

COMPUTING HOMOLOGY AND PERSISTENT HOMOLOGY USING ITERATED MORSE DECOMPOSITION

PAWEŁ DŁOTKO AND HUBERT WAGNER

ABSTRACT. In this paper we present a new approach to computing homology (with field coefficients) and persistent homology. We use concepts from discrete Morse theory, to provide an algorithm which can be expressed solely in terms of simple graph theoretical operations. We use *iterated Morse decomposition*, which allows us to sidetrack many problems related to the standard discrete Morse theory. In particular, this approach is provably correct in any dimension.

1. PREVIEW.

In this section we outline the purpose of the paper, assuming reader's familiarity with some concepts from computational topology. All the concepts will be carefully explained later. In this paper we introduce a new method to compute homology and persistent homology over field coefficients. The method is based on discrete Morse theory and is designed to be graph-theoretic.

We present a brief, intuitive illustration of our method. As an example, let us consider a triangulation of a Dunce hat presented in Figure 1.a. We want to remind that this space has trivial homology but nontrivial topology. We will use discrete Morse theory to simplify the space, while preserving the homology. First, let us build a discrete Morse matching on this triangulation. It is well known that a Dunce hat has no perfect¹ Morse complex [1]. Therefore, for any Morse matching we obtain some critical cells not corresponding to homology generators. The matching presented in Figure 1.b is optimal i.e. there are as few critical cells as possible. Now the Morse boundary is computed using the V-paths marked in Figure 1.c. The resulting Morse complex (with \mathbb{Z}_2 coefficients) is shown in Figure 1.d. Normally, in order to compute homology of a chain complex, boundary matrix is produced and Smith

¹A Morse complex is *perfect* if each critical cell corresponds to exactly one homology generator.

Normal Form diagonalization is performed. However we would like to introduce an alternative approach.

Using Kozlov’s version of discrete Morse theory, we can iterate the Morse complex construction. In other words, we build a Morse complex of a Morse complex etc. So, here we compute a Morse matching of a chain complex presented in Figure 1.d. The only matched pair is marked with an arrow. Once the (iterated) Morse complex is computed, we are left with a single 0-dimensional cell. It cannot be paired anymore and corresponds to the only homological feature: the connected component. This way we have computed the homology of the Dunce hat.

Later the presented technique will be referred to as *iterated Morse complex* construction or *iterated Morse decomposition*. In this paper we will show that with the presented technique one can always acquire homology with field coefficients. We will also generalize it to the setting of persistent homology. As a result, we introduce a novel way to compute homology and persistence over a field.

2. MOTIVATION

While persistent homology can potentially be applied to a plethora of different practical problems, ranging from sensor networks [35] to root architecture analysis [10], performance tends to be a problem. In particular, there is growing interest in analysis of high-dimensional topological features (going beyond connected components and 1-cycles). In low dimensions there exist efficient algorithms, but higher dimensional cases are still challenging.

Such applications include analysis of complex networks such as social-network and biological networks such as gene-regulatory networks. Computing homology or persistent homology of maps between spaces leads to high dimensional datasets as the studied space is the Cartesian product of the source and target space [19].

The only class of algorithms to compute persistent homology in the general case is based on matrix reductions. Such an approach was recently shown to expose roughly quadratic computational complexity, for data coming from certain practical applications [38]. Also, it is not very suitable for distributed computing, which is a necessity as the datasets grow larger.

Our aim is to propose an algorithmic framework which would scale reasonably well as the size of data (and its dimension) grows. In lower dimensions several algorithms were proposed and their efficiency stemmed from using fast techniques from graph-theory. Prompted by

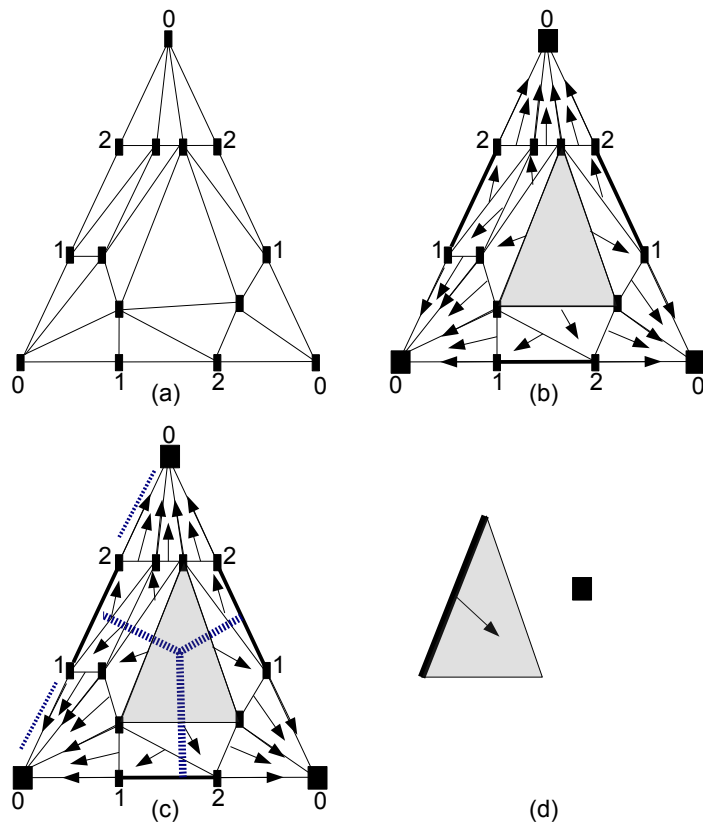


FIGURE 1. *Iterated* version of Morse complex construction on a Dunce hat. On the top, the 0 and 1 dimensional critical cells are marked with bold, the middle gray triangle is the unique critical 2-cell. In the bottom left picture the V-paths used to compute the Morse complex after the first iteration of the construction. On the bottom right, the second (and final) iteration of Morse complex construction. The remaining vertex corresponds to the unique homology generator in dimension 0.

this observation, we wanted to propose a similar approach which would work for any dimension.

However, directly extending the existing approaches to higher dimensions is (as we believe) impossible. Combinatorial techniques, for instance described in [7, 34], crucially depend on discrete Morse theory. In particular, a perfect Morse complex is (implicitly) constructed

in most cases, which allows to read homology information right away. Computing such perfect Morse complexes is hard (or even impossible) in higher dimensions.

In case of field coefficients, which are important for practical applications, the situation is more tractable. We use the properties of *iterated* Morse complexes [38], which always exist and are easy to compute. Additionally, we build on top of a recent theoretical framework by Mischaikow and Nanda [29], which extends Morse theory to filtrations of spaces.

The following paper describes the theory and algorithms for computing homology and persistent homology using *iterated* Morse decomposition. We prove that the algorithm is correct for chain-complexes of any dimension. This includes commonly used simplicial and cubical complexes. Further, we show that, just as we intended, the algorithms can be entirely expressed in terms of basic graph-theoretical techniques. It promises that in terms of implementations, using efficient graph libraries [27] will result in a scalable solution.

The paper is structured as follows. Section 3 is a survey of existing work in the topic. In Section 4 an introduction to homology and persistent homology is given, together with the necessary algorithmic background. In Section 5 a Discrete Morse Theory is highlighted. In Section 6 the concept of *iterated* Morse complex, crucial for this paper, is introduced. It is also explained there how this concept is used to compute homology. In Section 7 these results are extended to simplification of filtered complexes, which is a preprocessing step for computing persistence. Later in Section 8 it is explained how to compute persistence using solely *iterated* Morse complex. Finally in Section 9 conclusions are drawn.

