, \omega)$  satisfying  $p \geq 0$ ,  $\epsilon > 0$ , and  $\omega > 0$ , respectively, define the  $\mathcal{A}$ -parameterized *spectral persistent betti number* and *spectral multiplicity* as:

$$\hat{\beta}_p^{i,j}(\alpha) = \|\Phi_\epsilon^\alpha(\hat{I}_p^i)\|_* - \|\Phi_\epsilon^\alpha(\hat{\partial}_p^{1,i})\|_* - \|\Phi_\epsilon^\alpha(\hat{\partial}_{p+1}^{1,j})\|_* + \|\Phi_\epsilon^\alpha(\hat{\partial}_{p+1}^{i+\delta,j})\|_* \quad (3.12)$$

$$\hat{\mu}_{p,\epsilon}^R(\alpha) = \|\Phi_\epsilon^\alpha(\hat{\partial}_{p+1}^{j+\delta,k})\|_* - \|\Phi_\epsilon^\alpha(\hat{\partial}_{p+1}^{i+\delta,k})\|_* - \|\Phi_\epsilon^\alpha(\hat{\partial}_{p+1}^{j+\delta,l})\|_* + \|\Phi_\epsilon^\alpha(\hat{\partial}_{p+1}^{i+\delta,l})\|_* \quad (3.13)$$

where  $\Phi_\epsilon^\alpha(\partial) = (\Phi_\epsilon \circ \partial)(\alpha)$  for any  $\alpha \in \mathcal{A}$ ,  $\hat{I}_p^i = \text{diag}(\{\mathcal{S}_i^\omega(f(\tau_j))\}_{j=1}^n)$ , and  $\hat{\partial}_p$  is  $\mathcal{A}$ -parameterized  $p$ -th boundary matrix from definition 1 with step functions  $S_*$  replaced by the smoothstep  $\mathcal{S}_*^\omega$  from (3.11).

Since the sum  $f + g$  of two Lipschitz functions  $f$  and  $g$  is also Lipschitz, it is easy to verify both  $\hat{\mu}_p^R$  and  $\hat{\beta}_p^{i,j}$  are Lipschitz continuous functions in  $\alpha$  whenever  $f$  is Lipschitz by combining the smoothness of  $\mathcal{S}_*^\omega$  with the global Lipschitz continuity of (3.10).

## Stability

One disadvantage of rank functions restricted to subsets of the real-plane is that they are integer-valued and unstable. One may easily construct examples of  $\mathcal{A}$ -parameterized filtrations  $(K, f_\alpha)$  where  $\|\mu_p^R(\alpha) - \mu_p^R(\alpha + \delta)\| \sim O(|K_p|)$  for some arbitrarily small  $\delta > 0$ , as there may be up to  $O(|K_p|)$  points in  $\text{dgm}_p(K)$ . In what follows, we investigate how to exploit the smoothness of  $f_\alpha$  to stabilize the constitutive terms in  $\hat{\mu}_p^*$  and  $\hat{\beta}_p^*$ .

A common way of quantifying the sensitivity of the spectrum of a given linear operator  $M$  is through its condition number. For  $M = XX^T$  a given positive definite matrix, its

condition number  $\kappa(M)$  is defined as:

$$\kappa(M) = \|M^{-1}\| \|M\| = |\lambda_1(M)| / |\lambda_n(M)| \quad (3.14)$$

The condition number  $\kappa(M)$  directly measures of how sensitive the spectrum of  $M$  is too perturbations in its entries. In particular, if  $E \in \mathbb{R}^{n \times n}$  represents a small perturbation of  $M \in \mathbb{R}^{n \times n}$ , then:

$$\frac{\|(M + E)^{-1} - M^{-1}\|}{\|M^{-1}\|} \leq \kappa(M) \frac{\|E\|}{\|M\|} \quad (3.15)$$

Thus, the effect of adding  $\epsilon I_n$  to a given matrix can be interpreted as a means of reducing  $\kappa$  arbitrarily—at the expense of accuracy—to stabilize the pseudo-inverse. For operators  $\Phi_\epsilon(\cdot)$  in the form above, we can quantify this stabilization using perturbation analysis.

**Proposition 6.** *TODO stability statement*

**Regularization:** Another approach to stabilizing the rank function often employed in the context of large-scale sparse inverse problems is to regularize the solution. For example, the classical least-squares problem solves the [possibly ill-posed] linear system  $Ax = b$  via the minimization:

$$x^* = \arg \min_{x \in \mathbb{R}^n} \|Ax - b\|^2 \quad (3.16)$$

For a variety of reasons, such as giving preference to solutions  $x^*$  with small norm, (3.16) is often substituted with an explicit *Tikhonov regularization* (TR) for some  $\epsilon > 0$ :

$$x_\epsilon^* = \arg \min_{x \in \mathbb{R}^n} \|Ax - b\|^2 + \epsilon \|x\|^2 \quad (3.17)$$

The solution to (3.17) is given by the regularized form of the pseudo-inverse given below:

$$x_\epsilon^* = (A^T A + \epsilon I)^{-1} A^T b \quad (3.18)$$

To illustrate the relevance of TR with our proposed approximation, consider the parameterization<sup>2</sup> of  $\phi$  obtained by setting  $\nu(\epsilon) = \sqrt{\epsilon}$  and  $p(x) = 2x(x^2 + 1)^{-2}$ . By (3.7), one obtains:

$$\phi(x, \epsilon) = \int_{-\infty}^x \hat{\delta}(z, \epsilon) dz = \frac{x^2}{x^2 + \epsilon} \quad (3.19)$$

Substituting  $\text{sgn}_+ \mapsto \phi$  and composing with the singular value function (3.10), the corresponding spectral rank approximation reduces to:

$$\|\Phi_\epsilon(X)\|_* = \sum_{i=1}^n \frac{\sigma_i(X)^2}{\sigma_i(X)^2 + \epsilon} = \text{Tr} [(X^T X + \epsilon I_n)^{-1} X^T X] \quad (3.20)$$

The connection between the TR right hand side of (3.20) is now clear: obtaining  $\|\Phi_\epsilon(L)\|_*$  is equivalent to finding the eigenvalues of solving  $n$  TR least-squares problems of the form (3.17) where  $X$  is substituted with the appropriate boundary/coboundary operator  $\partial_p/\partial_p^T$  and  $b$  is substituted with an  $\alpha$ -parameterized  $p$ -(co)chain.

---

<sup>2</sup>Similar regularization interpretations may be derived for other choices of  $\nu(\epsilon)$  and  $p(x)$ .

**Remark 2.** One interpretation of the relaxation parameter  $\epsilon$  is as a bias term that preferences the (pseudo)-inverse towards smaller singular values. Larger values of  $\epsilon$  smooth out  $\|\Phi_\epsilon(\cdot)\|_*$  by making the pseudo-inverse less sensitive to perturbations, whereas smaller values of  $\epsilon$  lead to a more faithful approximation of the rank. This can be seen directly by (3.18) as well, as increasing  $\epsilon$  lowers the condition number of  $A^T A + \epsilon I$  monotonically, signaling a tradeoff in stability at the expense of accuracy.

### 3.2 Combinatorial Laplacians

One shortcoming of the stability result shown above is that the Lipschitz constant depends explicitly on the magnitude of  $f_\alpha$ , which in-turn affects the stability of the proposed relaxation if  $f_\alpha$  is scaled poorly, i.e.  $\lambda \cdot f_\alpha$ . Moreover, the continuous differentiability of  $\|\Phi_\epsilon(\cdot)\|_*$  only extends to linear operators defined over the positive semi-definite cone, and the explicit form of the differential  $\partial\|\Phi_\epsilon(\cdot)\|_*$  depends on the domain of its corresponding spectral function  $\phi(\cdot)$  being bounded to some interval  $(a, b)$  [4]. On the positive side, expressions like (2.11) and (2.12) may benefit from the identities  $\text{rank}(A) = \text{rank}(AA^T) = \text{rank}(A^T A)$  and the fact that the square roots of the non-zero eigenvalues of either  $AA^T$  or  $A^T A$  yield the singular values of  $A$ . Indeed, observe expressing  $\mu_0$  and  $\beta_0$  solely in terms of boundary operators allows use to substitute the constitutive terms of with  $L = \partial_1 \partial_1^T$ , the well studied *graph Laplacian*. More generally, we may consider the study of spectra of *combinatorial Laplacians*—generalizations of the graph Laplacian—as the study of singular values of boundary operators. Since these are central quantities studied thus far, we elaborate on this connection in what follows.

Given a simple undirected graph  $G = (V, E)$ , let  $A \in \{0, 1\}^{n \times n}$  denote its binary adjacency matrix satisfying  $A[i, j] = 1$  if the vertices  $v_i, v_j \in V$  are path-connected in  $G$ , and 0 otherwise. Moreover, denote with  $D$  the diagonal degree matrix  $D = \text{diag}(\{\deg(v_i)\})$ , where  $\deg(v_i) = \sum_{j \neq i} A[i, j]$ . The *graph Laplacian*  $L$  given in terms of adjacency, incidence, and element-wise relations, is defined as:

$$L = D - A = \partial_1 \circ \partial_1^T, \quad L[i, j] = \begin{cases} \deg(v_i) & \text{if } i = j \\ -1 & \text{if } i \sim j \\ 0 & \text{if } i \not\sim j \end{cases} \quad (3.21)$$

where we use the notation  $i \sim j$  to denote the path-connected relation in  $G$  between pairs of vertices. Its well known that  $L$  is symmetric, positive semi-definite, and has a combinatorial structure that captures the connectivity structure of  $G$ . A natural generalization for simplicial complexes is the  $p$ -th *combinatorial Laplacian*  $\Delta_p$ , given by:

$$\Delta_p(K) = \underbrace{\partial_{p+1} \circ \partial_{p+1}^T}_{L_p^{\text{up}}} + \underbrace{\partial_p^T \circ \partial_p}_{L_p^{\text{dn}}} \quad (3.22)$$

As all three operators  $\Delta_p$ ,  $L_p^{\text{up}}$ , and  $L_p^{\text{dn}}$  are self-adjoint, non-negative, and compact, their eigenvalues are real and non-negative. Note when  $p = 0$ , we have  $\Delta_0(K) = \partial_1 \partial_1^T = L$  since  $L_0^{\text{dn}} = 0$ . Among the first to study these combinatorial operators was Eckmann [], who proved a discrete version of the Hodge Theorem, which states:

$$\tilde{H}_p(K; \mathbb{R}) \cong \ker(\Delta_p(K)), \quad \beta_p = \text{nullity}(\Delta_p(K)) \quad (3.23)$$

Horak and Jost developed a study of the spectra of combinatorial Laplacians that unified the perspective many of the Laplace operators on simplicial complexes []. Among other results, they showed that the *non-zero* parts of the spectra  $\Lambda_+$  satisfy:

$$\Lambda_+(\Delta_p(K)) = \Lambda_+(L_p^{\text{up}}(K)) \cup \Lambda_+(L_p^{\text{dn}}(K)) \quad (3.24)$$

Moreover, it may be shown that the non-zero eigensets  $\Lambda_+(L_p^{\text{up}}(K))$  and  $\Lambda_+(L_{p+1}^{\text{dn}}(K))$  are identical—thus, it suffices to consider only one of them. Many subsequent publications studying higher order combinatorial Laplacians have subsequently found applications to areas such as graph optimization [], network circuit theory [], and sparsification problems.

