Computing Multidimensional Persistence*

Gunnar Carlsson[1], Gurjeet Singh[1], and Afra Zomorodian[2]

[1] Department of Mathematics, Stanford University, USA.
[2] Department of Computer Science, Dartmouth College, USA.
[To appear in ISAAC '09]

Abstract. The theory of multidimensional persistence captures the topology of a multifiltration – a multiparameter family of increasing spaces. Multifiltrations arise naturally in the topological analysis of scientific data. In this paper, we give a polynomial time algorithm for computing multidimensional persistence.

1 Introduction

In this paper, we give a polynomial time algorithm for computing the persistent homology of a multifiltration. The computed solution is compact and complete, but not an invariant. Theoretically, this is the best one may hope for since complete compact invariants do not exist for multidimensional persistence [1].

1.1 Motivation

Intuitively, a multifiltration models a growing space that is parameterized along multiple dimensions. For example, the complex with coordinate $(3, 2)$ in Figure 1 is filtered along the horizontal and vertical dimensions, giving rise to a bifiltration. Multifiltrations arise naturally in topological analysis of scientific data. Often, scientific data is in the form of a finite set of noisy samples from some underlying topological space. Our goal is to robustly recover the lost connectivity of the underlying space. If the sampling is dense enough, we approximate the space as a union of balls by placing ϵ-balls around each point. As we increase ϵ, we obtain a growing family of spaces, a one-parameter multifiltration also called a filtration. This approximation is the central idea behind many methods for computing the topology of a point set, such as *Čech*, *Rips-Vietoris* [2], or *witness* [3] complexes. Often, the input point set is also filtered through multiple functions. We now have multiple dimensions along which our space is filtered. That is, we have a multifiltration.

* The authors were partially supported by the following grants: G. C. by NSF DMS-0354543; A. Z. by DARPA HR 0011-06-1-0038, ONR N 00014-08-1-0908, and NSF CCF-0845716; all by DARPA HR 0011-05-1-0007.

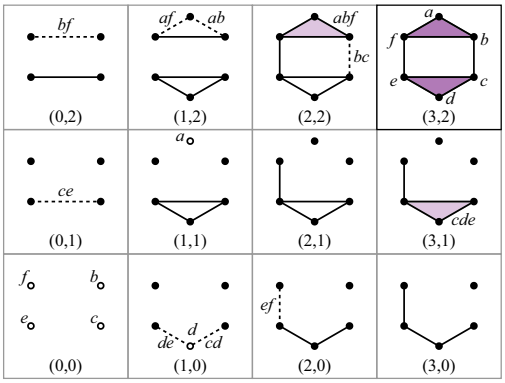

Fig. 1. A bifiltration. The complex with labeled vertices is at coordinate $(3, 2)$. Simplices are highlighted and named at the critical coordinates that they appear.

1.2 Prior Work

For one-dimensional filtrations, the theory of persistent homology provides a complete invariant called a *barcode*, a multiset of intervals [4]. Each interval in the barcode corresponds to the lifetime of a single topological attribute within the filtration. Since features have long lives, while noise is short-lived, a quick examination of the intervals gives a robust estimation of the topology. The existence of a complete compact invariant, as well as efficient algorithms and fast implementations have led to successful application of persistence to a variety of problems, such as shape description [5], denoising volumetric density data [6], detecting holes in sensor networks [7], analyzing neural activity in the visual cortex [8], and analyzing the structure of natural images [9], to name a few.

For multifiltrations of dimension higher than one, the situation is much more complicated. The theory of *multidimensional persistence* shows that no complete discrete invariant exists, where *discrete* means that the structure of the target for the invariant does not depend on the structure of the underlying field [1]. Instead, the authors propose an incomplete invariant, the rank invariant, which captures important persistent information. Unfortunately, this invariant is not compact, requiring large storage, so its direct computation using the one-dimensional algorithm is not feasible. A variant of the problem of multidimensional persistence has appeared in computer vision [10]. A partial solution, called *vineyards*, has been offered [11]. A full solution, however, has not been attempted by any prior work.