3. PREVIOUS WORK.

Computations of homology and persistent homology is a well established area of research with a rich history. In this section we want to summarize the main contributions and historical landmarks in this subject.

The classical way of computing homology is by using Smith Normal Form (SNF) of a boundary operator matrix, see [30]. The classical algorithm has hyper-cubical complexity (in case of integer coefficients), see [36]. It is however possible to perform SNF in cubic time when field coefficients are considered.

In the nineties Delfinado and Edelsbrunner provided an incremental algorithm for Betti numbers computation [7]. This algorithm exhibits

linear time complexity and works for sub-triangulations of 3 dimensional sphere. There are also algorithms to compute homology with chain contractions and so called AT-models [15].

In 2003, after a few earlier iterations ([39, 14, 33]), persistent homology was introduced in its contemporary form [11]. In the same paper, a matrix-reduction algorithm to compute persistence (Algorithm 1) was given. This is what is considered the standard algorithm to compute persistence. For a comprehensive presentation the reader should consult [10]. Algorithmic results are further discussed in Section 4.

Discrete Morse theory (DMT) was introduced by Robin Forman [13]. Later a more general, algebraic version was developed [25]. The comprehensive presentation of algebraic discrete Morse theory can be found in [25]. The idea of using DMT for homology computations has been introduced by Lewiner [26]. The complexity aspects of DMT has been discussed in [22]. The notions of F -perfect and F -optimal Morse complexes are discussed in [1]. A simplification algorithm for a Morse complexes on 2-manifolds has been presented in [2]. A divide and conquer algorithm to compute Morse complexes has been presented in [17].

Recently Robins, Wood and Sheppard have provided a practical link between discrete Morse theory and persistence [34]. In this paper they introduce an optimal simplification scheme for persistence in case 3-dimensional complexes. The optimality of the presented result is restricted to 3 dimensions due to some deep results from simple homotopy theory. An extension of Morse theory suitable for simplifying filtered chain complexes in any dimension was later provided by Mischaikow and Nanda [29]. In short, by performing Morse matchings independently for each filtration level, a simplified filtered Morse complex is obtained.

The idea of iterating Morse complex construction has already been used in [38, 19] as a tool to decrease the size of complexes before standard algebraic computations.

There are many software libraries to compute homology and persistent homology. For a homology software the reader should consult [3, 6, 24]. For persistent homology [23, 8, 32] are recommended.

4. BACKGROUND

4.1. Complexes. We assume that the input data is represented as a *chain complex* with field coefficients. In the most typical case, this chain complex comes from a CW-decomposition of a given space which is a decomposition of a space into *cells* of different dimensions. In practice, simplicial and cubical complexes are used. For simplicity, we

will use \mathbb{Z}_2 coefficients throughout the paper, as this is the standard setting for persistence. However, we want to remark that the presented algorithms work for any field coefficients.

4.2. Boundary maps. Let us fix a complex \mathcal{K} . Cells of \mathcal{K} have different dimensions and are connected by boundary relations. If a $(p-1)$ -cell a has non-zero boundary coefficient with a p -cell B , we say a is a proper face of B , and B is a proper coface of a . (Notation: capital letters denote higher dimensional cell where a cell and its face is considered). Let a p -chain be a formal sum of p -cells with the \mathbb{Z}_2 coefficients. The boundary operator ∂_p maps p -chains into $p-1$ -dimensional boundary chains. The chain of (co-)faces is called a (co-)boundary. We can extend the boundary operator linearly to p -chains. For any p -chain $c = \sum a_i c_i$, we have $\partial_p c = \sum a_i \partial_p c_i$. It is assumed that the boundary of a boundary is zero, or formally: $\partial_p \partial_{p+1} = 0$. The p -chains, together with addition modulo 2, form a group of p -chains, denoted by C_p .

The boundary operator ∂_p can be written as a binary matrix (also denoted ∂_p), whose columns represent the boundaries and rows represent coboundaries of cells.

4.3. Standard homology. Intuitively, homology can be used to capture holes of complex \mathcal{K} . In 3-dimensional case holes are: connected components, tunnels, and voids. To define it formally, let us first introduce the group of p -cycles, $Z_p(\mathcal{K}) = \ker \partial_p$ and its subgroup: the group of p -boundaries, $B_p(\mathcal{K}) = \text{im} \partial_{p+1}$. The p -th homology group is the quotient $H_p = Z_p(\mathcal{K})/B_p(\mathcal{K})$. The p -th Betti number, denoted by β_p , is the rank of this group and counts the number of p -dimensional holes.

4.4. Filtrations and persistence. For a given complex \mathcal{K} , a filtration is defined as a nested sequence of its subcomplexes: $\emptyset = \mathcal{K}_{-1} \subseteq \mathcal{K}_0 \subseteq \mathcal{K}_1 \subseteq \dots \subseteq \mathcal{K}_n = \mathcal{K}$ [10]. In case of persistence, filtrations are often generated by a *filtering function*, $g : \mathcal{K} \rightarrow \mathbb{Z}$ defined on the input complex. We require that $g(a) \leq g(B)$ whenever a is a face of B . This property guarantees that the sub-level sets $\mathcal{K}_t = g^{-1}(-\infty, t]$ are subcomplexes of \mathcal{K} for each value of $t \in \mathbb{Z}$. The inclusions from \mathcal{K}_i to \mathcal{K}_j , for $i \leq j$ induce homomorphisms, $f^{i,j} : H(\mathcal{K}_i) \rightarrow H(\mathcal{K}_j)$. Complex \mathcal{K} with filtration will be referred to as *filtered complex*.

Given a complex \mathcal{K} and a filtering function $g : \mathcal{K} \rightarrow \mathbb{Z}$, *persistent* homology studies homological changes of the sub-level complexes, $\mathcal{K}_t = g^{-1}(-\infty, t]$. Persistent homology captures the birth and death times of homology classes of the sub-level complexes, as t grows from $-\infty$ to $+\infty$. By birth, we mean that a homology feature is created; by death,

we mean it either becomes trivial or becomes identical to some other class born earlier. The *persistence*, or lifetime of a class, is the difference between the death and birth times. Often a multiset of *persistence intervals* is used to represent persistence. An interval encodes a lifetime of a homology class of a given dimension. We say that two spaces have the same persistence, if their corresponding persistence intervals are the same.

The formal definition is as follows (after [10]): The p -th persistent homology groups of filtered complex \mathcal{K} are the images of the homomorphisms induced by inclusion, $H^{i,j}(\mathcal{K}) = \text{im } f^{i,j}$. For a standard definition of persistence diagram and persistence intervals the reader should consult [10].

We want to remind a theorem saying when persistence of two filtered complexes are equal:

Theorem 4.1 (Persistence equivalence theorem, [10]). *Consider persistent homology of two filtered complexes X and Y . Let $\phi_i : H_*(X_i) \rightarrow H_*(Y_i)$:*

$$\begin{array}{ccccccc} H_*(X_0) & \longrightarrow & H_*(X_1) & \longrightarrow & \dots & \longrightarrow & H_*(X_{n-1}) & \longrightarrow & H_*(X_n) \\ \phi_0 \downarrow & & \phi_1 \downarrow & & & & \phi_{n-1} \downarrow & & \phi_n \downarrow \\ H_*(Y_0) & \longrightarrow & H_*(Y_1) & \longrightarrow & \dots & \longrightarrow & H_*(Y_{n-1}) & \longrightarrow & H_*(Y_n) \end{array}$$

If the ϕ_i are isomorphisms and all the squares commute, then the persistence diagrams of X and Y are the same.