Both the range of the spectrum of the combinatorial Laplacian and its structural form depend on choice of the inner product on the coboundary vector spaces. Recall that the cochain groups  $C^p(K, \mathbb{R}) := \text{Hom}(C_p(K, \mathbb{R}), \mathbb{R})$  are defined as the duals of the chain groups  $C_p(K, \mathbb{R})$ . As with  $C_p(K, \mathbb{R})$ , a basis for  $C^p(K, \mathbb{R})$  is given by the set of functions:

$$\{ \chi([\sigma]) \mid [\sigma] \in B_p(K, \mathbb{R}) \}, \text{ where } \chi([\sigma']) = \begin{cases} 1 & \text{if } [\sigma'] = [\sigma] \\ 0 & \text{otherwise} \end{cases} \quad (3.25)$$

The functions  $\chi(*)$  are called *elementary cochains*. For any choice of inner product on  $C^p(K, \mathbb{R})$  where elementary cochains forms an orthogonal basis, there exists a positive weight function  $f : K \rightarrow \mathbb{R}_+ \setminus \{0\}$  satisfying:

$$\langle g, h \rangle_f = \sum_{\sigma \in K^p} w(\sigma) g([\sigma]) h([\sigma]) \quad (3.26)$$

Moreover, there is a one-to-one correspondence between the choice of weight function and possible scalar products on  $C^p(K, \mathbb{R})$  wherein the elementary cochains are orthogonal. In this way, we say that the choice of weight function *induces* an inner product on  $C^p(K, \mathbb{R})$ . Thus, we will use the terms *weight function* and *scalar product* interchangeably.

Given a simplicial complex  $K$ , the corresponding Laplace operator is uniquely determined by the weight function equipped to the faces of  $K$ . By choosing the weights, one effectively determines the corresponding inner product, which in-turn determines the spectral range of the corresponding operator. Computationally, the weights of the simplices  $\sigma \in K$  manifest in the corresponding Laplace operators via positive diagonal matrices  $W_* = \text{diag}(\{w(\sigma_1), w(\sigma_2), \dots, w(\sigma_n)\})$ , yielding the following *weighted combinatorial laplacian*:

$$\Delta_p(K, f) := \underbrace{W_p^{-1} \partial_{p+1} W_{p+1} \partial_{p+1}^T}_{L_p^{\text{up}}} + \underbrace{\partial_p^T W_p^{-1} \partial_p W_{p+1}}_{L_p^{\text{dn}}} \quad (3.27)$$

In general, none of the constitutive terms in (3.27) are symmetric matrices, unless  $W_p = I_n$  (for  $L_p^{\text{up}}$ ) or  $W_{p+1} = I_m$  (for  $L_p^{\text{dn}}$ ). However, note that  $L_p^{\text{up}}$  may be written as follows:

$$L_p^{\text{up}}(K, f) = W_p^{-1} \partial_{p+1} W_{p+1} \partial_{p+1}^T = W_p^{-1/2} (W_p^{-1/2} \partial_{p+1} W_{p+1} \partial_{p+1}^T W_p^{-1/2}) W_p^{1/2} \quad (3.28)$$

Notice that the right hand side of (3.28) is of the form  $W^{-1} P W$  where  $P$  is symmetric positive semi-definite and  $W$  is a positive diagonal matrix. Thus,  $L_p^{\text{up}}$  is matrix-similar to



$W_p^{-1/2} \partial_{p+1} W_{p+1} \partial_{p+1}^T W_p^{-1/2}$ , which is symmetric and positive semi-definite. The same result holds for up-, down-, and combinatorial Laplacians  $\square$ . We refer to this inner positive semi-definite matrix as the *symmetric* weighted up-Laplacian of  $K$ . For now we summarize this fact with a proposition.

**Proposition 7** ( $\square$ ). *Let  $f : (\mathbb{R}^n, H_n) \rightarrow (\mathbb{R}^m, H_m)$  be a linear map between inner product matrices  $H_n \in \mathbb{R}^{n \times n}$  and  $H_m \in \mathbb{R}^{m \times m}$ , and let  $F \in \mathbb{R}^{m \times n}$  denote the matrix representation of  $f$ . Then, for any  $x \in \mathbb{R}^n$  and  $y \in \mathbb{R}^m$ , we have the following equivalence of inner products:*

$$\langle fx, y \rangle_{\mathbb{R}^m} = \langle x, f^* y \rangle_{\mathbb{R}^n} = x^T F^T H_m y = x^T H_n F^* y$$

where  $F^* = H_n^+ F^T H_m$  is a representative of the adjoint  $f^* : (\mathbb{R}^m, H_m) \rightarrow (\mathbb{R}^n, H_n)$  of  $f$ .

Consider the  $\mathcal{H}$ -parameterized boundary matrix from Definition (1). Assume that the filter function  $f : K \rightarrow \mathbb{R}_+$  is strictly positive. By taking the product  $(\hat{\partial}_p^{i,j})(\hat{\partial}_p^{i,j})^T$  for some choice of  $h \in \mathcal{H}$ , we have:

$$\begin{aligned} \hat{L}_p^{\text{up}}(K, f) &= (\hat{\partial}_p^{i,j})(\hat{\partial}_p^{i,j})^T \\ &= (V_p^i)^+ \circ \partial_p \circ (W_{p+1}^j)^2 \circ \partial_p^T \circ (V_p^i)^+ \\ &\propto (V_p^i)^{+/2} \circ \partial_p \circ W_{p+1}^j \circ \partial_p^T \circ (V_p^i)^{+/2} \end{aligned} \quad (3.29)$$

where  $A^{+/2} = (A^{1/2})^+$ . Thus, in this way we think of our definition of a parameterized boundary matrix as corresponding to a choice of a *non-negative scalar-product* on the vector space  $C^p(K, \mathbb{R})$ , which induces a (degenerate) inner product  $\langle \cdot, \cdot \rangle : C^p(K, \mathbb{R}) \times C^p(K, \mathbb{R}) \rightarrow \mathbb{R}$  on  $C^p(K, \mathbb{R})$ . Equivalently, our choice of weight function induces an inner product on the quotient space  $W = C^p(K, \mathbb{R}) / \{b : |b| = 0\}$ . We expect the spectra of  $\hat{L}_p^{\text{up}}$  to obey many of the properties enjoyed by  $L_p^{\text{up}}$  and its variants. Indeed, though the spectrum of  $\hat{L}_p^{\text{up}}$  is unbounded for general choices of  $f$ , we show that the normalized up-Laplacian given by:

$$\hat{\mathcal{L}}_p^{\text{up}}(K, f) = \mathcal{D}_p(\mathcal{S}_i \circ f_\alpha)^{+/2} \partial_p W_{p+1}(\mathcal{S}_j \circ f_\alpha) \circ \partial_p^T \circ (\mathcal{D}_p^i)^{+/2} \quad (3.30)$$

has its spectrum bounded in the interval  $[0, p+2]$ , where  $\tilde{D}_p^i$  corresponds to the weighted degree matrix with entries  $\{\deg_w(\sigma)\}$ . We defer discussion of the exact forms of these matrices, including the computational details the matrix-vector product  $x \mapsto \hat{L}_p^{\text{up}}$ , to section (4.2).

## 4 Computational Implications

In this section, we discuss computational advantages that stem from replacing the counting invariants with their spectral-based approximations from definition 3. As the spectral relaxations are defined completely in terms of singular values of boundary operators—which can be obtained are the square roots of eigenvalues of an appropriate Laplacian operator—we focus on spectral methods specialized for symmetric positive semi-definite operators. In particular, we focus on *iterative* methods which only require matrix-vector products as their primary operations for estimating the spectrum, as this enables us to decouple the simplicial representation of the Laplacian operator from its explicit matrix representation.

In the following sections, we (1) recall the method of minimized iterations in section 4.1, (2) explore the efficiencies and subtleties of simplicial Laplacian `matvec` operators, and (3) discuss the modern advances to estimating spectral quantities of combinatorial Laplacians efficiently, such as preconditioning methods and convergence rates.

## 4.1 The Lanczos iteration

For a real, square matrix  $A$  of order  $n$ , the quadratic form  $x^T A x$  defines a continuous real-valued function of  $x \in \mathbb{R}^n$ . It is well known that when  $A$  is symmetric, each eigenvalue  $\lambda$  satisfying  $Av = \lambda v$  is real-valued and each pair of eigenvectors  $v, v' \in \mathbb{R}^n$  satisfying  $\lambda \neq \lambda'$  is orthogonal. Thus, we may reveal the spectrum  $\Lambda(A)$  of  $A$  via orthogonal diagonalization:

$$A = V \Lambda V^T = \sum_{i=1}^n \lambda_i v_i v_i^T \quad (4.1)$$

Factorizing  $A$  as in (4.1) is known as the *symmetric eigenvalue problem*. Computing the eigen decompositions of symmetric matrices generally consists of two phases: (1) reduction to tridiagonal form  $Q^T A Q = T$  via orthogonal similarity transformations  $Q$ , and (2) diagonalization of the tridiagonal form  $T = Y \Theta Y^T$ . While the latter may be performed in  $O(n \log n)$  time [18], the former is effectively bounded below by  $\Omega(n^3)$  for dense full-rank matrices using non-Strassen-like operations, and thus it is the reduction to tridiagonal form that dominates the computation. Lanczos [19] proposed the *method of minimized iterations*—now known as the *Lanczos method*—as an attractive alternative for reducing  $A$  into a tridiagonal form and thus revealing its spectrum. As the Lanczos method is the central iterative method we exploit in this effort, we review it below.

The means by which the Lanczos method estimates eigenvalues is by projecting onto successive Krylov subspaces. Given a large, sparse, symmetric  $n \times n$  matrix  $A$  with eigenvalues  $\lambda_1 \geq \lambda_2 > \dots \geq \lambda_r > 0$  and a vector  $v \neq 0$ , the order- $j$  Krylov subspaces of the pair  $(A, v)$  are the spaces spanned by:

$$\mathcal{K}_j(A, v) := \text{span}\{v, Av, A^2v, \dots, A^{j-1}v\} = \text{range}(K_j(A, v)) \quad (4.2)$$

where  $K_j(A, v) = [v \mid Av \mid A^2v \mid \dots \mid A^{j-1}v]$  are their corresponding Krylov matrices. Krylov subspaces arise naturally from using the minimal polynomial of  $A$  to express  $A^{-1}$  in terms of powers of  $A$ . In particular, if  $A$  is nonsingular and its minimal polynomial has degree  $m$ , then  $A^{-1}v \in \mathcal{K}_m(A, v)$  and  $\mathcal{K}_m(A, v)$  is an invariant subspace<sup>3</sup> of  $A$ . Since  $A$  is symmetric, the spectral theorem implies that  $A$  is orthogonally diagonalizable and that we may obtain  $\Lambda(A)$  by generating an orthonormal basis for  $\mathcal{K}_n(A, v)$ . To do this, the Lanczos method constructs successive QR factorizations of  $K_j(A, v) = Q_j R_j$  for each  $j = 1, 2, \dots, n$ . Due to  $A$ 's symmetry and the orthogonality of  $Q_j$ , the identity  $q_k^T A q_l = q_l^T A^T q_k = 0$  is satisfied for all  $k > l + 1$ , giving the corresponding  $T_j = Q_j^T A Q_j$  a tridiagonal structure:

$$T_j = \begin{bmatrix} \alpha_1 & \beta_1 & & & \\ \beta_1 & \alpha_2 & \beta_2 & & \\ & \beta_2 & \alpha_3 & \ddots & \\ & & \ddots & \ddots & \beta_{j-1} \\ & & & \beta_{j-1} & \alpha_j \end{bmatrix}, \quad \beta_j > 0, \quad j = 1, 2, \dots, n \quad (4.3)$$

---

<sup>3</sup>Recall that if  $S \subseteq \mathbb{R}^n$ , then  $S$  is called an *invariant subspace* of  $A$  or *A-invariant* iff  $x \in S \implies Ax \in S$  for all  $x \in S$ .