1.3 Contributions

In this paper, we provide a complete solution to the problem of computing multidimensional persistence. We recast persistence as a problem within computational algebraic geometry, allowing us to utilize powerful algorithms from

this area. We then exploit the structure provided by a multifiltration to greatly simplify the algorithms. Finally, we show that the resulting algorithms are polynomial time, unlike their original counterparts, which are EXPSPACE-complete, requiring exponential space and time. We begin with a brief review of necessary concepts in Section 2, and recast the problem into an algebraic geometric framework. Section 3 contains the main contribution of this paper, where we use the structure of multifiltrations to simplify the traditional algorithms.

2 Background & Approach

In this section, we review concepts from algebraic topology and computational algebraic geometry. We then present our approach of computing multidimensional persistence using algorithms from the latter area. Due to lack of space, we omit significant portions of our work, referring the interested reader to our manuscript for a complete description [12]. Our treatment of algebraic geometry and its algorithms follow Chapter 5 of Cox, Little, and O'Shea [13].

Our goal is the computation of the persistent homology of a multifiltration. Let $\mathbb{N} \subseteq \mathbb{Z}$ be the set of non-negative integers. A topological space X is *multifiltered* if we are given a family of subspaces $\{X_u\}_u$, where $u \in \mathbb{N}^n$ and $X_u \subseteq X$ such that for $u, w_1, w_2, v \in \mathbb{N}^n$, the diagrams

$$\begin{array}{ccc} X_u & \longrightarrow & X_{w_1} \\ \downarrow & & \downarrow \\ X_{w_2} & \longrightarrow & X_v \end{array} \tag{1}$$

commute whenever $u \leq w_1, w_2 \leq v$. We call the family of subspaces $\{X_u\}_u$ a *multifiltration*, such as the example in Figure 1. In this paper, we assume our input is a multifiltered simplicial complex that has the following property:

Definition 1 (one-critical). *A multifiltered complex where each cell has a unique minimal critical grade at which it enters the complex is* one-critical.

The bifiltration in Figure 1 is one-critical, as are most multifiltrations that arise in practice [12].

Given a simplicial complex K, we may define *chain groups* C_i as the free Abelian groups on oriented i-simplices. The *boundary operator* $\partial_i \colon C_i \to C_{i-1}$ connects the chain groups into a *chain complex* C_*:

$$\cdots \to C_{i+1} \xrightarrow{\partial_{i+1}} C_i \xrightarrow{\partial_i} C_{i-1} \to \cdots . \tag{2}$$

Given any chain complex, the *ith homology group* is

$$H_i = \ker \partial_i / \operatorname{im} \partial_{i+1}. \tag{3}$$

Given a multifiltration $\{X_u\}_u$, for each pair $u \leq v \in \mathbb{N}^n$, $X_u \subseteq X_v$ by definition, so $X_u \hookrightarrow X_v$, inducing a map $\iota_i(u, v)$ at the homology level $H_i(X_u) \to H_i(X_v)$

that maps a homology class in X_u to the one that contains it in X_v. The *ith persistent homology* is the image of ι_i for all pairs $u \leq v$.

Our work rests on the theory of persistence [4, 1]. The key insight is this: Persistent homology of a multifiltration is standard homology of a single multigraded module that encodes the multifiltration using polynomial coefficients. Let $A^n = k[x_1, \ldots, x_n]$ be the n-graded *polynomial ring*, graded by $A^n_v = kx^v, v \in \mathbb{N}^n$. We define an n-graded module over this ring as follows.

Definition 2 (chain module). *Given a multifiltered simplicial complex* $\{K_u\}_u$*, the* ith chain module *is the* n*-graded module over the graded polynomial ring* A^n

$$C_i = \bigoplus_u C_i(K_u), \tag{4}$$

where the k*-module structure is the direct sum structure and* $x^{v-u} \colon C_i(K_u) \to C_i(K_v)$ *is the inclusion* $K_u \hookrightarrow K_v$*.*

These graded chain modules C_i are finitely generated, and for one-critical filtrations, they are also free, so we may choose bases for them.