4.5. Computing persistence. Let us have a filtered chain complex \mathcal{K} . Boundary matrix ∂ of \mathcal{K} encodes the boundary relations between cells of different dimensions. Column i corresponds to the boundary of cell c_i , row j corresponds to the coboundary of cell c_j . In case of \mathbb{Z}_2 coefficients it can be defined as follows:

$$\partial(i, j) = \begin{cases} 1 & \text{if } c_j \text{ is a face of } c_i \\ 0 & \text{otherwise} \end{cases}$$

By $\kappa(c_i, c_j) := \partial(i, j)$ we denote the *incidence index* of cells c_i and c_j . In order to compute persistence, a *sorted* boundary matrix is required: For two columns (or rows) $i < j$, the corresponding cells must satisfy: $g(c_i) \leq g(c_j)$. Using such a matrix, we can compute persistence using matrix-reduction Algorithm 1 as defined in [10]. The value $\text{low}(i)$ marks the maximum (lowest) position of a one in column i , if any. We assume it to be zero for zeroed columns. We say that there is a *collision* at column j , if there exists a column $k < j$ such that

$low(k) = low(j)$, provided $low(k)$ and $low(j)$ are nonzero. The matrix is said to be reduced if there are no collisions. In such a case all the lowest ones are unique. As proven in [10], persistent homology is fully determined by the positions of lowest ones in the reduced sorted matrix: If column i is zero, corresponding p -dimensional cell c_i creates a infinite p -dimensional persistent homology class. For a non-zero column j with $k = low(j)$, the corresponding $(p+1)$ -cell c_j kills a persistent homology class created by p -cell c_k .

The algorithm proceeds with columns from left to right, removing any collisions. Later we will analyze the behavior of the algorithm to prove the correctness of our simplification algorithm.

Algorithm 1 Compute reduced matrix

Input: Sorted binary matrix ∂ of size $n \times n$

Output: Reduced binary matrix R , which encodes persistence

- 1: $R := \partial$
 - 2: **for** $j := 1$ to n **do**
 - 3: **while** there exists in R a nonzero column $k < j$ with $low(k) = low(j)$ **do**
 - 4: add column k to column j (mod 2) and store as column j
-

A simple illustration of Algorithm 1 can be found in Figure 8 in the Appendix.

4.6. Algorithms and their complexity. Applying the presented matrix-reduction algorithm to the input complex is the standard way to compute persistent homology groups. It works for general complexes in arbitrary dimensions. The worst-case complexity is $O(n^3)$, where n is the size of the input complex. Milosavljevic et al. [28] showed that persistent homology can be computed in matrix multiplication time $O(n^\omega)$ where the currently best estimation of ω is 2.3727. Chen and Kerber [4] proposed a randomized algorithm to compute only pairs with persistence above a chosen threshold. Despite improving the theoretical complexity, it is unclear whether these methods are better in practice.

When focusing on 0-dimensional homology, union-find data structures can be used to compute persistence in time $O(n\alpha(n))$ [10], where α is the inverse of the Ackermann functions and n the size of the input.

A recent variation of the standard algorithm, introduced by Chen and Kerber [5] significantly reduces the amount of computations. This idea was also used in [37] to compute persistence for n -dimensional images. In general, the regular structure of cubical complexes can be

exploited, which allows for handling large inputs. In such a situation the size of the boundary matrix is the main obstacle. Preprocessing the input complex using discrete Morse theory, as proposed by Robins [34], significantly reduces the size of the boundary matrix, while preserving persistence. In case of 3D grayscale images, an efficient parallel implementation was proposed in [16], allowing for handling large ($\approx 1200^3$) images on commodity hardware. The standard matrix-reduction algorithm is used in the final step of computations.

The approach by Robins works for arbitrary complexes and in dimension three the preprocessing results in the smallest possible boundary matrix [34] (counting the number of rows/columns). The algorithm used in [34] depends crucially on simple-homotopy theory, which makes it hard to directly generalize the optimality result to higher dimensions. A recent paper by Mischaikow et al. [29] proposes a handy theoretical framework, where discrete Morse theory is extended from complexes to filtrations.

In our approach, we use the existing algorithms for discrete Morse complex construction. We use them to iteratively simplify the input complex. Note that our approach is significantly different from the simplification scheme by Pascucci et al., where Morse complexes are iteratively simplified in terms of (roughly speaking) topology or in a sense: persistent homology. Our aim is different: persistence is never affected, and the simplification is only in terms of the number of cells representing the complex.

5. DISCRETE MORSE THEORY.

5.1. Morse matching and Morse graph. In this section an algorithmic introduction to discrete Morse theory is given. For further theoretical details please consult [13, 25]. Let us have a complex \mathcal{K} . Discrete Morse theory partitions the cells of \mathcal{K} into *matched cells* and *critical cells*. The critical cells, together with a boundary operator we describe later, form a chain complex called the *Morse complex*. Importantly, this Morse complex has homology isomorphic with the homology of the initial complex. A procedure to compute Morse complex is given in Algorithm 2.

The first step in constructing a Morse complex requires finding an acyclic Morse matching M of the complex \mathcal{K} . The matching is a partial map $M : \mathcal{K} \rightarrow \mathcal{K}$. Each cell of \mathcal{K} can be matched with exactly one of its co-faces. Some cells can remain unmatched, and are called *critical*. These cells constitute the resulting Morse complex.

Algorithm 2 Compute Morse complex

Input: Input complex \mathcal{K}

Output: Resulting Morse complex

- 1: $M :=$ acyclic Morse matching on \mathcal{K}
 - 2: $C :=$ list of critical cells of M
 - 3: $G :=$ Morse graph of M, C
 - 4: $bd :=$ compute Morse Boundary of G (as in Algorithm 4)
 - 5: return (C, bd)
-

Let us introduce a concept of a *Morse graph* of a complex \mathcal{K} and matching M . It is a directed graph whose vertices are formed by cells of a complex. A directed edge from vertex A to vertex b is added whenever b is in the boundary of A and the cells A and b are not matched in M . If they are matched in M , a direct edge from b to A is added in the Morse graph. The matching M is called *acyclic* if the corresponding Morse graph is a directed acyclic graph (DAG). The paths of this graph are often called V -paths. There are various strategies of obtaining acyclic Morse matchings. Later in this paper we assume that every matching is acyclic.

A Morse matching is called *perfect* if each unmatched cell corresponds to a homology generator of the original complex. (We choose to talk about (perfect) matchings, but in the literature (perfect) Morse functions, vector-fields and complexes are discussed.) This depends on the choice of coefficients, for example in case of a field coefficients F , we can talk about F -perfect matchings [1]. Some spaces do not admit perfect matchings, for example the Dunce hat (presented in Section 1), being contractible but non-collapsible. In this case some critical cells are spurious, in the sense that they do not correspond to any homology generators.