Unlike the spectral decomposition  $A = V\Lambda V^T$ —which identifies a diagonalizable  $A$  with its spectrum  $\Lambda(A)$  up to a change of basis  $A \mapsto M^{-1}AM$ —there is no canonical choice of  $T_j$  due to the arbitrary choice of  $v$ . However, there is a connection between the iterates  $K_j(A, v)$  and the full tridiagonalization of  $A$ : if  $Q^T A Q = T$  is tridiagonal and  $Q = [q_1 \mid q_2 \mid \cdots \mid q_n]$  is an  $n \times n$  orthogonal matrix  $Q Q^T = I_n = [e_1, e_2, \dots, e_n]$ , then:

$$K_n(A, q_1) = Q Q^T K_n(A, q_1) = Q [e_1 \mid T e_1 \mid T^2 e_1 \mid \cdots \mid T^{n-1} e_1] \quad (4.4)$$

is the QR factorization of  $K_n(A, q_1)$ . Thus, tridiagonalizing  $A$  with respect to a unit-norm  $q_1$  determines  $Q$ . Indeed, the Implicit Q Theorem [17] asserts that if an upper Hessenberg matrix  $T \in \mathbb{R}^{n \times n}$  has only positive elements on its first subdiagonal and there exists an orthogonal matrix  $Q$  such that  $Q^T A Q = T$ , then  $Q$  and  $T$  are *uniquely* determined by  $(A, q_1)$ . As a result, given an initial pair  $(A, q_1)$  satisfying  $\|q_1\| = 1$ , we may restrict and project  $A$  to its  $j$ -th Krylov subspace  $T_j$  via:

$$A Q_j = Q_j T_j + \beta_j q_{j+1} e_j^T \quad (\beta_j > 0) \quad (4.5)$$

where  $Q_j = [q_1 \mid q_2 \mid \cdots \mid q_j]$  is an orthonormal set of vectors mutually orthogonal to  $q_{j+1}$ . Equating the  $j$ -th columns on each side of (4.5) and rearranging the terms yields the *three-term recurrence*:

$$\beta_j q_{j+1} = A q_j - \alpha_j q_j - \beta_{j-1} q_{j-1} \quad (4.6)$$

where  $\alpha_j = q_j^T A q_j$ ,  $\beta_j = \|r_j\|_2$ ,  $r_j = (A - \alpha_j I) q_j - \beta_{j-1} q_{j-1}$ , and  $q_{j+1} = r_j / \beta_j$ . Equation (4.6) is a variable-coefficient second-order linear difference equation, and it is a known fact that such equations have unique solutions: if  $(q_{j-1}, \beta_j, q_j)$  are known, then  $(\alpha_j, \beta_{j+1}, q_{j+1})$  are completely determined. The sequential process that iteratively builds  $T_j$  via the recurrence from (4.6) is called the *Lanczos iteration*. Note that if  $A$  is singular and we encounter  $\beta_j = 0$  for some  $j < n$ , then  $\text{range}(Q_j) = \mathcal{K}_j(A, q_1)$  is an  $A$ -invariant subspace, the iteration stops, and we have solved the symmetric eigenvalue problem (4.1):  $\Lambda(T_j) = \Lambda(A)$ ,  $j = \text{rank}(A)$ , and  $T_j$  is orthogonally similar to  $A$ .

The Lanczos iteration and its many variants are part of a family of so-called “matrix free” methods—obtaining an eigen-decomposition of a symmetric real matrix  $A$  requires only a matrix-vector  $v \mapsto Av$  operator. Indeed, since  $A$  is not modified during the computation, the entire iteration may be carried out without explicitly storing  $A$  in memory. Moreover, the three-term recurrence from (4.6) implies the Lanczos iteration requires just three  $O(n)$ -sized vectors and a few  $O(n)$  vector operations to execute. We summarize these benefits with a proposition.

**Proposition 8** ([22, 23]). *Given a symmetric rank- $r$  matrix  $A \in \mathbb{R}^{n \times n}$  whose matrix-vector operator  $A \mapsto Ax$  requires  $O(\eta)$  time and  $O(\nu)$  space, the Lanczos iteration computes  $\Lambda(A) = \{\lambda_1, \lambda_2, \dots, \lambda_r\}$  in  $O(\max\{\eta, n\} \cdot r)$  time and  $O(\max\{\nu, n\})$  space, when computation is done in exact arithmetic.*

As in [22], the assumption of exact arithmetic simplifies both the presentation of the theory and the corresponding complexity statements. Although this assumption is unrealistic in practical settings, it gives us a grounded expectation of what is possible to achieve with any *finite-precision* algorithm based on the Lanczos method.

**Corollary 1.** *Given the same inputs as Lemma 8, any implementation that computes  $\Lambda(A) = \{\lambda_1, \lambda_2, \dots, \lambda_r\}$  using the Lanczos iteration in finite-precision arithmetic requires  $\Omega(\max\{\eta, n\} \cdot r)$  time and  $\Omega(\max\{\nu, n\})$  space complexity.*

In practice, finite-precision arithmetic introduces both rounding and cancellation errors into the computation, which manifests as loss of orthogonality between the Lanczos vectors. These errors not only affect the methods convergence rate towards an invariant subspace, but in fact they muddle the termination condition entirely. As a result, several decades of research have been dedicated to developing orthogonality-enforcement schemes that retain the simplicity of the Lanczos iteration without increasing either the time or space complexities by non-trivial factors. As these extensions are complex, multifaceted, and beyond the scope of the present work, we defer their discussion to the appendix A.2 and refer the curious reader to [17] and references therein for an overview.

## 4.2 The combinatorial Laplacian matvec

As proposition 8 shows, the efficiency of the Lanczos method depends crucially on the existence of the three-term recurrence and a fast matrix-vector product. The former arises in the decomposition of symmetric matrices while the latter depends on the structure of the  $x \mapsto Ax$  operation. In general, a symmetric rank- $r$  matrix  $A \in \mathbb{R}^{n \times n}$  with an average of  $\nu$  nonzeros per row incurs approximately  $(2\nu + 8)n$  floating point operations (FLOPS) per Lanczos step, implying a  $O(n\nu r)$  time complexity for a single iteration [17]. This is markedly improved in the case of the graph Laplacian derived from a graph  $G = (V, E)$ : the  $x \mapsto \partial_p \partial_p^T x$  operation can be carried out in a time linear in  $|E|$  by summing across edges—no sparse matrix need be explicitly constructed. In contrast, it is not clear one may readily generalize this process to the combinatorial Laplacian for more general  $p > 1$  as the form of the `matvec` depends on the specific Laplace operator one considers. Towards accelerating the computation, we examine these nuanced differences more in detail.

To begin understanding the computational nuances associated with simplicial Laplacian operators, first consider the graph Laplacian  $L = D - A$  derived from an undirected simple graph  $G = (V, E)$ . As in [9], the linear and quadratic forms of  $L$  may be succinctly expressed using the path-connected relation  $i \sim j$ , shown below:

$$(\forall x \in \mathbb{R}^n) \quad (Lx)_i = \deg(v_i) \cdot x_i - \sum_{i \sim j} x_j, \quad x^T Lx = \sum_{i \sim j} (x_i - x_j)^2 \quad (4.7)$$

Observe that, if  $G = (V, E)$  has  $m$  edges and  $n$  vertices taking labels in the set  $[n]$ , the matrix-vector product in (4.7) can be evaluated in  $O(m)$  time and  $O(n)$  storage via two  $O(m)$  edge traversals: one to accumulate the degree of each vertex, and one to subtract off components  $x_j$  from incident edges at  $i$ . By precomputing the degree of each vertex and storing an auxiliary  $O(n)$  amount of memory, this computation can be reduced to a one  $O(n)$  vector multiplication and a single  $O(m)$  pass through the edges, which is often beneficial for repeated computation. Note that the exact form of equation (4.7) does not immediately generalize to combinatorial Laplacians for  $p > 0$ , as the orientation of higher dimensional simplices changes the sign of the off-diagonal entries in  $L_p^*$  and it is not immediately clear how to generalize the notion of path-connected to higher dimensions.

Noting the observations in section 3.2, we exclusively consider the symmetric version of  $\hat{L}_p^{\text{up}}$ . The down-Laplacian  $\hat{L}_p^{\text{dn}}$  can be handled in essentially the same way. Towards extending (4.7) to higher dimensions, we need to generalize the path-connected relation  $i \sim j$ . Let  $\text{co}(\tau) = \{\sigma \in K^{p+1} \mid \tau \subset \sigma\}$  denote the set of proper cofaces of  $\tau \in K^p$ , or *cofacets*, and let

$$\deg_f(\tau) = \sum_{\sigma \in \text{co}(\tau)} f(\sigma)$$

denote the (weighted) *degree* of  $\tau$ . Observe setting  $f(\sigma) = 1$  for all  $\sigma \in K$  recovers the integral notion of degree representing the number of cofacets a given simplex has. Now, since  $K$  is a simplicial complex, if  $\text{co}(\tau) \cap \text{co}(\tau') \neq \emptyset$  then the faces  $\tau, \tau'$  not only share a common cofacet  $\sigma$ , but this cofacet  $\{\sigma\} = \text{co}(\tau) \cap \text{co}(\tau')$  is *unique* [16]. Thus, given two distinct  $p$ -simplices  $\tau \neq \tau'$  in  $K^p$  we may use the notation  $\tau \stackrel{\sigma}{\sim} \tau'$  to denote this cofacet relation and now the operator from (3.28) can now be written element-wise as:

$$\hat{L}_p^{\text{up}}(\tau, \tau') = \begin{cases} \deg_f(\tau) \cdot f^+(\tau) & \text{if } \tau = \tau' \\ s_{\tau, \tau'} \cdot f^{+1/2}(\tau) \cdot f(\sigma) \cdot f^{+1/2}(\tau') & \text{if } \tau \stackrel{\sigma}{\sim} \tau' \\ 0 & \text{otherwise} \end{cases} \quad (4.8)$$

where  $s_{\tau, \tau'} = \text{sgn}([\tau], \partial[\sigma]) \cdot \text{sgn}([\tau'], \partial[\sigma])$ . Ordering the  $p$ -faces  $\tau \in K^p$  along a total order and choosing an indexing function  $h : K^p \rightarrow [n]$  enables explicit computation of the corresponding matrix-vector product:

$$(L_p^{\text{up}} x)_i = \deg_f(\tau) \cdot f(\tau) \cdot x_i + \sum_{\tau \stackrel{\sigma}{\sim} \tau'} s_{\tau, \tau'} \cdot x_{h(\tau')} \cdot f^{+1/2}(\tau) \cdot f(\sigma) \cdot f^{+1/2}(\tau') \quad (4.9)$$

Observe (4.9) can be evaluated now in the same  $O(m(p+1))$  time and  $O(n)$  storage, where  $m = |K^{p+1}|$  and  $n = |K^p|$  via the two-pass algorithm described for the graph Laplacian if both the sign term  $s_{\tau, \tau'}$  and the indexing function  $h$  can be evaluated in  $O(1)$  time.

In general, the signs of the coefficients  $\text{sgn}([\tau], \partial[\sigma])$  and  $\text{sgn}([\tau'], \partial[\sigma])$  depends on the position of  $\tau, \tau'$  as summands in  $\partial[\sigma]$  via (2.3), which itself depends on the orientation of  $[\sigma]$  defined by (2.2). Thus, naive evaluation of these sign terms takes  $O(k)$  time to determine for a given  $\tau \in \partial[\sigma]$  with  $\dim(\sigma) = k$ . However, observe that the sign of their product is in fact invariant in the orientation of  $[\sigma]$  (see Remark 3.2.1 of [16]). Thus, if we fixed the orientation of the simplices of  $K^p$ , the sign pattern  $s_{\tau, \tau'}$  for every  $\tau \stackrel{\sigma}{\sim} \tau'$  can be precomputed and stored ahead of time, reducing the evaluation  $s_{\tau, \tau'}$  in  $O(1)$  time and  $O(m)$  storage. Alternatively, if the labels of the  $p+1$  simplices  $\sigma \in K^{p+1}$  follow a consistent orientation—for example by inducing the orientation of all  $[\sigma] \in K^{p+1}$  via the total order on  $V$ —then we can remove the storage requirement entirely and simply follow a fixed sign pattern during the computation.