Definition 3 (standard basis). *The* standard basis *for the* i^{th} *chain module* C_i *is the set of* i*-simplices in critical grades.*

Given standard bases, we may write the boundary operator $\partial_i \colon C_i \to C_{i-1}$ explicitly as a matrix with polynomial entries. We now have a new n-graded chain complex (2) that encodes the multifiltration. The homology of this chain complex is the persistent homology of the multifiltration [1]. By definition (3), we may compute homology in three steps:

1. Compute $\operatorname{im} \partial_{i+1}$: This is a submodule of the polynomial module, and its computation is the *submodule membership problem* in computational algebraic geometry. We may solve this problem by computing the *reduced Gröbner basis* using the Buchberger and reduction algorithms, and then dividing using the Divide algorithm.
2. Compute $\ker \partial_i$: The is the *(first) syzygy module*, which we may compute using *Schreyer's algorithm.*
3. Compute H_i: This task is simple, once the above two tasks are complete. We need to test whether the generators of the syzygy submodule are in the boundary submodule, a task which may be completed using the tools above.

While the above algorithms solve the membership problem, they have not been used in practice due to their complexity. The submodule membership problem is a generalization of the *Polynomial Ideal Membership Problem (PIMP)* which is Expspace-complete, requiring exponential space and time [14, 15]. Indeed, the Buchberger algorithm, in its original form is doubly-exponential. Therefore, while our reformulation of multidimensional persistence gives us algorithms, we need to make them faster to make this approach feasible.

3 Multigraded Algorithms

In this section, we show that multifiltrations provide additional structure that may be exploited to simplify the algorithms for our three tasks. These simplifications convert these intractable algorithms into polynomial time algorithms.

3.1 Exploiting Homogeneity

The key property that we exploit for simplification is homogeneity.

Definition 4 (homogeneous). *Let M be an $m \times n$ matrix with monomial entries. The matrix M is* homogeneous *iff*

1. *every column $\mathbf{f}$ of M is associated with a coordinate in the multifiltration ($u_{\mathbf{f}}$) and thus a corresponding monomial $x^{u_{\mathbf{f}}}$,*
2. *every non-zero element M_{jk} may be expressed as the quotient of the monomials associated with column k and row j, respectively.*

Any vector $\mathbf{f}$ endowed with a coordinate $u_{\mathbf{f}}$ that may be written as above is homogeneous*, e.g. the columns of M.*

We will show that (1) all boundary matrices ∂_i may be written as homogeneous matrices initially, and (2) the algorithms for computing persistence only produce homogeneous matrices and vectors. That is, we maintain homogeneity as an invariant throughout the computation. We begin with our first task.

Lemma 1. *For a one-critical multifiltration, the matrix of $\partial_i \colon C_i \to C_{i-1}$ written in terms of the standard bases is homogeneous.*

Proof. Recall that we may write the boundary operator $\partial_i \colon C_i \to C_{i-1}$ explicitly as a $m_{i-1} \times m_i$ matrix M in terms of the standard bases for C_i and C_{i-1}, as shown in matrix (Equation 5) for ∂_1. From Definition 3, the standard basis for C_i is the set of i-simplices in critical grades. In a one-critical multifiltration, each simplex σ has a unique critical coordinate u_σ (Definition (1)). In turn, we may represent this coordinate by the monomial x^{u_σ}. For instance, simplex a in Figure 1 has critical grade $(1,1)$ and monomial $x^{(1,1)} = x_1x_2$. We order these monomials using $>_{\text{lex}}$ and use this ordering to rewrite the matrix for ∂_i. The matrix entry M_{jk} relates σ_k, the kth basis element for C_i to $\hat{\sigma}_j$, the jth basis element for C_{i-1}. If $\hat{\sigma}_j$ is not a face of σ_k, then $M_{jk} = 0$. Otherwise, $\hat{\sigma}_j$ is a face of σ_k. Since a face must precede a co-face in a multifiltration, $u_{\sigma_k} >_{\text{lex}} u_{\hat{\sigma}_j} \implies x^{u_{\sigma_k}} >_{\text{lex}} x^{u_{\hat{\sigma}_j}}$, and $M_{jk} = x^{u_{\sigma_k}}/x^{u_{\hat{\sigma}_j}} = x^{u_{\sigma_k} - u_{\hat{\sigma}_j}}$. That is, the matrix is homogeneous.