One can try to construct a best possible matching, minimizing the number of critical cells. This problem is known to be NP-complete and MAXSNP-hard [22]. It means that computing the best possible matching is computationally expensive and there is no hope to find a fully-polynomial time approximation strategy. As a result, no polynomial-time algorithm can give arbitrarily good bounds on the number of spurious critical cells.

Once an acyclic Morse matching M is obtained, we proceed with computing the Morse boundary. This procedure is described in [13]. The idea is illustrated in Figure 3. Forman [13] proved that the resulting Morse complex has isomorphic homology to the homology of the initial complex. He also provided a formula which computes the

boundary of each cell in a Morse complex. Kozlov generalized these proofs to the setting of arbitrary chain complexes [25]. One can use Morse complex construction to compute homology. Later we show that similar construction can be also used to compute persistence.

5.2. Discrete Morse theory for filtered complexes. Recently there were first successful attempts to use discrete Morse theory to compute persistence [34, 16] (in case of 3-d gray-scale images), [2] (in case of 2-manifolds). The first successful attempt to provide a Morse-theoretic categorical framework for persistent homology was made in [29].

In this section we will first recall the basic ideas from [29]. We say that the Morse matching M is *compatible with filtration* of \mathcal{K} if for every matched $A \in \mathcal{K}$, $g(A) = g(M(A))^2$. In other words, the matchings are made between elements of the same filtration level. Consequently, directed paths cannot move upwards the filtration (if they did, we would lose the sub-complex filtration property in the corresponding Morse complex, because a cell could enter the filtration strictly before its faces).

The key result in [29] is that the persistence diagram of a filtered complex \mathcal{K} and a Morse complex $\mathbb{M}(\mathcal{K})$ with the Morse matchings compatible with filtration are the same. In Figure 2 an example of Morse matchings compatible and non-compatible with filtration are shown.

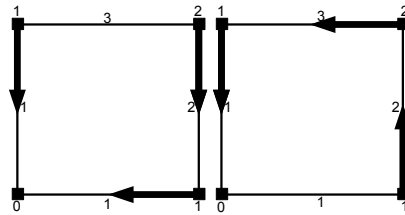


FIGURE 2. On the left the Morse matching compatible with filtration. In this case persistent homology for both initial and the Morse complex is $[0, \infty]$ in dimension 0 and $[3, \infty]$ in dimension 1. On the right a correct Morse matching in a sense of standard Discrete Morse Theory which is not compatible with filtration is depicted. The persistent homology of the Morse complex on the right in dimension one is $[1, \infty]$ which is not correct.

²By $M(A)$ we denote the element matched with A .

5.3. Computing Morse complex with graph algorithms. In this section we will show that the entire Morse complex construction can be computed using standard graph algorithms. The chain complex (with \mathbb{Z}_2 coefficients) can be interpreted as a graph – namely the Morse graph. We assume that initially no matchings are made. The whole construction can be divided into two essential parts: finding an acyclic Morse matching and computing the Morse boundary. Both parts are described below. We want to point out that for presentation’s sake the algorithms presented in this section are not necessarily optimal. For that reason we also present just a version for binary coefficients. They can be easily generalized to arbitrary field coefficients.

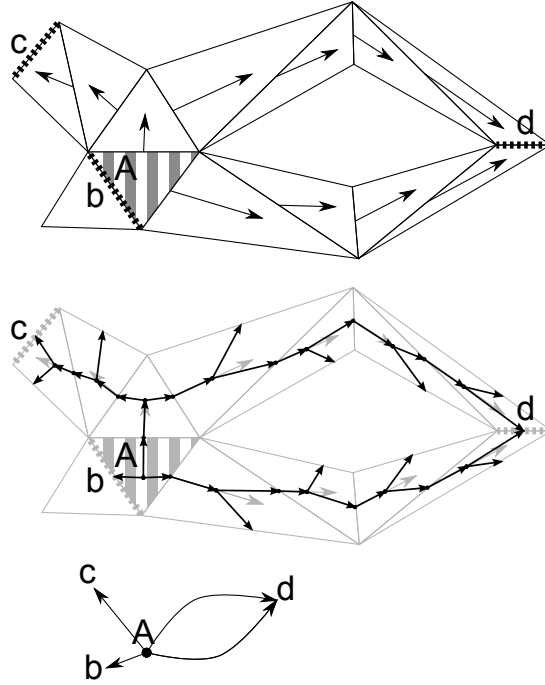


FIGURE 3. This picture shows the process of computing the Morse boundary of cell A . A Morse matching is shown. The part of corresponding (acyclic) Morse graph is built. Finally boundary relations between cells are computed, by following paths emanating from the boundary of cell A and ending at critical cells. Note that there are two paths between cells A and d , which yields boundary relation equal to zero for coefficients in \mathbb{Z}_2 (d is not in the boundary of A).

5.3.1. *Computing acyclic matching.* There are many possible graph-based strategies for Morse matchings. Some strategies based on the idea of BFS spanning tree have been described in the literature [22, 26, 38]. In general, the problem of performing a Morse matching is equivalent to the following one – which directed edges in Morse graph can be reversed so that the graph remains acyclic. This is closely related to the *minimum feedback arc set* problem. While this problem is NP-hard, efficient approximation schemes exist [12].

For an illustration, a basic algorithm described in [18] will be reminded in Algorithm 3. The proof that the obtained graph is acyclic is easy

Algorithm 3 Compute Morse matching

Input: Morse graph G ;

Output: Changed graph G (edges between matched elements are reversed);

- 1: **while** Not all vertices of G are marked **do**
 - 2: Let i be the minimal dimension of unmarked element in G ;
 - 3: Pick A , unmarked i -dimensional cell in G and mark as *critical*;
 - 4: **while** There exists unmatched element $A \in G$ with unique unmatched element b in boundary **do**
 - 5: Make a Morse matching (A, b) , i.e. reverse an edge from A to b in G ;
 - 6: Mark A and b as matched;
-

but technical. Therefore it will not be presented here.

To put restrictions on the matching one should modify line 4 of Algorithm 3. In particular, one can ensure that the matchings are compatible with filtration. This is needed in case of persistence computations.

5.3.2. *Computing Morse boundaries.* Let us state the problem of computing Morse boundary in terms of graph theory. The Morse boundary of a critical cell is formed by the set of critical cells which are reachable by an *odd* number of paths in the Morse graph. This is a special case of Forman’s formula [13] in case of \mathbb{Z}_2 coefficients. Since the number of such paths can grow exponentially in the number of cells, brute force calculation is ineffective (as noted in [29]). However, this problem can be solved efficiently, exploiting the fact that the Morse graph is acyclic.

The following Algorithm 4 is simple to implement, and works in pessimistic time $O(c * (\|V\| + \|E\|))$, where V and E are respectively the vertices and edges of graph G and c is the number of critical cells. For a runtime- and memory-optimal one see [16].

Let $P_s(t)$ denote the number of distinct paths leading from vertex s to t , and $prev(v) = \{x \mid (x, v) \in E\}$ is the set of vertices preceding v in the directed graph. We have an obvious recurrence relation:

$$P_s(u) = \begin{cases} 1 & \text{for } u = s \\ \sum_{v \in prev(u)} P_s(v) & \text{for } u \neq s \end{cases}$$

This recurrence can be computed directly and also efficiently using memoization, but we propose an elegant graph-theoretical algorithm. To compute $P_s(v)$ the summation is done indirectly in line 10 of the Algorithm 4 by adding the value $P_s(u)$ where $u \in prev(v)$.