A subtle but important aspect of algorithmically evaluating (4.9) is the choice of indexing function  $h : K^p \rightarrow [n]$ . This map is necessary to deduce the contributions of the components  $x_*$  during the operation. While one may use any standard associative array to generate this map, typical implementations that rely on collision-resolution schemes such as open addressing or chaining only have  $O(1)$  lookup time in expectation. Taking advantage again of the fact that our simplicial complex  $K$  is fixed, we note that one can build a hash function  $h$  with deterministic  $O(1)$  access by building a *minimal perfect hash function* (PMHF) on the

simplices  $\tau \in K^p$ . It is known how to build PMHF's over fixed input sets of size  $n$  in  $O(n)$  time and  $O(n \log m)$  bits []. Note that this process happens only once for a fixed simplicial complex  $K$ : once  $h$  has been constructed, it is fixed for every `matvec` operation.

Since a  $p+1$ -simplex has  $p$  proper faces in its boundary, and we need to iterate through the  $p+1$  simplices  $\sigma \in K^{p+1}$  just once in the case of the up-Laplacian, we may combine the above observations into a proposition.

**Proposition 9.** *For any  $p \geq 0$  and simplicial pair  $(K, f)$ , there exists a two-phase algorithm for computing the product:*

$$L_p^{\text{up}}x = (W_p \circ \partial_{p+1} \circ W_{p+1} \circ \partial_{p+1}^T \circ W_p)x \quad (4.10)$$

*in  $O(m(p+1))$  time and  $O(n)$  storage, where  $n = |K^p|$  and  $m = |K^{p+1}|$ .*

From a practical perspective,  $p \geq 0$  is generally a very small constant—typically no more than 2—thus we conclude that we can evaluate  $x \mapsto \partial_{p+1} \partial_{p+1}^T x$  is  $\approx O(m)$  time, essentially linear in the number of  $p+1$  simplices. More details on the practical optimization one may make, along with pseudocode of the two-phase algorithm for general  $p \geq 0$ , is given in appendix C.

## 5 Applications

### 5.1 Matrix-free persistence computation

The first and most general application of the work presented here is the matrix-free computation of the spectra of various combinatorial Laplacian operators. Since we may recover both the persistence diagram [8] and the rank invariants by repeated spectral computations, we focus examining the time and storage characteristics of Laplacian spectral computation itself rather than its corresponding persistence computations.

We sampled 30 random graphs according to the Watts-Strogatz [] rules with parameters  $n = 500, k = 10, p = 0.15$ . These graphs tend to exhibit ‘small world’ characteristics inherited by many real-world networks, such as social networks, gene networks, and transportation networks. For our purposes, since the graph distance between pairs of nodes scale logarithmically with the size of the graph, we ensure the sampled random graphs to be uniformly sparse. The corresponding incidence matrix  $\partial_1 \in \mathbb{R}^{n \times m}$  and up-Laplacians  $L_0^{\text{up}} \in \mathbb{R}^{n \times n}$  would have  $\approx 5,000$  and  $\approx 5,500$  non-zero entries, were they to be formed explicitly, which are weighted according randomly by embedding the graph in the plane and filtering graph via its sublevel sets in a random direction. A much smaller Watts-Strogatz graph of the same type (but with only 50 nodes) is shown on the left-side of figure 3, colored by the filtering of its lower stars. To test the scability of the laplacian operator studied here, we computed various

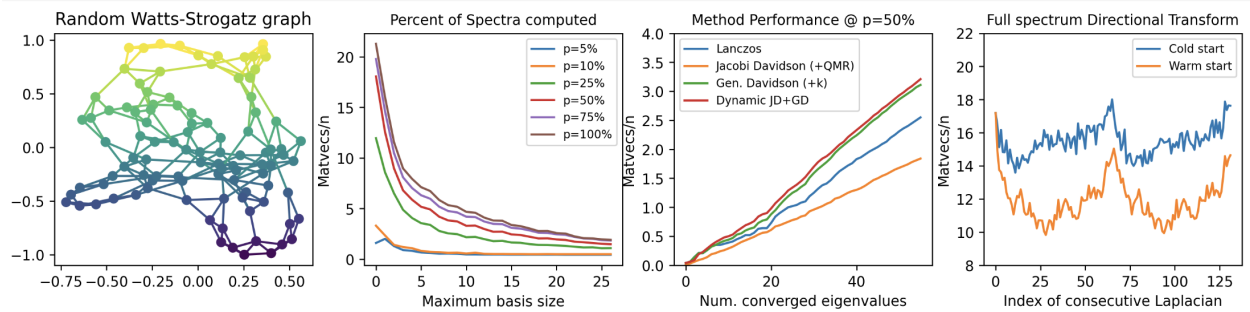
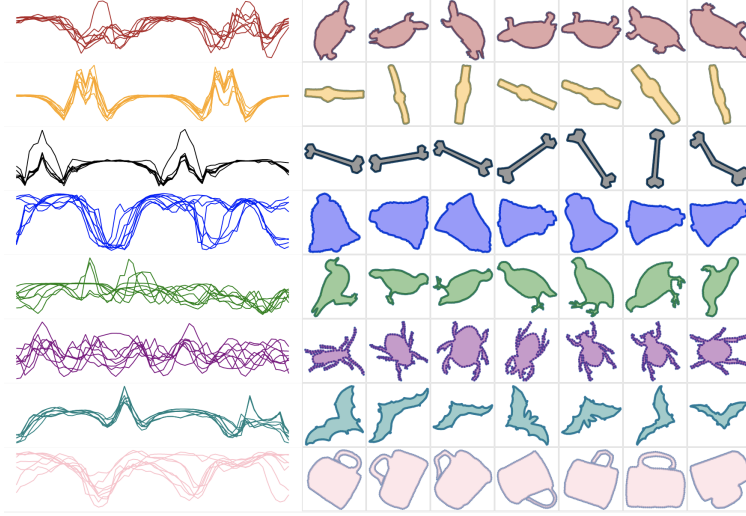


Figure 3: Random Watts-Strogatz “Small world” graph example

percentages of the spectra of these 30 graphs via iterative methods discussed in section ?? and reported various of their time- and storage- related statistics in figure 3. All statistics reported are the average statistics collected from all 30 random graphs, which were collected using various iterative methods implemented the PRIMME software []. On the far left of figure 3, we display a random metric embedding of a small Watts-Strogatz graph to convey the structure of the type of graphs we consider.

**Storage requirements:** On the left side of figure 3 next to the examle network model, we record the ratio of `matvec` operations (relative to  $n$ ) needed to compute  $p\%$  of the spectrum as a function of the maximum number basis vectors kept in-memory for reorthogonalization purposes. The ideal Lanczos method needs just 3 such vectors in exact arithmetic due to the three-term recurrence, justifying the space complexity record in 8; in contrast, with finite-precision arithmetic, one needs additional basis vector to ensure the orthonormality of



the eigenvectors to machine precision. Each additional basis vector simultaneously increases both the cost of performing a Lanczos step and the accuracy of the orthogonalization, which subsequently decreases the number of total  $\triangleright \triangleright \approx \approx \approx$  operations needed. As one can see from the plot, having  $\approx 20 - 25$  basis vectors is more than enough to ensure the ratio of `matvec` operations is kept to a small constant (in this case, less than 5) when approximating any portion of the spectra. This justifies our claim that combinatorial Laplacian operators, for many real-world data sets, requires just  $O(m)$  memory complexity to compute eigenvalues (and thus, the persistent rank invariants).

**Time requirements:** The remaining two figures on the right side of figure 3 show the same ratio of `matvecs`/ $n$ —effectively the constant associated with quadratic time complexity statement in 8—

## 5.2 Shape comparison

In general, both combinatorial and topological aspects of a given topological space are encoded in the spectra of Laplacian operators. For example, the Laplacians of a simplicial complex encode its basic topology via its homology groups, which is characterized by the nullspace of the corresponding operator—this is identical for most of the Laplace operators, whether they are normalized, weighted, signless, and so on [1]. In contrast, these operators differ in the nonzero part of the spectrum, which when equipped with a scalar-product encode specific geometric features in addition to topological properties.

## 5.3 Filtration optimization

A common setting in topological data analysis is the setting wherein one has access to a means of building a filtration  $(K, f)$  where  $f : K \rightarrow \mathbb{R}$  is a filter function satisfying  $f(\tau) \leq f(\sigma)$  for all  $\tau \subseteq \sigma \in K$ , but the filter function  $f$  itself is parameterized by some hyper-parameters. For example, a common setting is the one where the data set  $(X, d_X)$  comes equipped with some notion of density  $f : X \rightarrow \mathbb{R}_+$ , and one would like to build a persistence diagram on  $d_X$  in a way that is robust to local fluctuations in density. This is a common practical setting often



encountered in practice, as it is known that persistence is unstable with respect to *strong outliers* [], which prevents persistent homology from detecting a spaces prominent underlying topological structure, when it exists. Most work seeking to remedy this issue proceeds by either (1) removing such outliers according to some heuristic [], (2) transforming the metric to lessen the importance of such points in the persistence diagram [], or (3) creating a 2-parameter persistence module with one dimension filtered by (co)-density. Of the three, (1) is ultimately a heuristic not useful for complex data sets as it discards important data; (2) imposes a parameter that must be set to proceed, and (3) is perhaps ideal but currently considered both analytically and computationally intractable in practice.

To illustrate an alternative approach to the ways mentioned above, consider a fixed Delaunay complex  $K$  built on a set of points sampled noisily around  $S^1$  in the plane, shown in figure 4. Ultimately, we would like to detect the presence of the circle in  $X$  via its persistence diagram, as that is the original purpose of persistence []. We reframe the problem as follows: rather than filtering  $K$  according to its ambient metric  $d_X$ , we ask first whether there *exists* significant topological information at any density scale  $\alpha$ . Generically, we consider the following optimization problem:

$$\alpha^* = \arg \max_{\alpha \in \mathbb{R}} \mu_p(K, f_\alpha)|_R \quad (5.1)$$

If there exists a density scale  $\alpha \in \mathbb{R}_+$  wherein a cycle  $c \in H_1(K; \mathbb{R})$  is highly persistent and  $R \subset \Delta_+$  is appropriately chosen, then any maximizer of (5.1) yields an appropriate density scale  $\alpha^*$  with which to detect the topology of the data. From another perspective, we refer to this process of choosing appropriate filtration (hyper-)parameters to yield persistence information as *filtration optimization*.

As an introductory example, we sampled 80 points noisily around  $S^1$  and then sampled an additional 30 points in  $[-1, 1] \times [-1, 1]$  to act as “strong outliers.” After constructing a Delaunay complex  $K$  of these points, we parameterized a filter function  $f_\alpha : K \times \mathbb{R} \rightarrow \mathbb{R}_+$  by assigning the filtration values of each simplex according to the lower stars of a kernel (co)density estimate, upon which we computed its vineyard [11] along a subset  $\mathcal{A} \subset \mathbb{R}$ . The vineyard, colored by  $\alpha$ , is shown on the left side of figure 5. In this example, we choose the rectangle  $R = \frac{1}{5}([1, 2] \times [3, 4])$  out of simplicity; the corresponding multiplicity function is shown in the black curve on the right of figure 5. By inspection, the optimal density parameter satisfying (5.1) is any parameter  $\alpha$  lying approximately in the interval  $[0.40, 0.44]$ . Observe that any first-order optimization procedure initialized in the interval  $\alpha_0 \in [0.3, 0.5]$  yields a maximizer  $\hat{\alpha}$  in the interval  $[0.36, 0.46]$ , which is quite close to the interval  $[0.40, 0.44]$ . In this toy example, this is sufficient, however if a better estimate was required (e.g. the multiplicity was required to be positive as a constraint) then observe one could iteratively shrink  $\epsilon$  to obtain a better approximation of  $\mu_1$ , and then repeat the first-order optimization. This is synonymous to the *iterative thresholding* techniques often in high-dimensional statistics and machine learning, see [] for an overview.