For example, $\hat{\sigma}_1 = a$ is a face of $\sigma_1 = ab$, so $M_{11} = x_1x_2^2/x_1x_2 = x_2$ in the matrix for ∂_1 for the bifiltration in Figure 1.

Corollary 1. *For a one-critical multifiltration, the boundary matrix ∂_i in terms of the standard bases has monomial entries.*

Proof. The result is immediate from the proof of the previous lemma. The matrix entry is either 0, a monomial, or $x^{u(\sigma_k)-u(\hat{\sigma}_j)}$, a monomial.

Below, we show the homogeneous matrix for ∂_1 for the bifiltration in Figure 1, where we augment the matrix with the associated monomials. We assume we are computing over $\mathbb{Z}_2$.

$$\left[\begin{array}{cc|cccccccc} & & ab & bc & cd & de & ef & af & bf & ce \\ & & x_1x_2^2 & x_1^2x_2^2 & x_1 & x_1 & x_1^2 & x_1x_2^2 & x_2^2 & x_2 \\ \hline a & x_1x_2 & x_2 & 0 & 0 & 0 & 0 & x_2 & 0 & 0 \\ d & x_1 & 0 & 0 & 1 & 1 & 0 & 0 & 0 & 0 \\ b & 1 & x_1x_2^2 & x_1^2x_2^2 & 0 & 0 & 0 & 0 & x_2^2 & 0 \\ c & 1 & 0 & x_1^2x_2^2 & x_1 & 0 & 0 & 0 & 0 & x_2 \\ e & 1 & 0 & 0 & 0 & x_1 & x_1^2 & 0 & 0 & x_2 \\ f & 1 & 0 & 0 & 0 & 0 & x_1^2 & x_1x_2^2 & x_2^2 & 0 \end{array}\right] \tag{5}$$

We next focus on our second task, showing that given a homogeneous matrix as input, the algorithms produce homogeneous vectors and matrices. Let F be an $m \times n$ homogeneous matrix. Let $\{\mathbf{e_1}, \ldots, \mathbf{e_m}\}$ and $\{\hat{\mathbf{e}}_\mathbf{1}, \ldots, \hat{\mathbf{e}}_\mathbf{n}\}$ be the standard bases for the graded polynomial rings R^m and R^n, respectively. A homogeneous matrix associates a coordinate and monomial to the row and column basis elements. For example, since x_1 is the monomial for row 2 of matrix (5), we have $u_{\mathbf{e_2}} = (1, 0)$ and $x^{u_{\mathbf{e_2}}} = x_1$. Each column $\mathbf{f}$ in F is homogeneous and may be written in terms of rows:

$$\mathbf{f} = \sum_{i=1}^{m} c_i \frac{x^{u_\mathbf{f}}}{x^{u_{\mathbf{e_i}}}} \mathbf{e_i}, \tag{6}$$

where $c_i \in k$ and we allow $c_i = 0$ when a row is not used. For example, column $\mathbf{g}$ for edge ab in our bifiltration may be written as:

$$\mathbf{g} = x_2\mathbf{e_1} + x_2x_2^2\mathbf{e_3} = \frac{x_2x_2^2}{x_1x_2}\mathbf{e_1} + \frac{x_2x_2^2}{1}\mathbf{e_3} = \frac{x^{u_\mathbf{g}}}{x^{u_{\mathbf{e_1}}}}\mathbf{e_1} + \frac{x^{u_\mathbf{g}}}{x^{u_{\mathbf{e_3}}}}\mathbf{e_3} = \sum_{i\in\{1,3\}} \frac{x^{u_\mathbf{g}}}{x^{u_{\mathbf{e_i}}}}\mathbf{e_i}.$$

Consider the BUCHBERGER algorithm [13]. The algorithm repeatedly computes S-polynomials of homogeneous vectors.