Algorithm 4 Compute Morse boundaries from Morse graph

Input: Directed Morse graph $G := (V, E)$

Output: Boundary relation ∂ of the Morse complex

```

1: sort G topologically
2: for each critical vertex  $s$  do
3:   assign  $P_s(v) := 0$  for each vertex  $v \neq s$ 
4:   assign  $P_s(s) := 1$ 
5:   for each vertex  $c$  following  $s$  in topological order do
6:     if  $c$  is critical then
7:        $\partial(s, c) := P_s(c) \bmod 2$ 
8:     else
9:       for each  $v \mid (c, v) \in E$  do
10:         $P_s(v) += P_s(c)$ 

```

Theorem 5.1. *Algorithm 4 is correct.*

Proof. We prove the correctness of the algorithm by induction on the iteration of the loop in line 5. The desired invariant is that whenever $P_s(c)$ is used, this value is already final and correct. Clearly, the value $P_s(s)$ is initialized correctly in line 4, so the correct value is used during the first iteration. Assume that the invariant holds for the first i iterations and vertex c is now processed. Note that the value of c depends on the values of $prev(c)$. Since we proceed in topological-sort order, all the vertices in $prev(c)$ have already been processed. By inductive assumption the values used to compute $P_s(c)$ were correct and final, therefore the value $P_s(c)$ is also correct and final. \square

6. ITERATED MORSE COMPLEX FOR HOMOLOGY

In this section the concept of *iterated Morse complex* is presented. Normally one aims at finding a Morse complex minimizing the number of critical cells. As mentioned earlier this is a hard algorithmic problem

and we do not tackle it. Instead we use an algorithm to iteratively construct a sequence of Morse complexes. If at a certain stage the obtained Morse complex is far from optimal, further iterations will be necessary to compute the homology of the considered complex. Still, the worst case computational time is cubical. The results presented in this section has already been sketched in [38].

Let \mathcal{C} be a category of chain complexes and let $\mathbb{M} : \mathcal{C} \rightarrow \mathcal{C}$ be a functor taking a chain complex and assigning it a Morse complex constructed on it. There are many possible strategies to construct Morse complexes. We assume that if a chain complex $\mathcal{K} \in \mathcal{C}$ has some available Morse matchings, \mathbb{M} does at least one of them³. This property of \mathbb{M} will be referred to as *vitality*. Except from vitality no extra assumptions are put on \mathbb{M} .

For a given chain complex $\mathcal{K} \in \mathcal{C}$, *iterated* Morse complex $\mathbb{M}^\infty(\mathcal{K})$ is the fixed point of the iteration $\mathbb{M}(\mathcal{K}), \mathbb{M}^2(\mathcal{K}) = \mathbb{M}(\mathbb{M}(\mathcal{K})), \mathbb{M}^3(\mathcal{K}), \dots$. It is clear that $\|\mathcal{K}\| \geq \|\mathbb{M}(\mathcal{K})\| \geq \|\mathbb{M}^2(\mathcal{K})\| \geq \dots$. Moreover due to the vitality of \mathbb{M} the above inequalities are strict as long as there are some Morse matchings to be made in the intermediate complexes. Therefore, the fixed point $\mathbb{M}^\infty(\mathcal{K})$ is obtained in a finite number of iterations. Below we show that $\mathbb{M}^\infty(\mathcal{K})$ gives an instant information about homology of \mathcal{K} . To achieve this, we will use algebraic version of discrete Morse theory due to Kozlov [25]. It states that two elements A, B can be matched if and only if $\kappa(A, B)$ is invertible. Since in this paper we consider only homology with field coefficients, $\kappa(A, B)$ is always invertible provided it is nonzero. This fact implies the following straightforward lemmas:

Lemma 6.1. For every $A \in \mathbb{M}^\infty(\mathcal{K})$ both boundary and coboundary of A are empty.

Lemma 6.2. $\beta_i(\mathcal{K}) = \|\{A \in \mathbb{M}^\infty(\mathcal{K}) \mid \dim A = i\}\|$.

The proof of the first lemma is a direct consequence of vitality of \mathbb{M} . If there exists $A \in \mathbb{M}^\infty(\mathcal{K})$ with B in (co)boundary, then \mathbb{M} would eventually make a Morse matching between A and B .

The second lemma is a direct consequence of the first one. Once every element has empty boundary, it is a cycle. Once it has empty coboundary, it cannot be a boundary. Therefore every element in $\mathbb{M}^\infty(\mathcal{K})$ generates a homology class of \mathcal{K} . Moreover, due to Section 11.3 in [25] it is

³The simplest example of algorithm that fulfill this requirement is \mathbb{M} which searches for a first possible Morse matching in \mathcal{K} , makes it, and computes the Morse complex.

clear that the homology of $\mathbb{M}^i(\mathcal{K})$ and $\mathbb{M}^{i+1}(\mathcal{K})$ are isomorphic. Therefore the homologies of the complex are preserved through the entire iteration.

As already mentioned, the idea of iteration of Morse complex construction implies that we do not have to construct near optimal Morse complexes. Let n be the cardinality of \mathcal{K} . In the worst case, after $\lfloor n/2 \rfloor$ iterations of Morse complex procedure, an *iterated* Morse complex is obtained⁴. This is a consequence of vitality of \mathbb{M} .

Algorithm 5 Compute homology with iterated Morse decomposition

Input: Initial complex C of dimension d

Output: Betti numbers β_i

```

1: while true do
2:    $M :=$  Build Morse complex of  $C$  (Algorithm 2)
3:   if  $M = C$  then
4:     break
      $C := M$ 
5: for  $i := 0$  to  $d$  do
6:    $\beta_i :=$  number of  $i$ -dimensional cells of  $C$ 

```

Algorithm 5 describes how to compute the Betti numbers using iterated Morse decomposition. An example of the Morse complex construction on a Dunce hat has already been presented in Section 1. In case of the Dunce hat there does not exist a perfect Morse complex. At the end of this section, in Figure 5, we show a simple example of the presented construction, using a sub-optimal algorithm \mathbb{M} to construct Morse complexes with sub-optimal Morse matchings.

7. ITERATED MORSE COMPLEX FOR PERSISTENT HOMOLOGY

In [29] it is shown that when a Morse complex is constructed based on a Morse matching compatible with filtration, persistent homology of the initial complex and the one of the Morse complex are isomorphic. Here we provide a further consequence of this result. Let us take a vital functor \mathbb{M} acting from a category of *filtered* chain complexes to itself. We assume that the Morse matching used to construct $\mathbb{M}(\mathcal{K})$ is compatible with filtration of \mathcal{K} . Filtration values of cells in $\mathbb{M}(\mathcal{K})$ are inherited from the filtration of cells in \mathcal{K} . As in Section 6, we construct $\mathbb{M}^\infty(\mathcal{K})$.

⁴Maximal number of iterations needed is the floor of cardinality of basis of \mathcal{K} divided by two minus sum of all the Betti numbers of \mathcal{K} .

In this section we show that each cell in $\mathbb{M}^\infty(\mathcal{K})$ either creates or kills a feature of nonzero persistence. Therefore, if we want to minimize the number of cells, the resulting complex is the minimal complex encoding persistence of the original complex. An example is presented in Figure 4.