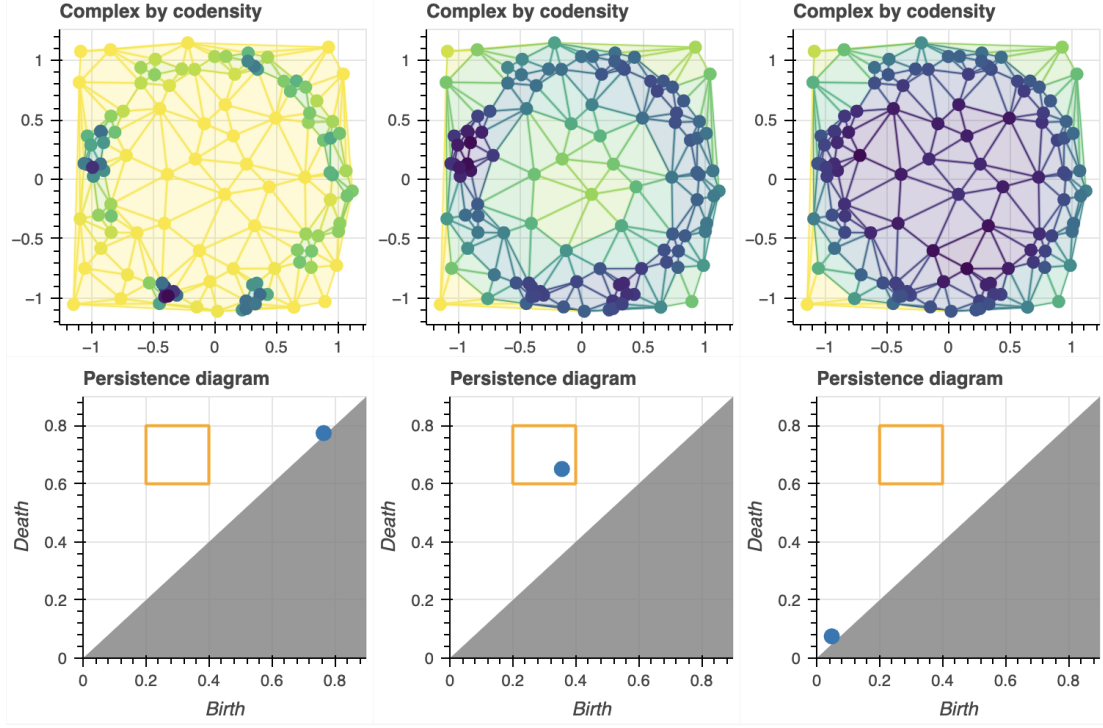


Figure 4: A fixed Delaunay complex filtered by a kernel (co)density estimate for different bandwidth parameters  $\alpha$ . Observe that either too small (left) or too large (right) a choice of bandwidth can obscure the underlying topological structure, whereas an appropriate choice of bandwidth creates a filtration that detects the underlying circle.

## A Appendix

### Expanded Intro

Though homology is primarily studied as a topological invariant, the fact that persistent homology encodes both topological and geometric information in its diagram has motivated its use not only as a shape descriptor but also as a metric invariant. Metric invariants, or “signatures,” are commonly used in metric learning to ascertain whether two comparable data sets  $X, Y$  represent the same object—typically up to a some notion of invariance. One mathematically attractive model for measuring the dissimilarity between shapes/datasets is the Gromov-Hausdorff (GH) distance  $d_{\text{GH}}((X, d_X), (Y, d_Y))$  between compact metric spaces  $(\mathcal{X}, d_X), (\mathcal{Y}, d_Y)$ : by altering the choice of metric  $(d_X, d_Y)$ , the corresponding metric-distance  $d_{\text{GH}}$  can be adapted to a chosen notion of invariance [1] or to increase its discriminating power [2]. Though it is NP-hard to compute [3], the GH distance defines a metric on the set of isomorphism classes of compact metric spaces endowed with continuous real-valued functions, justifying its study as a mathematical model for shape matching and metric learning. Moreover, it is known that the GH distance is tightly lower-bounded by the bottleneck distance between persistence diagrams constructed over Rips filtrations  $R(X, d_X), R(Y, d_Y)$  [4], which can be computed in polynomial time. Indeed, Solomon et al [5] showed distributed persistence invariants characterize the quasi-isometry type of the underlying space, allowing

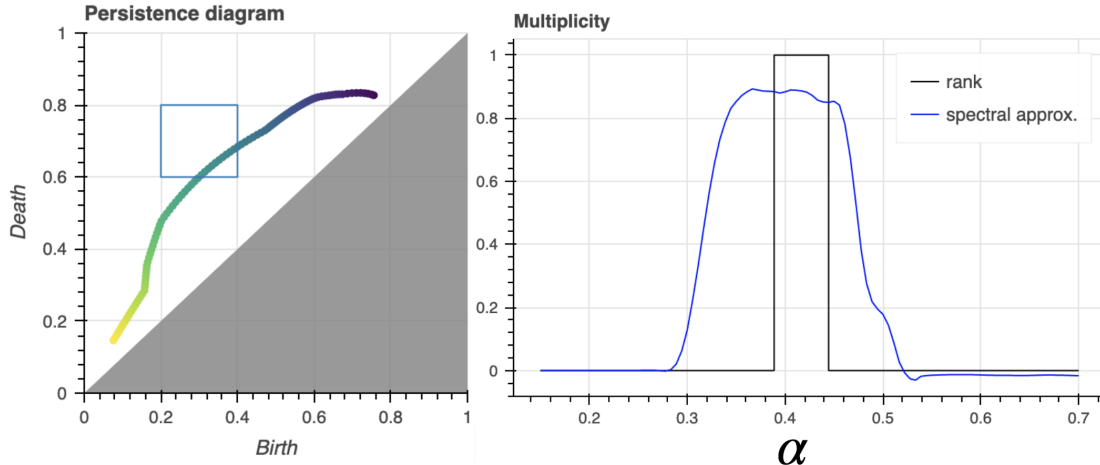


Figure 5: (Left) Vineyard of the codensity  $\alpha$ -parameterized filtration from figure 4. (Right) The exact multiplicity  $\mu_1(K, f_\alpha)$  (black) and the proposed spectral relaxation (smoothed, blue) with relaxation parameter  $\epsilon = 1e-3$ .

one to provably interpolate between geometric and topological structure.

Though theoretically well-founded and information dense, persistence diagrams come with their own host of practical issues: they are sensitive to strong outliers, far from injective, and their de-facto standard computation exhibits high algorithmic complexity. Moreover, the space of persistence diagrams  $\mathcal{D}$  is a Banach space, preventing one from doing even basic statistical operations, such as averaging [1]. As a result, many researchers have focused on extending, enhancing, or otherwise supplementing persistence diagrams with additional information. Turner et al [2] proposed associating a collection of shape descriptors with a PL embedded  $X \subset \mathbb{R}^d$ —one descriptor for each point on  $S^{d-1}$ —which they called a *transform*. More exactly, suppose both the data  $X$  and its geometric realization  $K$  are PL embedded in  $\mathbb{R}^d$  and has centered and scaled appropriately. The main theorem in [2] is that associating a persistence diagram, or even a simpler descriptor such as the Euler characteristic, for every point on  $S^{d-1}$  is actually sufficient information to theoretically reconstruct  $K$ .

Missing from the above work is the are two important directions: how do you configure such transforms to retain the important topological/geometric information and discard irrelevant information, and (2) how may we efficiently compute them? The former question is synonymous with choosing the invariance model in the GH framework, which seems to be highly domain specific. In the latter case, though we know the number of directions is bounded [3], the bound is simply too high to be of any practical use. While there are efficient algorithms for both the ECC and persistence computations in static settings, the state of the art in parameterized settings is non-trivial and ongoing research area.

## Letters

As topological invariants, Betti numbers are invariant under homeomorphisms: any pair of filtrations  $(K, f)$  and  $(K', f')$  that are homotopy equivalent have identical homology classes and thus isomorphic persistence diagrams. This invariance can be a useful thing at the level

of homology, as non-homeomorphic spaces can sometimes be differentiated by inspecting differences between their corresponding homology classes. However, invariance under homeomorphisms can at times discard geometric information that may be useful for differentiating objects. For example, consider creating a classifier for the alphabet of English characters in the font shown below:

**A B C D E F G H I J K L M N O P Q R S T U V W X Y Z**

If one were to triangulate images of each of the letters shown above and compute their Betti numbers, one would find just three homology classes: one class for those letters that have two holes (B), one class of letters that have one hole (A, D, O, P, Q, and R), and one class for the rest of the letters, which collapse to points. Thus, if one were concerned with differentiating letters of the alphabet, one may conclude that homology is not simply not strong enough of an invariant to do so.

It would be beneficial to have an invariant that was sensitive to the geometries between shapes, but also stable in some sense.

## Directional Transform

The canonical interpretation of the information displayed by a persistence diagram is that it summarizes the persistence of the sublevel sets of filtered space. Given a filtration pair  $(K, f)$  where  $K$  is a finite simplicial complex and  $f : K \rightarrow \mathbb{R}$  is a real-valued function, the sublevel sets  $|K|_i = f^{-1}(-\infty, i]$  deformation retract to... If  $K$  is embedded in  $\mathbb{R}^d$ , then geometrically  $f$  takes on the interpretation of a ‘height’ function whose range yields the ‘height’ of every simplex in  $K$ .

Let  $X \subset \mathbb{R}^d$  denote a data set which can be written as a finite simplicial complex  $K$  whose simplices are PL-embedded in  $\mathbb{R}^d$ . Given this setting, define the *directional transform* (DT) of  $K$  as follows:

$$\begin{aligned} \text{DT}(K) : S^{d-1} &\rightarrow K \times C(K, \mathbb{R}) \\ v &\mapsto (K_\bullet, f_v) \end{aligned}$$

where we write  $(K_\bullet, f)$  to indicate the filtration on  $K$  induced by  $f_v$  for all  $\alpha \in \mathbb{R}$ , i.e.:

$$K_\bullet = K(v)_\alpha = \{x \in X \mid \langle x, v \rangle \leq \alpha\} \quad (\text{A.1})$$

Conceptually, we think of DT as an  $S^{d-1}$ -parameterized family of filtrations.

The Persistent Homology Transform (PHT) is a shape statistic that establishes a fundamental connection between the topological information summarized by  $K$ ’s PH groups and the geometry of its associated embedding. Given a complex  $K$  built from  $X$ , it is defined as:

$$\begin{aligned} \text{PHT}(K) : S^{d-1} &\rightarrow \mathcal{D}^d \\ v &\mapsto (\text{dgm}_0(K, v), \text{dgm}_1(K, v), \dots, \text{dgm}_{d-1}(K, v)) \end{aligned} \quad (\text{A.2})$$

where  $\mathcal{D}$  denotes the space of  $p$ -dimensional persistence diagrams, for all  $p = 0, \dots, d-1$  and  $S^{d-1}$  the unit  $d-1$  sphere. The stability of persistence diagrams ensures that the map  $v \mapsto \text{dgm}_p(K, v)$  is Lipschitz with respect to the bottleneck distance metric  $d_B(\cdot, \cdot)$  whenever  $K$  is a finite simplicial complex. Thus, the PHT may be thought of as an element in  $C(S^{d-1}, \mathcal{D}^d)$ .

The primary result of [ ] is that the PHT is injective on the space of subsets of  $\mathbb{R}^d$  that can be written as finite simplicial complexes<sup>4</sup>, which we denote as  $\mathcal{K}_d$ . Equivalently,  $\mathcal{K}_d$  decomposes space of all pairs  $(K, f)$  under the equivalence  $(K, f) \sim (K, f')$  when  $f(K) = f'(K)$ .