Lemma 2. *The S-polynomial $S(\mathbf{f}, \mathbf{g})$ of homogeneous vectors $\mathbf{f}$ and $\mathbf{g}$ is homogeneous.*

Proof. A zero S-polynomial is trivially homogeneous. A non-zero S-polynomial $S(\mathbf{f}, \mathbf{g})$ implies that $\mathbf{h} = \text{LCM}(\text{LM}(\mathbf{f}), \text{LM}(\mathbf{g}))$ is non-zero. By the definition of LCM, the leading monomials of $\mathbf{f}$ and $\mathbf{g}$ contain the same basis element $\mathbf{e_j}$. We have, $\text{LM}(\mathbf{f}) = \frac{x^{u_\mathbf{f}}}{x^{u_{\mathbf{e_j}}}}\mathbf{e_j}$, $\text{LM}(\mathbf{g}) = \frac{x^{u_\mathbf{g}}}{x^{u_{\mathbf{e_j}}}}\mathbf{e_j}$, and:

$$\mathbf{h} = \text{LCM}(\text{LM}(\mathbf{f}), \text{LM}(\mathbf{g})) = \text{LCM}\left(\frac{x^{u_\mathbf{f}}}{x^{u_{\mathbf{e_j}}}}, \frac{x^{u_\mathbf{g}}}{x^{u_{\mathbf{e_j}}}}\right)\mathbf{e_j} = \frac{\text{LCM}\left(x^{u_\mathbf{f}}, x^{u_\mathbf{g}}\right)}{x^{u_{\mathbf{e_j}}}}\mathbf{e_j}.$$

Let $x^\ell = \text{LCM}(x^{u_\mathbf{f}}, x^{u_\mathbf{g}}) = x^{\text{LCM}(u_\mathbf{f}, u_\mathbf{g})}$, giving us $\mathbf{h} = \frac{x^\ell}{x^{u_{\mathbf{e_j}}}}\mathbf{e_j}$. We now have

$$\frac{\mathbf{h}}{\text{LT}(\mathbf{f})} = \frac{\frac{x^\ell}{x^{u_{\mathbf{e_j}}}}\mathbf{e_j}}{c_\mathbf{f}\frac{x^{u_\mathbf{f}}}{x^{u_{\mathbf{e_j}}}}\mathbf{e_j}} = \frac{x^\ell}{c_\mathbf{f} x^{u_\mathbf{f}}},$$

where $c_\mathbf{f} \neq 0$ is the field constant in the leading term of $\mathbf{f}$. Similarly, we get

$$\frac{\mathbf{h}}{\text{LT}(\mathbf{g})} = \frac{x^\ell}{c_\mathbf{g} x^{u_\mathbf{g}}}, \quad c_\mathbf{g} \neq 0.$$

Putting it together, we have

$$\begin{aligned} S(\mathbf{f},\mathbf{g}) &= \frac{\mathbf{h}}{\text{LT}(\mathbf{f})}\mathbf{f} - \frac{\mathbf{h}}{\text{LT}(\mathbf{g})}\mathbf{g} \\ &= \left(\frac{x^\ell}{c_\mathbf{f} x^{u_\mathbf{f}}} \sum_{i=1}^m c_i \frac{x^{u_\mathbf{f}}}{x^{u_{\mathbf{e_i}}}}\mathbf{e_i}\right) - \left(\frac{x^\ell}{c_\mathbf{g} x^{u_\mathbf{g}}} \sum_{i=1}^m c'_i \frac{x^{u_\mathbf{g}}}{x^{u_{\mathbf{e_i}}}}\mathbf{e_i}\right) = \sum_{i=1}^m d_i \frac{x^\ell}{x^{u_{\mathbf{e_i}}}}\mathbf{e_i}, \end{aligned}$$

where $d_i = c_i/c_\mathbf{f} - c'_i/c_\mathbf{g}$. Comparing with Equation (6), we see that $S(\mathbf{f},\mathbf{g})$ is homogeneous with $u_{S(\mathbf{f},\mathbf{g})} = \ell$.