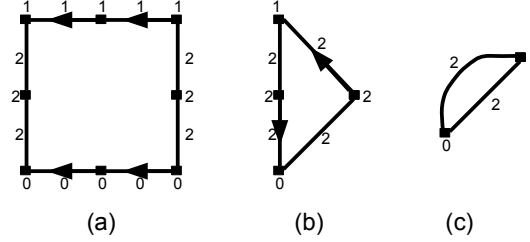


FIGURE 4. (a) The initial complex \mathcal{K} with a few Morse matchings indicated with arrows. (b) Morse complex $\mathbb{M}^1(\mathcal{K})$ obtained from \mathcal{K} . A few possible Morse matchings indicated with arrows. (c) Final Morse complex $\mathbb{M}^2(\mathcal{K})$ obtained from $\mathbb{M}^1(\mathcal{K})$.

In [29] it is only assumed that the filtered chain complex is given at the input. Since at each stage of *iterated* Morse complex computation we have a chain complex, one can iterate further the construction. The main strength of our approach is based on the following novel observation. The resulting complex $\mathbb{M}^\infty(\mathcal{K})$ has the following property: For every $A \in \mathbb{M}^\infty(\mathcal{K})$ and for every b_1, \dots, b_n in boundary of A we have $g(A) > g(b_1), \dots, g(b_n)$. It is because if there existed b_i in the boundary of A such that $g(b_i) = g(A)$, then a Morse matching could be made between A and b_i (since coefficients in a field are used). This would contradict the vitality assumption of \mathbb{M} . Having this simple property of the resulting complex $\mathbb{M}^\infty(\mathcal{K})$ we can now present the main theorem of this section:

Theorem 7.1. *Let $\mathbb{M}^\infty(\mathcal{K})$ be the iterated Morse complex obtained from the initial filtered chain complex \mathcal{K} by iterative construction of Morse complexes using Morse matchings compatible with filtration. Then: \mathcal{K} and $\mathbb{M}^\infty(\mathcal{K})$ have the same persistence. (Correctness)
Every element $A \in \mathbb{M}^\infty(\mathcal{K})$ either starts or terminates a nonzero length persistence interval. (Optimality)*

Proof. Correctness: To show that the algorithm is correct we will apply the Persistence Equivalence Theorem for each iteration:

$$\begin{array}{ccccccc}
\dots & \xrightarrow{j_{l-1}} & H_*(\mathbb{M}_l^i) & \xrightarrow{j_l} & H_*(\mathbb{M}_{l+1}^i) & \xrightarrow{j_{l+1}} & \dots \\
\downarrow & & \downarrow bd_l^{\mathbb{M}} & & \downarrow bd_{l+1}^{\mathbb{M}} & & \downarrow \\
\dots & \xrightarrow{k_{l-1}} & H_*(\mathbb{M}_l^{i+1}) & \xrightarrow{k_l} & H_*(\mathbb{M}_{l+1}^{i+1}) & \xrightarrow{k_{l+1}} & \dots
\end{array}$$

We need to show two things:

- (1) That the vertical maps are isomorphisms on homology level.
- (2) That all squares commute.

First note that the vertical maps send each chain in the input complex to a corresponding chain of the Morse complex. This is equivalent to computing Morse boundaries, as described in Section 5. We remind that to find a corresponding Morse chain we follow appropriate V-paths in the Morse graph.

1) Vertical arrows are isomorphisms: this is a consequence of Theorem 2.1 from [25], which states that the upper chain complex M is decomposed by the Morse construction into an *acyclic* part and the Morse complex having homology isomorphic with M .

2) To prove that the squares commute for each i, l , let us take a chain $c \in \mathbb{M}_l^i = \sum c_j$. We show that $(bd_{l+1}^{\mathbb{M}} \circ j_l)(c) = (k_l \circ bd_l^{\mathbb{M}})(c)$.

Down and right $(k_l \circ bd_l^{\mathbb{M}})$: If c_j is critical, it is unchanged by the vertical map. Otherwise, we follow the paths of the Morse graph to compute the corresponding chain in the Morse complex. Repeating this computation for every c_j , the value of $bd_l^{\mathbb{M}}$ on c is obtained. Moving right with inclusion, the chain remains the same.

Right and down $(bd_{l+1}^{\mathbb{M}} \circ j_l)$: first the chain c is inserted by inclusion into level $l+1$ of filtration, so it is unchanged. But now we move with the vertical arrow, which might be richer on this level, as additional paths enter the Morse graph. Note that since we force the paths to be non-increasing with filtration, $bd_{l+1}^{\mathbb{M}}$ restricted to level l is the same as $bd_l^{\mathbb{M}}$. In other words, any V-path starting at c_j at level at most l can only reach cells of lower or equal filtration values. In particular, it will never reach any critical cell introduced at level $l+1$.

Therefore the two images of chain c are the same and the diagram commutes.

We can apply Persistence Equivalence Theorem and finish the proof that persistent homology is unchanged during our *iterated* Morse complex construction.

Optimality:

We want to show that the simplified complex, $\mathbb{M}^\infty(\mathcal{K})$ contains only significant information, that is *no intervals of persistence zero* are

present. The argument is based on the analysis of the behavior of the standard (left-to-right) matrix-reduction algorithm (Algorithm 1) run on the boundary matrix of the final *iterated* Morse complex $\mathbb{M}^\infty(\mathcal{K})$. The lowest-ones of the reduced matrix directly indicate persistence intervals. Zero columns indicate that a given cell creates an infinite interval.

The argument is inductive with respect to the iteration of the outer loop in the Algorithm 1. Specifically, we consider the first k reduced columns of the matrix. For $k = 0$ this submatrix is empty, so there are no zero-persistence pairs.

Let us assume that the argument holds for some $k \geq 0$.

Suppose by contradiction that there is a zero-length persistence interval generated by a pair (A, b) , where A is the $k + 1$ column. It means that at this stage we have the following situation (dots mark arbitrary entries):

The matrix after $k + 1$ iterations (first $k + 1$ columns are reduced).

$$\begin{array}{cccc}
 & & & k + 1 \\
 & & & A \\
 & \cdot & \cdot & \cdot \\
 & & \cdot & \\
 b & 0 \dots & 0 & 1 \\
 & & & 0 \\
 & & \cdot & 0 \quad \cdot \\
 & & & 0
 \end{array}$$

The matrix after k iterations (first k columns are reduced).

$$\begin{array}{cccc}
 & & & k + 1 \\
 & A1 & & A0 \\
 & \cdot & \cdot & \cdot \quad \cdot \\
 & \cdot & & \cdot \\
 b & \dots & \dots & \cdot \\
 & \cdot & & 0 \\
 & \cdot & \cdot & \cdot \quad \cdot \\
 b1 & 1 & & 1
 \end{array}$$

There are two possibilities:

- (1) During the reduction of $A0$ there were no collisions, so $A0 = A$ is a cell in the complex $\mathbb{M}^\infty(\mathcal{K})$. Then, since $g(A) = g(b)$ and $\kappa(A, b) \neq 0$, it was possible to make a Morse matching between A and b , which gives a contradiction with the fact that all possible Morse matchings compatible with filtration were made in $\mathbb{M}^\infty(\mathcal{K})$.

- (2) A is represented as a sum of the preceding columns, which generated collisions. In this case $A = A_0 + A_1 + \dots + A_n$, for A_0 being the column in the unreduced matrix and A_1, \dots, A_n being columns preceding A_0 in the matrix.