### A.1 Complexity of Persistence & Related work

We briefly recount the main complexity results of the persistence computation. With a few key exceptions, the majority of persistent homology implementations and extensions is based on the *reduction algorithm* introduced by Edelsbrunner and Zomorodian [15]. This algorithm factorizes the filtered boundary into a decomposition  $R = \partial V$ , where  $V$  is full rank upper-triangular and  $R$  is said to be in reduced form: if its  $i$ -th and  $j$ -th columns are

---

<sup>4</sup>Implicit in the injectivity statement of the PHT is that, given a subset  $X \subset \mathbb{R}^d$  which may be written as finite simplicial complex  $K$ , the restriction  $f : X \rightarrow \mathbb{R}$  to any simplex in  $K$  must be linear.

nonzero, then  $\text{low}_R(i) \neq \text{low}_R(j)$ , where  $\text{low}_R(i)$  denotes the row index of the lowest non-zero in column  $i$ . We refer to [15, 2, 12] for details.

Given a filtration  $(K, f)$  of size  $m = |K|$  with filter  $f : K \rightarrow [m]$ , the reduction algorithm in form given in [15] computes  $\text{dgm}_p(K; \mathbb{Z}/2) = \{(\tau_1, \sigma_1), (\tau_2, \sigma_2), \dots, (\tau_k, \sigma_k)\}$  runs in time proportional to the sum of the squared (index) persistences  $\sum_{i=1}^k (f(\sigma_i) - f(\tau_i))^2$ . As  $k$  is at most  $m/2$ , this implies a  $O(m^3)$  upper bound on the complexity of the general persistence computation, which incidentally Morozov showed was a tight  $\Theta(m^3)$  under the assumption that each column reduction takes  $O(m)$  time. By exploiting the matrix-multiplication results, a similar result can be shown to reduce to  $O(m^\omega)$ , where  $\omega$  is the matrix-multiplication constant, which is  $\approx 2.37$  as of this time of writing. It worth remarking that the complexity statements above are all given in terms of the number of *simplices*  $m$ : if  $n = |K^0|$  is the size of the vertex set, the above implies a worst-case bound of  $O(n^{\omega(p+2)})$  on the general persistence computation. For example, if we use non-Strassen-based matrix multiplication ( $\omega = 3$ ) and we are concerned with  $p = 1$  homology computation, the complexity of the reduction algorithm scales  $O(n^9)$  in the number of vertices of the complex, which is essentially intractable for most real world application settings.

Despite the seemingly immense intractability of the persistence computation, decades of advancements have been made in reducing the complexity or achieving approximate results in reasonable time and space complexities. The complexity of the reduction algorithm is complicated by the fact that it depends heavily on the structure of the associated filtration  $K$ , the homology dimension  $p$ , the field of coefficients  $\mathbb{F}$ , and the assumptions about the space  $K$  manifests from. In [1], Sheehy presented an algorithm for producing a sparsified version  $(\tilde{K}, \tilde{f})$  of a given Vietoris-Rips filtration  $(K, f)$  constructed from an  $n$ -point metric space  $(X, d_X)$  whose total number of  $p$ -simplices is bounded above by  $n \cdot (\epsilon^{-1})^{O(pd)}$ , where  $d$  is the doubling dimension of  $X$ . It was shown that  $\text{dgm}_p(\tilde{K})$  is guaranteed to be a multiplicative  $c$ -approximation to the  $\text{dgm}_p(K)$ , where  $c = (1 - 2\epsilon)^{-1}$  and  $\epsilon \leq 1/3$  is a positive approximation parameter. When  $p = 0$  and the filtration function  $f : K \rightarrow \mathbb{R}$  is PL, the reduction algorithm can be bypassed entirely in favor of simple  $O(n \log n + \alpha(n)m) \approx O(m)$  algorithm (see Algorithm 5 in [12]), where  $n = |K^0|$  and  $m = |K^1|$  and  $\alpha(n)$  is the extremely slow-growing inverse Ackermann function. Moreover, the  $d - 1$  persistence pairs can be computed in  $O(n\alpha(n))$  time algorithm for filtrations of simplicial  $d$ -manifolds essentially reducing the problem to computing persistence on a dual graph [12]. For clique complexes, the apparent pairs optimization—which preemptively removes zero-persistence pairs from the computation prior to the reduction—has been empirically observed to reduce the number of columns needing reduced for clique complexes by  $\approx 98 - 99\%$  [2]. Numerous other optimizations, including e.g. the *clearing optimization*, the use of *cohomology*, the *implicit reduction* technique, have further reduced both the non-asymptotic constant factors of the reduction algorithm significantly, see [2] and references therein for a full overview.

Despite the dramatic reductions in time and space needed for the persistence algorithm to complete, to the author knowledge relatively little has been done in improving the complexity and effective runtime of the reduction in parameterized settings. Although both of these algorithms have shown significant constant-factor reductions in the (re)-reduction of the associated sparse matrices, all of the techniques require  $O(m^2)$  storage to execute as the  $R$  and  $V$  matrices must be maintained throughout the computation. Moreover, all

three of the above methods intrinsically work within the reduction framework, wherein simulating persistence in dynamic contexts effectively reduces to the combinatorial problem of maintaining a valid  $R = \partial V$  decomposition.

As noted in [12], the reduction algorithm is essentially a variant of Gaussian elimination. Indeed, the persistence of a given filtration can be computed by the PLU factorization of a matrix. The explicit decompositional approach of factorizing a large matrix into constitutive parts is known historically in numerical linear algebra as a *direct method*—methods would yield the exact solution within a finite number of steps. In contrast, iterative methods start with approximate solution and progressively update the solution up to arbitrary accuracy. The iterative methods well-known to the numerical linear algebra community, such as Krylov methods, are typically often attractive not only due to the reduction in computational work over direct approaches but also of the limited amount of memory that is required. Despite the success of iterative methods in efficiently solving linear systems manifesting from diagonally dominant sparse matrices is [], such advancements have not yet been extended to the persistence setting.

## Output sensitive multiplicity and Betti

We record this fact formally with two corollaries. Let  $R_p(k)$  denotes the complexity of computing the rank of square  $k \times k$  matrix with at most  $O((p+1)k)$  non-zero  $\mathbb{F}$  entries. Then we have:

**Corollary 2.** *Given a filtration  $K_\bullet$  of size  $N = |K_\bullet|$  and indices  $(i, j) \in \Delta_+^N$ , computing  $\beta_p^{i,j}$  using expression (2.11) requires  $O(\max\{R_p(n_i), R_{p+1}(m_j)\})$  time, where  $n_i = |K_i^p|$  and  $m_j = |K_j^{p+1}|$ .*

Observe the relation  $\partial_{p+1}^{i+1,j} \subseteq \partial_{p+1}^{1,j}$  implies the dominant cost of computing (2.11) lies in computing either  $\text{rank}(\partial_p^{1,i})$  or  $\text{rank}(\partial_{p+1}^{1,j})$ , which depends on the relative sizes of  $|K^p|$  and  $|K^{p+1}|$ . In contrast,  $\mu_p^R$  is localized to the pair  $(K_i, K_l)$  and depends only on the  $(p+1)$ -simplices in the interval  $[i, l]$ , yielding the following corollary.

**Corollary 3.** *Given a filtration  $K_\bullet$  of size  $N = |K_\bullet|$  and a rectangle  $R = [i, j] \times [k, l]$  with indices  $0 \leq i < j \leq k < l \leq N$ , computing  $\mu_p^R$  using expression (2.12) requires  $O(R_{p+1}(m_{il}))$  time  $m_{il} = |K_l^{p+1}| - |K_i^{p+1}|$ .*

## A.2 Finite-precision arithmetic

It is well established in the literature that the Lanczos iteration, as given in its original form, it effectively useless in practice due to significant rounding and cancellation errors. Such errors manifest as loss of orthogonality between the computed Lanczos vectors, which drastically affects the convergence of the method. At first glance, this seems to be a simple numerical issue, however the analysis from Parlett [22] showed, loss of orthogonality is not merely the result of gradual accumulation of roundoff error—it is in fact is intricately connected to the convergence behavior of Lanczos iteration. One obvious remedy to this is to reorthogonalize the current Lanczos vectors  $\{q_{j-1}, q_j, q_{j+1}\}$  against all previous vectors using Householder matrices [17]—a the *complete reorthogonalization* scheme. This process

guarantees orthogonality to working precision, but incurs a cost of  $O(jn)$  for each Lanczos step, effectively placing the iteration back into the cubic time and quadratic memory regimes the direct methods exhibit. A variety of orthogonality enforcement schemes have been introduced over years, including implicit restart schemes, selective reorthogonalization, thick restarts, block methods, and so on; see [] for an overview.

### A.3 Laplacian Interpretation

In what follows we make a connection between boundary matrices and the graph Laplacian to illustrate how the Laplacian captures the “connectivity” aspects of the underlying simplicial complex.

**Example A.1** (Adapted from [21]). Suppose the vertices of  $G$  are ordered and labeled from 1 to  $n$  arbitrarily such that, given any subset  $X \subseteq V$ , we may define column vector  $x = (x_i)$  whose components  $x_i = 1$  indicate  $i \in X$  and  $x_i = 0$  otherwise. Given such a set  $X \subseteq V$ , let  $X' = V \setminus X$  denote its complement set. By  $L$ ’s definition, we have:

$$\begin{aligned} (Lx)_i &> 0 \iff i \in X \text{ and } |c_i(X)| = (Lx)_i \\ (Lx)_i &< 0 \iff i \in X' \text{ and } |c_i(X')| = |(Lx)_i| \\ (Lx)_i &= 0 \iff i \in X \cup X' \text{ and } c_i(X) = \emptyset \end{aligned}$$

where  $c_v(X) = \{(v, w) \in E \mid v \in X \text{ and } w \in V \setminus X\}$  denotes the *cutset* of  $X$  restricted to  $v$ , i.e. the set of edges having as one endpoint  $v \in X$  and another endpoint outside of  $X$ .

In other words, example A.1 demonstrates that  $L$  captures exactly how  $X$  is connected to the rest of  $G$ . Notice that if  $X = V$ , then  $Lx = 0$  and thus 0 must be an eigenvalue of  $L$  with an eigenvector pair  $\mathbf{1}$ . Like the adjacency matrix, the interpretation of the matrix-vector product has a natural extension to powers of  $L$ , wherein just as entries in  $A^k$  model paths, entries in  $L^k$  are seen to model boundaries [21].

### Parameterizing Settings

We include a few examples of potential application areas of work. Namely, we show a few promising examples of “parameterized settings” that may naturally benefit from our efforts here.

**Dynamic Metric Spaces:** Consider an  $\mathbb{R}$ -parameterized metric space  $\delta_X = (X, d_X(\cdot))$  where  $X$  is a finite set and  $d_X(\cdot) : \mathbb{R} \times X \times X \rightarrow \mathbb{R}_+$ , satisfying:

1. For every  $t \in \mathbb{R}$ ,  $\delta_X(t) = (X, d_X(t))$  is a pseudo-metric space<sup>5</sup>
2. For fixed  $x, x' \in X$ ,  $d_X(\cdot)(x, x') : \mathbb{R} \rightarrow \mathbb{R}_+$  is continuous.

---

<sup>5</sup>This is required so that if one can distinguish the two distinct points  $x, x' \in X$  in case  $d_X(t)(x, x') = 0$  at some  $t \in \mathbb{R}$ .



When the parameter  $t \in \mathbb{R}$  is interpreted as *time*, the above yields a natural characterization of a “time-varying” metric space. More generally, we refer to an  $\mathbb{R}^h$ -parameterized metric space as *dynamic metric space* (DMS). Such space have been studied more in-depth [] and have been shown...