Having computed the S-polynomial, Buchberger next divides it by the current homogeneous basis G using a call to the Divide algorithm [13].

Lemma 3. Divide$(\mathbf{f}, (\mathbf{f_1}, \ldots, \mathbf{f_t}))$ *returns a homogeneous remainder vector* $\mathbf{r}$ *for homogeneous vectors* $\mathbf{f}, \mathbf{f_i} \in R^m$.

Proof. Initially, we set $\mathbf{r}$ and $\mathbf{p}$ to be $\mathbf{0}$ and $\mathbf{f}$, respectively, so they are both trivially homogeneous. Since both $\mathbf{f_i}$ and $\mathbf{p}$ are homogeneous, we have $\mathbf{f_i} = \sum_{j=1}^m c_{ij}\frac{x^{u_{\mathbf{f_i}}}}{x^{u_{\mathbf{e_j}}}}\mathbf{e_j}$, $\mathbf{p} = \sum_{j=1}^m d_j \frac{x^{u_\mathbf{p}}}{x^{u_{\mathbf{e_j}}}}\mathbf{e_j}$. Since $\text{LT}(\mathbf{f_i})$ divides $\text{LT}(\mathbf{p})$, the terms must share basis element $\mathbf{e_k}$ and we have $\text{LT}(\mathbf{f_i}) = c_{ik}\frac{x^{u_{\mathbf{f_i}}}}{x^{u_{\mathbf{e_k}}}}\mathbf{e_k}$, $\text{LT}(\mathbf{p}) = d_k \frac{x^{u_\mathbf{p}}}{x^{u_{\mathbf{e_k}}}}\mathbf{e_k}$, $\text{LT}(\mathbf{p})/\text{LT}(\mathbf{f_i}) = \frac{d_k}{c_{ik}} \cdot \frac{x^{u_\mathbf{p}}}{x^{u_{\mathbf{f_i}}}}$, where $x^{u_\mathbf{p}} >_{\text{lex}} x^{u_{\mathbf{f_i}}}$ so that the division makes sense. Then, $\mathbf{p}$ is assigned to

$$\begin{aligned} \mathbf{p} - (\text{LT}(\mathbf{p})/\text{LT}(\mathbf{f_i}))\mathbf{f_i} &= \sum_{j=1}^m d_j \frac{x^{u_\mathbf{p}}}{x^{u_{\mathbf{e_j}}}}\mathbf{e_j} - \left(\frac{d_k}{c_{ik}} \cdot \frac{x^{u_\mathbf{p}}}{x^{u_{\mathbf{f_i}}}}\right) \sum_{j=1}^m c_{ij} \frac{x^{u_{\mathbf{f_i}}}}{x^{u_{\mathbf{e_j}}}}\mathbf{e_j} \\ &= \sum_{j=1}^m \left(d_j - \frac{d_k \cdot c_{ij}}{c_{ik}}\right) \frac{x^{u_\mathbf{p}}}{x^{u_{\mathbf{e_j}}}}\mathbf{e_j} = \sum_{j=1}^m d'_j \frac{x^{u_\mathbf{p}}}{x^{u_{\mathbf{e_j}}}}\mathbf{e_j}, \end{aligned}$$

where $d'_j = d_j - d_k \cdot c_{ij}/c_{ik}$ and $d'_k = 0$, so the subtraction eliminates the kth term. The final sum means that $\mathbf{p}$ is now a new homogeneous polynomial with the same coordinate $u_\mathbf{p}$ as before. Similarly, $\text{LT}(p)$ is added to $\mathbf{r}$ and subtracted from $\mathbf{p}$, and neither action changes the homogeneity of either vector. Both remain homogeneous with coordinate $u^\mathbf{p}$.