For the proof we need to find the lowest nonzero position of all the columns A_i , $i \in \{1, \dots, n\}$. During the process of reducing a single column the lowest-ones can never increase, therefore the first collision yields the lowest-one we search for. We call the column A_1 and this lowest position b_1 and note that $g(b) \leq g(b_1)$. Also note that b_1 marks the lowest one in the reduced column A_1 and the original column A_0 , so $g(b_1) \leq g(A_1)$ and $g(b_1) \leq g(A_0) = g(A)$.

From the assumption we have $g(A) = g(b)$. From the filtration of the complex we have $g(A_1) \leq g(A)$ and $g(b) \leq g(b_1)$. Putting this together we get: $g(A) = g(b) \leq g(b_1) \leq g(A_1) \leq g(A)$, consequently $g(b_1) = g(A_1)$. This means that there was a zero-persistence pair within the first k columns. This contradicts the inductive assumption.

□

We have the following theorem which is a direct consequence of Theorem 7.1:

Theorem 7.2. *Let \mathcal{K} be the initial filtered chain complex. Let p be the number of finite and k the number of infinite persistence intervals of \mathcal{K} . Then $\|\mathbb{M}^\infty(\mathcal{K})\| = 2p + k$.*

This theorem indicates that the complex $\mathbb{M}^\infty(\mathcal{K})$ is the minimal complex encoding the persistence of the initial complex. We can now apply the matrix reduction method, to get persistence intervals in time $O((2p + k)^3)$. Therefore if the number of persistence intervals is small, this computation can be efficient. As an alternative, we propose a new algorithm presented in Section 8, which relies only on graph operations.

Algorithm 6 describes the simplification procedure. To compute persistence based on simplified complex C use Algorithm 1. We want to point out that there is no obvious way of relaxing the condition $g(A) = g(M(A))$ for matched elements. See the Appendix for more details.

At the end of this section, in Figure 5 we present an example of the *iterated* Morse complex construction.

Algorithm 6 Simplification for persistence computations

Input: Initial filtered complex C of dimension d

Output: Simplified complex C , having the same persistent homology

- 1: **while** true **do**
 - 2: $M :=$ Build Morse complex of C using only matchings compatible with filtration
 - 3: **if** $M = C$ **then**
 - 4: break
 - $C := M$
-

8. PERSISTENCE INTERVALS VIA ITERATED MORSE APPROACH.

In this section we compute persistence intervals using the *iterated* Morse complex approach. It will be shown that the presented approach can be interpreted as a variation of matrix-reduction algorithm. It is however based on graph theory, rather than matrix algebra.

We stress that in this section, unlike previous one, we allow to make Morse matchings between elements having different level of filtration. Therefore, we will construct Morse matchings that are *not* compatible with filtration in a sense that element A can be matched with b if $g(A) \geq g(b)$.

Algorithm 7 is used to compute persistence intervals of $\mathbb{M}^\infty(\mathcal{K})$.

We want to point out that the Morse boundary procedure is always performed on the whole complex $\mathbb{M}^\infty(\mathcal{K})$ even if the matchings are made on a proper subcomplex $\mathbb{M}_i^\infty(\mathcal{K})$. We start from the complex $\mathbb{M}_f^\infty(\mathcal{K})$. Since $\mathbb{M}^\infty(\mathcal{K})$ is an iterated Morse complex, is clear that in $\mathbb{M}_{f-1}^\infty(\mathcal{K})$ and $\mathbb{M}_f^\infty(\mathcal{K}) \setminus \mathbb{M}_{f-1}^\infty(\mathcal{K})$ there are no Morse matchings to be made. But in $\mathbb{M}_f^\infty(\mathcal{K})$ there can be a Morse matching (A, b) such that $A \in \mathbb{M}_f^\infty(\mathcal{K}) \setminus \mathbb{M}_{f-1}^\infty(\mathcal{K})$ and $b \in \mathbb{M}_{f-1}^\infty(\mathcal{K})$. This means that $b \in \mathbb{M}_{f-1}^\infty(\mathcal{K})$ is a homology generator in $\mathbb{M}_{f-1}^\infty(\mathcal{K})$ which is killed in $\mathbb{M}_f^\infty(\mathcal{K})$ (since the matching (A, b) can be made without changing homology of $\mathbb{M}_f^\infty(\mathcal{K})$). Making such a matching indicates a persistence interval generated by the pair (A, b) , since the homology class generated by b is killed by A . We assume that all possible matchings in $\mathbb{M}_f^\infty(\mathcal{K})$ are made (by iterating Morse complex construction) before proceeding to $\mathbb{M}_{f+1}^\infty(\mathcal{K})$.

When processing complex $\mathbb{M}_i^\infty(\mathcal{K})$ we assume that in $\mathbb{M}_{i-1}^\infty(\mathcal{K})$ no more Morse matchings can be made. We are searching for matchings (A, b) such that $A \in \mathbb{M}_i^\infty(\mathcal{K}) \setminus \mathbb{M}_{i-1}^\infty(\mathcal{K})$ and $b \in \mathbb{M}_{i-1}^\infty(\mathcal{K})$. If two or more elements b_1, \dots, b_n can be matched with A we always choose b_j having maximal value of filtration. Morse matching (A, b) indicates an interval generated by (A, b) , since b generates nontrivial homology class

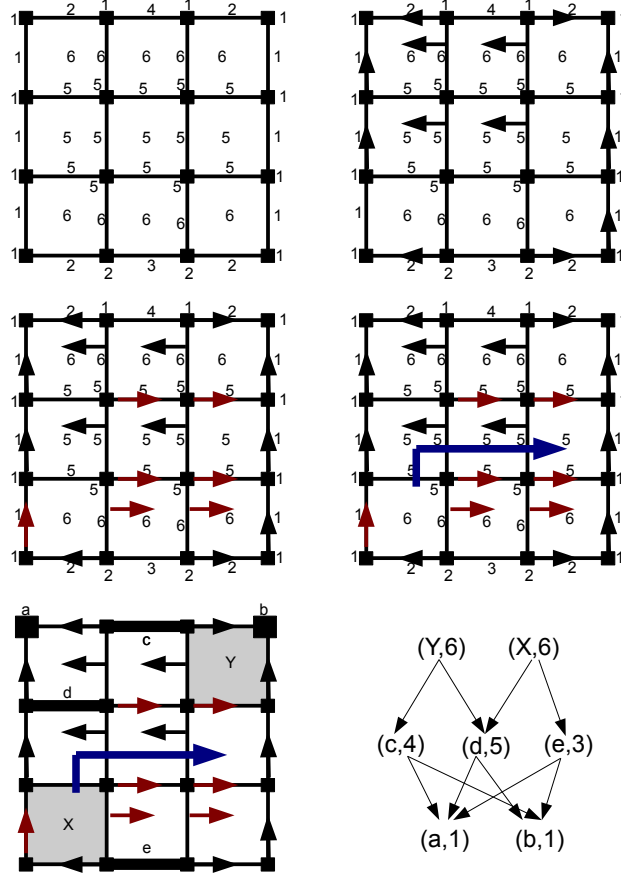


FIGURE 5. On the top left the initial filtered complex \mathcal{K} is depicted. On the top right, the first iteration, $\mathbb{M}^1(\mathcal{K})$ of the Morse complex compatible with filtration construction. On the middle left with red arrows the second iteration $\mathbb{M}^2(\mathcal{K})$, and on the middle right the third and final iteration of $\mathbb{M}^\infty(\mathcal{K})$ construction. On the bottom left, the cells of $\mathbb{M}^\infty(\mathcal{K})$ are named, and on bottom right the boundary relation is presented in a form of diagram. The levels indicate the gradation – vertices a, b at the bottom, edges c, d, e in the middle and faces X and Y at the top. The numbers indicate filtration values of elements and arrows – boundary relation.

in the level of $g(b)$, which becomes trivial or identical to some other class born earlier in the level of $g(A)$.