**Rayleigh Ritz values** Though the Lanczos iterations may be used to obtain the full tridiagonalization  $A = QTQ^T$ , intermediate spectral information is readily available in  $T_j$ , for  $j < \text{rank}(A)$ . Diagonalizing  $T_j = Y\Theta Y^T$  yields value/vector pairs  $\{(\theta_1^{(j)}, y_1^{(j)}), \dots, (\theta_j^{(j)}, y_j^{(j)})\}$  satisfying  $w^T(Ay - \theta y) = 0$  for all  $w \in \mathcal{K}_j(A, q_1)$ , called *Ritz pairs*. The values  $\theta$  are called *Ritz values* and their associated vectors  $v = Qy$  in the range of  $Q$  are called *Ritz vectors*. From the Ritz perspective, the Lanczos iteration implicitly maintains two orthonormal basis for  $K_j(A, q_1)$ —a Lanczos basis  $Q$  and the Ritz basis  $Y$ :

$$A = QTQ^T = QY\Theta Y^T Q^T \iff AQY = QY\Theta$$

In principle, the Lanczos basis  $\{q_i\}_{i=1}^j$  changes each iteration, while the Ritz basis  $\{Qy_i^{(j)}\}_{i=1}^j$  changes after each subspace projection. The way in which the Ritz values approach the spectrum of  $A$  is well-studied [], as they are known to be Rayleigh-Ritz approximations of  $A$ ’s eigenpairs  $\Lambda(A) = \{(\lambda_1, v_1), \dots, (\lambda_j, v_j)\}$ , and they are collectively known to be optimal in the sense that  $T_k = B$  is the matrix that minimizes  $\|AQ_k - Q_k B\|_2$  over the space of all  $k \times k$  matrices. Moreover, Ritz values contain intrinsic information of the distance between  $\Lambda(T_j)$  and  $\Lambda(A)$ . To see this, note that:

$$\|Av_i^{(j)} - v_i^{(j)}\theta_i^{(j)}\| = \beta_i^{(j)} = \beta_{j+1} \cdot |\langle e_j, y_i^{(j)} \rangle| \quad (\text{A.3})$$

Thus, we need not necessarily keep the Lanczos vectors  $Q$  in memory to monitor how close the spectra of the  $T_j$ ’s approximate  $\Lambda(A)$ . In fact, it is known that the Ritz values  $\{\theta_1^{(1)}, \theta_1^{(2)}, \dots, \theta_1^{(j)}\}$  of  $T_j$  satisfy:

$$|\lambda - \theta_i^{(j)}| \leq (\beta_i^{(j)})^2 / (\min_{\mu} |\mu - \theta_i^{(j)}|) \quad (\text{A.4})$$

The full convergence of the Ritz values to the eigenvalues of  $A$  is known to converge at a rate that depends on the ratio between  $\lambda_1/\lambda_n$ . A full analysis is done in terms of Chebychev Polynomials in [17]. In practice, it has been observed that the Lanczos iteration converges super-linearly towards the extremal eigenvalues of the spectrum, whereas for interior eigenvalues one typically must apply a shifting scheme.

## Convergence Rate

The ability of the Krylov subspace iteration to capture the extremal portions of the spectrum remains unparalleled, and by using  $O(n)$  memory, the Lanczos iteration uses optimal memory. As mentioned in section ??, when the computation is carried out in finite-precision arithmetic, one may observe loss of orthogonality in the Lanczos vectors. Fortunately, the connection between the Lanczos method and the Rayleigh quotient ensures *eventual* termination of the procedure under by restarting the Lanczos method, and continue with the iteration until the spectrum has been approximated to some prescribed accuracy. Unfortunately, if the number of iterations  $k$  is e.g. larger than  $n^2$ , then the method may approach

to  $O(r \max(\mathcal{M}(n), n), n) \approx O(n^3)$  complexity one starts with. If the supplied matrix-vector product operation is fast, the number of iterations  $k$  needed for convergence of the Lanczos method becomes the main bottleneck estimating the spectrum of  $A$ .

Loss of orthogonality can be mitigated by re-orthogonalizing against all previous Lanczos vectors, but this increases the Lanczos complexity to  $\approx O(n^2)$  per iteration. Thus, the goal is strike a balance: find a way to keep all  $n$  Lanczos numerically orthonormal, so as to ensure super-linear convergence of the Ritz values  $\theta$ , but do so using  $c \cdot n$  memory, where  $c$  is a relatively small constant.

Since rates of convergence  $\alpha$  increases the number of correct digits by an exponential rate with factor *alpha*, any super-linear convergent ( $\alpha > 1$ ) method needs at most  $c$  terms to approximate an eigen-pair up to numerical precision. In the context of the Lanczos method, achieving even quadratic convergence would imply the number of iterations needed to obtain machine-precision is bounded by  $T(c \cdot \mathcal{M}(n) \cdot r)$ , where  $c$  is a small constant. We say that a method which achieves *superlinear* convergence has complexity *essentially*  $O(c \cdot n) \approx O(n)$ .

Among the more powerful methods for achieving super linear convergence towards a given eigenvalue  $\lambda$  is the Jacobi-Davidson method. This method seeks to correct:

Solving for  $t$  results in the *correction equation*

$$(I - uu^T)(A - \sigma I)(I - uu^T)t = \theta u - Au \quad (\text{A.5})$$

where, since  $u$  is unit-norm,  $I - uu^T$  is a projector onto the complement of  $\text{span}(u)$ . It's been shown that solving exactly for this correction term essentially constructs an cubically-convergent sequence towards some  $\theta \mapsto \lambda$  in the vicinity of  $\sigma$ . Solving for the correction equation exactly is too expensive, sparking efforts to approximate it. It turns out that, just as the Lanczos method in exact arithmetic is highly related to the conjugate gradient method for solving linear systems, solving for the correction equation exactly is in some ways conceptually similar to making an Newton step in the famous Newtons method from nonlinear optimization. Since (??) is approximated, the JD method is often called in the literature akin to making an “inexact newton step” [1].

The JD method with inexact Newton steps yields an individual eigenvalue estimate with quadratic convergence—*essentially*  $O(m)$  time after some constant number matrix-vector products and  $O(n)$  memory. The Lanczos method, in contrast, estimates all eigenvalues in essentially quadratic time if the convergence rate is superlinear. Pairing these two methods is a non-trivial endeavor. In a sequence of papers, Stathopoulos et al [2] investigated various strategies for approximately solving the correction equation. In [3], they give both theoretical and empirical evidence to suggest that by employing generalized Davidson and Jacobi-Davidson like solvers within an overarching Lanczos paradigm, they achieve nearly optimal methods for estimating large portions of the spectrum using  $O(1)$  number of basis vectors. By approximating the inner iterations with the symmetric Quasi-Minimal Residual (QMR) method, they argue that JD cannot converge more than three times slower than the optimal method, and empirically they find the constant factor to be less than 2.

## A.4 Proofs

### Proof of rank equivalence

In general, it is not true that  $\text{rank}(A) = \text{rank}(\text{sgn}(A))$ . However, it is true that  $\text{rank}(\partial_p) = \text{rank}(\text{sgn}(\partial_p))$ .

### Proof of Lemma 1

*Proof.* The Pairing Uniqueness Lemma [12] asserts that if  $R = \partial V$  is a decomposition of the total  $m \times m$  boundary matrix  $\partial$ , then for any  $1 \leq i < j \leq m$  we have  $\text{low}_R[j] = i$  if and only if  $r_\partial(i, j) = 1$ . As a result, for  $1 \leq i < j \leq m$ , we have:

$$\text{low}_R[j] = i \iff r_R(i, j) \neq 0 \iff r_\partial(i, j) \neq 0 \quad (\text{A.6})$$

Extending this result to equation (2.10) can be seen by observing that in the decomposition,  $R = \partial V$ , the matrix  $V$  is full-rank and obtained from the identity matrix  $I$  via a sequence of rank-preserving (elementary) left-to-right column additions.  $\square$

### Proof of Proposition 1

*Proof.* We first need to show that  $\beta_p^{i,j}$  can be expressed as a sum of rank functions. Note that by the rank-nullity theorem, so we may rewrite (2.7) as:

$$\beta_p^{i,j} = \dim(C_p(K_i)) - \dim(B_{p-1}(K_i)) - \dim(Z_p(K_i) \cap B_p(K_j))$$

The dimensions of groups  $C_p(K_i)$  and  $B_p(K_i)$  are given directly by the ranks of diagonal and boundary matrices, yielding:

$$\beta_p^{i,j} = \text{rank}(I_p^{1,i}) - \text{rank}(\partial_p^{1,i}) - \dim(Z_p(K_i) \cap B_p(K_j))$$

To express the intersection term, note that we need to find a way to express the number of  $p$ -cycles born at or before index  $i$  that became boundaries before index  $j$ . Observe that the non-zero columns of  $R_{p+1}$  with index at most  $j$  span  $B_p(K_j)$ , i.e.  $\{\text{col}_{R_{p+1}[k]} \neq 0 \mid k \in [j]\} \in \text{Im}(\partial_{p+1}^{1,j})$ . Now, since the low entries of the non-zero columns of  $R_{p+1}$  are unique, we have:

$$\dim(Z_p(K_i) \cap B_p(K_j)) = |\Gamma_p^{i,j}| \quad (\text{A.7})$$

where  $\Gamma_p^{i,j} = \{\text{col}_{R_{p+1}[k]} \neq 0 \mid k \in [j], 1 \leq \text{low}_{R_{p+1}}[k] \leq i\}$ . Consider the complementary matrix  $\bar{\Gamma}_p^{i,j}$ , given by the non-zero columns of  $R_{p+1}$  with index at most  $j$  that are not in  $\Gamma_p^{i,j}$ , i.e. the columns satisfying  $\text{low}_{R_{p+1}}[k] > i$ . Combining rank-nullity with the observation above, we have:

$$|\bar{\Gamma}_p^{i,j}| = \dim(B_p(K_j)) - |\Gamma_p^{i,j}| = \text{rank}(R_{p+1}^{i+1,j}) \quad (\text{A.8})$$

Combining equations (A.7) and (A.8) yields:

$$\dim(Z_p(K_i) \cap B_p(K_j)) = |\Gamma_p^{i,j}| = \dim(B_p(K_j)) - |\bar{\Gamma}_p^{i,j}| = \text{rank}(R_{p+1}^{1,j}) - \text{rank}(R_{p+1}^{i+1,j}) \quad (\text{A.9})$$

Observing the final matrices in (A.9) are *lower-left* submatrices of  $R_{p+1}$ , the final expression (2.11) follows by applying Lemma 1 repeatedly.  $\square$

## Proof of boundary matrix properties

*Proof.* First, consider property (1). For any  $t \in T$ , applying the boundary operator  $\partial_p$  to  $K_t = \text{Rips}_\epsilon(\delta_X(t))$  with non-zero entries satisfying (??) by definition yields a matrix  $\partial_p$  satisfying  $\text{rank}(\partial_p) = \dim(B_{p-1}(K_t))$ . In contrast, definition (1) always produces  $p$ -boundary matrices of  $\Delta_n$ ; however, notice that the only entries which are non-zero are precisely those whose simplices  $\sigma$  that satisfy  $\text{diam}(\sigma) < \epsilon$ . Thus,  $\text{rank}(\partial_p^t) = \dim(B_{p-1}(K_t))$  for all  $t \in T$ . < (show proof of (2))> Property (3) follows from the construction of  $\partial_p$  and from the inequality  $\|A\|_2 \leq \sqrt{m}\|A\|_1$  for an  $n \times m$  matrix  $A$ , as  $\|\partial_p^t\|_1 \leq (p+1)\epsilon$  for all  $t \in T$ . □

## References

- [1] Ulrich Bauer. Ripser: efficient computation of vietoris–rips persistence barcodes. *Journal of Applied and Computational Topology*, 5(3):391–423, 2021.
- [2] Ulrich Bauer and Michael Lesnick. Persistence diagrams as diagrams: A categorification of the stability theorem. In *Topological Data Analysis*, pages 67–96. Springer, 2020.
- [3] Ulrich Bauer, Talha Bin Masood, Barbara Giunti, Guillaume Houry, Michael Kerber, and Abhishek Rathod. Keeping it sparse: Computing persistent homology revised. *arXiv preprint arXiv:2211.09075*, 2022.
- [4] Rajendra Bhatia. *Matrix analysis*, volume 169. Springer Science & Business Media, 2013.
- [5] Shujun Bi, Le Han, and Shaohua Pan. Approximation of rank function and its application to the nearest low-rank correlation matrix. *Journal of Global Optimization*, 57(4):1113–1137, 2013.
- [6] Andrea Cerri, Barbara Di Fabio, Massimo Ferri, Patrizio Frosini, and Claudia Landi. Betti numbers in multidimensional persistent homology are stable functions. *Mathematical Methods in the Applied Sciences*, 36(12):1543–1557, 2013.
- [7] Frédéric Chazal, Vin De Silva, Marc Glisse, and Steve Oudot. *The structure and stability of persistence modules*, volume 10. Springer, 2016.
- [8] Chao Chen and Michael Kerber. An output-sensitive algorithm for persistent homology. In *Proceedings of the twenty-seventh annual symposium on Computational geometry*, pages 207–216, 2011.
- [9] Fan RK Chung. *Spectral graph theory*, volume 92. American Mathematical Soc., 1997.
- [10] David Cohen-Steiner, Herbert Edelsbrunner, and John Harer. Stability of persistence diagrams. In *Proceedings of the twenty-first annual symposium on Computational geometry*, pages 263–271, 2005.