Theorem 1 (homogeneous Gröbner). *The* BUCHBERGER *algorithm computes a homogeneous Gröbner basis for a homogeneous matrix.*

Proof. Initially, the algorithm sets G to be the set of columns of the input matrix F, so the vectors in G are homogeneous by Lemma 1. The algorithm then computes the S-polynomial of homogeneous vectors $\mathbf{f}, \mathbf{g} \in G$. By Lemma 2, the S-polynomial is homogeneous. It then divides the S-polynomial by G. Since the input is homogeneous, DIVIDE produces a homogeneous remainder $\mathbf{r}$ by Lemma 3. Since only homogeneous vectors are added to G, it remains homogeneous. We may extend this result easily to the reduced Gröbner basis.

Using similar arguments, we may show the following result.

Theorem 2 (homogenous syzygy). *For a homogeneous matrix, all matrices encountered in the computation of the syzygy module are homogeneous.*

3.2 Optimizations

We have shown that the structure inherent in a multifiltration allows us to compute using homogeneous vectors and matrices whose entries are monomials only. We next explore the consequences of this restriction on both the data structures and complexity of the algorithms.

By Definition (4), an $m \times n$ homogeneous matrix naturally associates monomials to the standard bases for R^m and R^n. Moreover, every non-zero entry of the matrix is a quotient of these monomials as the matrix is homogeneous. Therefore, we do not need to store the matrix entries, but simply the field elements of the matrix along with the monomials for the bases. We may modify two standard data structures to represent the matrix.

- *linked list:* Each column stores its monomial as well as a linked-list of its non-zero entries in sorted order. The non-zero entries are represented by the row index and the field element. The matrix is simply a list of these columns in sorted order.
- *matrix:* Each column stores its monomial as well as the column of field coefficients. If we are computing over a finite field, we may pack bits for space efficiency.

The linked-list representation is appropriate for sparse matrices as it is space-efficient at the price of linear access time. This is essentially the representation used for computing in the one-dimensional setting [4]. In contrast, the matrix representation is appropriate for dense matrices as it provides constant access time at the cost of storing all zero entries. The multidimensional setting provides us with denser matrices, as we shall see, so the matrix representation becomes a viable structure.

In addition, the matrix representation is optimally suited to computing over the field $\mathbb{Z}_2$, the field often commonly employed in topological data analysis. The matrix entries each take one bit and the column entries may be packed

into machine words. Moreover, the only operation required by the algorithms is *symmetric difference* which may be implemented as a binary XOR operation provided by the chip. This approach gives us bit-level parallelism for free: On a 64-bit machine, we perform symmetric difference 64 times faster than on the list. The combination of these techniques allow the matrix structure to perform better than the linked-list representation in practice.

We may also exploit homogeneity to speed up the computation of new vectors and their insertion into the basis. We demonstrate this briefly using the Buchberger algorithm. We order the columns of input matrix G using the POT rule for vectors [13]. Suppose we have $\mathbf{f}, \mathbf{g} \in G$ with $\mathbf{f} > \mathbf{g}$. If $S(\mathbf{f}, \mathbf{g}) \neq 0$, $\text{LT}(\mathbf{f})$ and $\text{LT}(\mathbf{g})$ contain the same basis, which the S-polynomial eliminates. So, we have $S(\mathbf{f}, \mathbf{g}) < \mathbf{g} < \mathbf{f}$. This implies that when dividing $S(\mathbf{f}, \mathbf{g})$ by the vectors in G, we need only consider vectors that are smaller than $\mathbf{g}$. Since the vectors are in sorted order, we consider each in turn until we can no longer divide. By the POT rule, we may now insert the new remainder column here into the basis G. This gives us a constant time insertion operation for maintaining the ordering, as well as faster computation of the Gröbner basis.

3.3 Complexity

Our optimizations from the last section allow us to give simple polynomial bounds on our multigraded algorithms. These bounds, in turn, imply that we may compute multidimensional persistence in polynomial time.

Lemma 4. *Let F be an $m \times n$ homogeneous matrix of monomials. The Gröbner basis G contains $O(n^2 m)$ vectors in the worst case. We may compute G using* Buchberger *in $O(n^4 m^3)$ worst-case time.*

Proof. In the worst case, F contains nm unique monomials. Each column $\mathbf{f} \in F$ may have any of the nm monomials as its monomial when included in the Gröbner basis G Therefore, the total number of columns in the G is $O(n^2 m)$. In computing the Gröbner Basis, we compare all columns pairwise, so the total number of comparisons is $O(n^4 m^2)$. Dividing the S-polynomial takes $O(m)$ time. Therefore, the worst-case running time is $O(n^4 m^3)$.