- [11] David Cohen-Steiner, Herbert Edelsbrunner, and Dmitriy Morozov. Vines and vineyards by updating persistence in linear time. In *Proceedings of the twenty-second annual symposium on Computational geometry*, pages 119–126, 2006.
- [12] Tamal Krishna Dey and Yusu Wang. *Computational topology for data analysis*. Cambridge University Press, 2022.
- [13] Chao Ding, Defeng Sun, Jie Sun, and Kim-Chuan Toh. Spectral operators of matrices. *Mathematical Programming*, 168(1):509–531, 2018.
- [14] Herbert Edelsbrunner and John L Harer. *Computational topology: an introduction*. American Mathematical Society, 2022.
- [15] Herbert Edelsbrunner, David Letscher, and Afra Zomorodian. Topological persistence and simplification. In *Proceedings 41st annual symposium on foundations of computer science*, pages 454–463. IEEE, 2000.
- [16] Timothy E Goldberg. Combinatorial laplacians of simplicial complexes. *Senior Thesis, Bard College*, 6, 2002.
- [17] Gene H Golub and Charles F Van Loan. *Matrix computations*. JHU press, 2013.
- [18] Ming Gu and Stanley C Eisenstat. A divide-and-conquer algorithm for the symmetric tridiagonal eigenproblem. *SIAM Journal on Matrix Analysis and Applications*, 16(1):172–191, 1995.
- [19] Cornelius Lanczos. An iteration method for the solution of the eigenvalue problem of linear differential and integral operators. 1950.
- [20] Olvi Mangasarian and Chunhui Chen. A class of smoothing functions for nonlinear and mixed complementarity problems. Technical report, 1994.
- [21] Michael William Newman. The laplacian spectrum of graphs. Master’s thesis, 2001.
- [22] Beresford N Parlett. Do we fully understand the symmetric lanczos algorithm yet. *Brown et al*, 3:93–107, 1994.
- [23] Horst D Simon. Analysis of the symmetric lanczos algorithm with reorthogonalization methods. *Linear algebra and its applications*, 61:101–131, 1984.
- [24] Afra Zomorodian and Gunnar Carlsson. Computing persistent homology. In *Proceedings of the twentieth annual symposium on Computational geometry*, pages 347–356, 2004.

## A Boundary matrix factorization

**Definition 4** (Boundary matrix decomposition). Given a filtration  $K_\bullet$  with  $m$  simplices, let  $\partial$  denote its  $m \times m$  filtered boundary matrix. We call the factorization  $R = \partial V$  the *boundary matrix decomposition* of  $\partial$  if:

I1.  $V$  is full-rank upper-triangular

I2.  $R$  satisfies  $\text{low}_R[i] \neq \text{low}_R[j]$  iff its  $i$ -th and  $j$ -th columns are nonzero

where  $\text{low}_R(i)$  denotes the row index of lowest non-zero entry of column  $i$  in  $R$  or null if it doesn't exist. Any matrix  $R$  satisfying property (I2) is said to be *reduced*; that is, no two columns share the same low-row indices.

## B Laplacian facts

In general, the spectrum of the graph Laplacian  $L$  is unbounded, [] and instead many prefer to work within the “normalized” setting where eigenvalues are bounded. The *normalized Laplacian*  $\mathcal{L}$  of a graph  $G$  is typically given as:

$$\mathcal{L}(G) = D^{-1/2} L D^{-1/2} \quad (\text{B.1})$$

with the convention that  $D^{-1}(v_i, v_i) = 0$  for  $\deg(v_i) = 0$ . The variational characterization of eigenvalues in terms of the Rayleigh quotient of  $\mathcal{L}$  convey a particular form. Specifically, for any real-valued function  $f : V \rightarrow \mathbb{R}$  on  $G$ , when viewed as a column vector,  $\mathcal{L}$  satisfies:

$$\frac{\langle f, \mathcal{L}f \rangle}{\langle f, f \rangle} = \frac{\sum_{i \sim j} (g(v_i) - g(v_j))^2}{\sum_i g(v_i)^2 \cdot \deg(v_i)} \quad (\text{B.2})$$

where  $f = D^{1/2}g$  and  $\langle f, g \rangle$  denotes the standard inner product in  $\mathbb{R}^n$ . Equation (B.2) may be used to show that the spectrum  $\Lambda(\mathcal{L})$  is bounded in the interval  $[0, 2]$ . In particular, it is known that:

$$\lambda_i \leq \sup_f \frac{\langle f, \mathcal{L}f \rangle}{\langle f, f \rangle} \leq 2 \quad (\text{B.3})$$

Recall that, when  $G$  is connected, 0 is an eigenvalue of both  $L$  and  $\mathcal{L}(G)$ , with multiplicity  $\text{cc}(G)$ . Moreover, if  $G$  is the union of disjoint graphs  $G_1, G_2, \dots, G_k$ , then it has as its spectrum the union of the spectra  $\Lambda(G_1), \Lambda(G_2), \dots, \Lambda(G_k)$ . Certain parts of the spectrum of  $\mathcal{L}$  can be deduced explicitly for very structured types of  $G$ , such as complete graphs, complete bipartite graphs, star graphs, path graphs, and cycle graphs, and  $n$ -cubes. For a list of additional properties the graph and normalized Laplacians satisfy, including bounds on eigenvalues, relation to random walks and rapidly-mixing Markov chains, identities tied to isoperimetric properties of graphs, and explicit connections to spectral Riemannian geometry, see [9] and references within.

## C Laplacian matvec products

Below is pseudocode outlining how to evaluate a weighted (up) Laplacian matrix-vector multiplication built from a simplicial complex  $K$  with  $m = |K^{p+1}|$  and  $n = |K^p|$  in essentially  $O(m)$  time when  $m > n$  and  $p$  is considered a small constant. Key to the runtime of the operation being essentially linear is the constant-time determination of orientation between  $p$ -faces  $(s_{\tau, \tau'})$ —which can be inlined during the computation—and the use of a deterministic

$O(1)$  hash table  $h : K^p \rightarrow [n]$  for efficiently determining the appropriate input/output offsets to modify ( $i$  and  $j$ ). Note the degree computation occurs only once.

---

**Algorithm 1** `matvec` for weighted  $p$  up-Laplacians in  $O(m(p+1)) \approx O(m)$  time ( $p \geq 0$ )

---

**Require:** Fixed oriented complex  $K$  of size  $N = |K|$

**Optional:** Weight functions  $w_{p+1} : K^{p+1} \rightarrow \mathbb{R}_+$  and  $w_p^l, w_p^r : K^p \rightarrow \mathbb{R}_+$

```

1: # Precompute weighted degrees  $\deg_w$ 
2:  $h : K^p \rightarrow [n]$ 
3:  $\deg_w \leftarrow \mathbf{0}$ 
4: for  $\sigma \in K^{p+1}$  do:
5:   for  $\tau \in \partial[\sigma]$  do:
6:      $\deg_w[h(\tau)] \leftarrow \deg_w[h(\tau)] + w_p^l(\tau) \cdot w_{p+1}(\sigma) \cdot w_p^r(\tau)$ 
7:
8: # Compute Laplacian matvec for  $x$ 
9: function UPLAPLACIANMATVEC( $x \in \mathbb{R}^n$ )
10:   $y \leftarrow \deg_w \odot x$  (element-wise product)
11:  for  $\sigma \in K^{p+1}$  do:
12:    for  $\tau, \tau' \in \partial[\sigma] \times \partial[\sigma]$  where  $\tau \neq \tau'$  do:
13:       $s_{\tau, \tau'} \leftarrow \text{sgn}([\tau], \partial[\sigma]) \cdot \text{sgn}([\tau'], \partial[\sigma])$ 
14:       $i, j \leftarrow h(\tau), h(\tau')$ 
15:       $y_i \leftarrow y_i + s_{\tau, \tau'} \cdot x_j \cdot w_p^l(\tau) \cdot w_{p+1}(\sigma) \cdot w_p^r(\tau')$ 
return  $y$ 

```

$\triangleright (y = \langle L_p^{\text{up}}, x \rangle)$

---

## D Parameterized setting & Perturbation theory

If  $f$  is a real-valued filter function that varies smoothly in  $\mathcal{H}$ , one would expect the spectra of the constitutive terms in  $\beta_p^*$  and  $\mu_p^*$  to also vary smoothly as functions of  $\mathcal{H}$ . Indeed, since Laplacian matrices are normal matrices, we expect their spectra to be quite stable under perturbations  $\square$ .

Small condition numbers often improve the convergence of iterative solvers and improve stability of spectrum with respect to perturbations in the entries of the matrix.  $\kappa(M^{-1}A)$

$$M^{-1}Ax = M^{-1}b$$

where  $M$  is symmetric positive definite.

$$\min_{x \perp \mathbf{1}} \frac{1}{2} x^T (L + \epsilon I_n) x - b^T x \quad (\text{D.1})$$

Since this nonsingular, positive definite, strictly diagonally dominant matrix, thus we may apply the famous Conjugate Gradient (CG) algorithm to solve such a system. It's well known that CG converges to the solution of  $Ax = b$  in exactly  $O(n)$  iterations (and often much earlier), of which each iteration requires one  $O(m)$  matrix-vector product, implying a runtime of  $O(mn^2)$  (compare with...). Moreover, and since this is a Laplacian matrix, the wealth of tools developed for said matrices may also be used. In particular,  $\square$  showed

that *low-stretch spanning trees* act as good preconditioners to accelerate Laplacian solvers, wherein it's been shown that the preconditioned Conjugate Gradient (PCG) requires at most  $O(\sqrt{m} \log n)$  iterations, each of which requires one matrix-vector product using  $L_G$  and in  $O(m^{1/3} \log n \ln 1/\epsilon)$  iterations. This was later improved by, who showed that one can solve Laplacian systems effectively in  $O(m \log^{O(1)} n)$  time, giving a bound of  $O(rm \log^{O(1)} n)$  time to obtain....

Of course, if one wants to compute either of the counting invariants in... exactly for  $p = 0$ , of course, the fastest algorithm is to reduce the problem to the well-known elder-rule problem, which takes  $O(m \log m + m\alpha(n))$  time for a general filtration. It is unlikely that we may beat this bound, either in theory or in practice, for  $p = 0$ . However, the fastest known algorithm for computing the full persistence diagram for  $p \geq 1$  is  $O()$ , which is quite a jump in complexity; there is no generalization of disjoint-set algorithm for the case where  $p \geq 1$ . Moreover, these direct methods tend to be memory bound operations, pushing researchers who want to compute these diagrams in practice to focus on ways of reducing the memory usage, such as using  $\mathbb{Z}_2$  field coefficients. In contrast, the means by which we compute these invariants scales quite well with larger  $p$ , it produces a stronger invariant, and is far more reaching to other areas of mathematics.