We omit the proof of the following due to lack of space and refer the reader to the full manuscript [12].

Lemma 5. *Let F be an $m \times n$ homogeneous matrix of monomials and G be the Gröbner Basis of F. The Syzygy module S for G may be computed using* Schreyer's algorithm *in $O(n^4 m^2)$ worst-case time.*

Theorem 3. *Multidimensional persistence may be computed in polynomial time.*

4 Conclusion

In this paper, we develop polynomial time algorithms for multidimensional persistence by recasting the problem into computational algebraic geometry. Although the recast problem is Expspace-complete, we exploit the multigraded

setting to develop practical algorithms. We have implemented all our algorithms and provide statistical experiments to demonstrate their feasibility in the full manuscript [12]. For additional speedup, we plan to parallelize the computation by batching and threading the XOR operations. We also plan to apply our algorithms toward studying scientific data. For instance, for zero-dimensional homology, multidimensional persistence corresponds to *clustering* multiparameterized data, This gives us a fresh perspective, as well as a new arsenal of computational tools, to attack an old and significant problem in data analysis.

References

1. Carlsson, G., Zomorodian, A.: The theory of multidimensional persistence. Discrete & Computational Geometry **42**(1) (2009) 71–93
2. Gromov, M.: Hyperbolic groups. In Gersten, S., ed.: Essays in Group Theory. Springer Verlag, New York, NY (1987) 75–263
3. de Silva, V., Carlsson, G.: Topological estimation using witness complexes. In: Proc. Symposium on Point-Based Graphics. (2004) 157–166
4. Zomorodian, A., Carlsson, G.: Computing persistent homology. Discrete & Computational Geometry **33**(2) (2005) 249–274
5. Collins, A., Zomorodian, A., Carlsson, G., Guibas, L.: A barcode shape descriptor for curve point cloud data. Computers and Graphics **28** (2004) 881–894
6. Gyulassy, A., Natarajan, V., Pascucci, V., Bremer, P.T., Hamann, B.: Topology-based simplification for feature extraction from 3D scalar fields. In: Proc. IEEE Visualization. (2005) 275–280
7. de Silva, V., Ghrist, R., Muhammad, A.: Blind swarms for coverage in 2-D. In: Proceedings of Robotics: Science and Systems. (2005) `http://www.roboticsproceedings.org/rss01/`.
8. Singh, G., Memoli, F., Ishkhanov, T., Sapiro, G., Carlsson, G., Ringach, D.L.: Topological analysis of population activity in visual cortex. Journal of Vision **8**(8) (6 2008) 1–18
9. Carlsson, G., Ishkhanov, T., de Silva, V., Zomorodian, A.: On the local behavior of spaces of natural images. International Journal of Computer Vision **76**(1) (2008) 1–12
10. Frosini, P., Mulazzani, M.: Size homotopy groups for computation of natural size distances. Bull. Belg. Math. Soc. Simon Stevin **6**(3) (1999) 455–464
11. Cohen-Steiner, D., Edelsbrunner, H., Morozov, D.: Vines and vineyards by updating persistence in linear time. In: Proc. ACM Symposium on Computational Geometry. (2006) 119 – 126
12. Carlsson, G., Singh, G., Zomorodian, A.: Computing multidimensional persistence `http://arxiv.org/abs/0907.2423`.
13. Cox, D.A., Little, J., O'Shea, D.: Using algebraic geometry. Second edn. Volume 185 of Graduate Texts in Mathematics. Springer, New York (2005)
14. Mayr, E.W.: Some complexity results for polynomial ideals. Journal of Complexity **13**(3) (1997) 303–325
15. von zur Gathen, J., Gerhard, J.: Modern Computer Algebra. Second edn. Cambridge University Press, Cambridge, UK (2003)