Algorithm 7 Compute persistent homology via Iterated Morse complex.

Input: Filtered iterated Morse complex $C = \mathbb{M}^\infty(\mathcal{K})$.

Output: Persistence intervals of $\mathbb{M}^\infty(\mathcal{K})$.

```

1:  $S :=$  empty multiset of persistence intervals;
2: int  $f :=$  second filtration level of  $C$ ;
3: int  $l :=$  last filtration level of  $C$ ;
4:  $M$  is a vital strategy to make Morse matchings. Matchings between
   elements of different filtrations are allowed in  $M$ . If element  $A$  can be
   matched with two (or more) elements  $b_1, b_2, \dots$ , we match it with the one
   with maximal filtration value.
5: for int  $i := f$  to  $l$  do
6:   while true do
7:     Construct Morse matching on  $C_i$  using  $M$  (remark: for every  $(A, b)$ 
       matched by  $M$  we have  $g(b) < g(A) = i$ ).
8:     if Nothing was matched by  $M$  then
9:       break
10:    for Every  $(A, b)$  matched by  $M$  do
11:       $S := S \cup [g(b), g(A)]$ ;
12:     $C :=$  Morse complex on  $C$  constructed based on Morse matching
        $M$ ;
13: for every cell  $c$  in complex  $C$  do
14:    $S := S \cup [g(c), \infty]$ ;
15: return  $S$ ;
```

When all the possible matchings are made, in the last for loop the unmatched elements are found. They generate infinite persistence intervals.

It is clear that the levels of filtration need to be processed in order. See the Appendix for more details.

Let us now show that the presented technique can be interpreted in terms of the standard algebraic Algorithm 1.

Theorem 8.1. *Let \mathcal{K} be a filtered chain complex. Let $\mathbb{M}^\infty(\mathcal{K})$ be the iterated Morse complex described in Section 6. Then the persistence intervals of \mathcal{K} and the intervals obtained from $\mathbb{M}^\infty(\mathcal{K})$ by the described algorithm are the same.*

Proof. From Theorem 7.1 it is clear that the persistence intervals of \mathcal{K} and $\mathbb{M}^\infty(\mathcal{K})$ are the same. Let (A, b) be the first Morse matching made by the presented algorithm (i.e. there does not exist a possible matching (A', b') such that $g(A') < g(A)$ and there does not exist b' , a face of A with $g(b') > g(b)$).

Let $\mathbb{M}_{(A,b)}(\mathbb{M}^\infty(\mathcal{K}))$ be the Morse complex obtained from $\mathbb{M}^\infty(\mathcal{K})$ by making a Morse matching (A, b) . To prove the theorem it suffices to show that the multiset of persistence intervals of $\mathbb{M}^\infty(\mathcal{K})$ minus the interval $[g(A), g(b)]$ is equal to the multiset of persistence intervals of $\mathbb{M}_{(A,b)}(\mathbb{M}^\infty(\mathcal{K}))$ ⁵

First let us remind how the procedure to compute Morse boundary works and how it is interpreted in matrix-reduction algorithm. The details can be found in [13]. Let us assume just one Morse matching, (A, b) , is made. And also that b is in the boundary of A, A_1, \dots, A_n . Then boundaries of A_1, \dots, A_n need to be changed in the following way: $\partial A_i = \partial A_i \setminus b \cup \partial A \setminus b$ i.e. b is replaced in boundary of A_i with boundary of A excluding b , as in Figure 6.

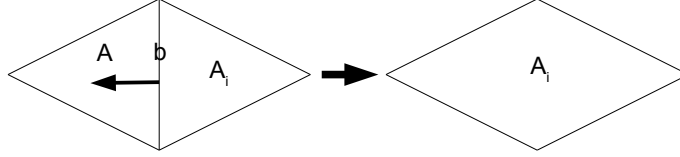


FIGURE 6. Illustration how performing a single pairing changes the complex.

Now, suppose Algorithm 1 is run on the complex $\mathbb{M}^\infty(\mathcal{K})$. Without loss of generality we may assume that the column A is the first nonzero column in the matrix. Since there cannot be a collision there, Algorithm 1 leaves the column A unchanged and the interval $[g(b), g(A)]$ is obtained in dimension of b – as in the case of the algorithm presented in this section. But, row b may cause some collisions later in the course of execution of Algorithm 1. Suppose the first collision in Algorithm 1 occurs:

$$\begin{array}{cc}
 & A & A_i \\
 & \cdot & \cdot \\
 & \cdot & \cdot \\
 b & \cdot \quad \cdot & 1 & 1 \\
 & 0 & 0 \\
 & \cdot & \cdot \\
 & 0 & 0
 \end{array}$$

To remove this collision, Algorithm 1 performs the addition $A_i := A_i + A$. In the algorithm presented in this section, when processing

⁵This follows from the fact that $\mathbb{M}_{(A_i,b_i),\dots,(A_n,b_n)} = \mathbb{M}_{(A_i,b_i)} \circ \dots \circ \mathbb{M}_{(A_n,b_n)}$. Easy proof is left for the reader.

cell A the matching (A, b) was made and the Morse boundaries were computed. As one can see, the computations of Morse boundary after the matching (A, b) is simply equivalent to summing $A_i := A_i + A$ for all A_i having b in boundary. Therefore all the future collisions caused by b are resolved. Consequently making a Morse matching is equivalent to resolving all the future collisions at once. This simple observation proves the theorem. \square

In Figure 7 an illustration of the presented procedure on the complex $\mathbb{M}^\infty(\mathcal{K})$ from Figure 5 is given.

We want to point out that this approach promises to parallelize well. The details will be presented in a more technical paper [9].

Moreover, the approach will be as scalable as the implementations of the underlying graph algorithms. Therefore, using the available, mature libraries, we hope to achieve good practical performance.

We also want to point out that Algorithm 7 can be used with minor modification for initial filtered complex \mathcal{K} . Easy changes are left for the reader.

9. CONCLUSIONS

In our opinion the presented technique has several advantages, comparing to the standard way of computing homology and persistence:

- (1) It is combinatorial, does not require any algebraic matrix operation.
- (2) It is based on graph theory – we can use efficient algorithm (exact and approximate) and their existing implementations – in particular libraries for distributed graph operations [27].
- (3) It is intuitive – it is easy to visualize the process of homology computations.

We are aware of some drawbacks and complications of our method:

- (1) There may exist bad cases, where the complexity will be unsatisfactory.
- (2) This technique might not be suitable for cubical data. Existing methods [16, 37] rely on the compact representation of cubical grids. In our case the complex need not be a cubical complex after the first iteration, preventing us from storing it efficiently.

The presented techniques can be used to formalize and generalize homology-preserving properties of graph pyramids used in image recognition [31]. Moreover they can be beneficial in verified homology computation [20] by avoiding matrix operations which are costly to verify automatically.

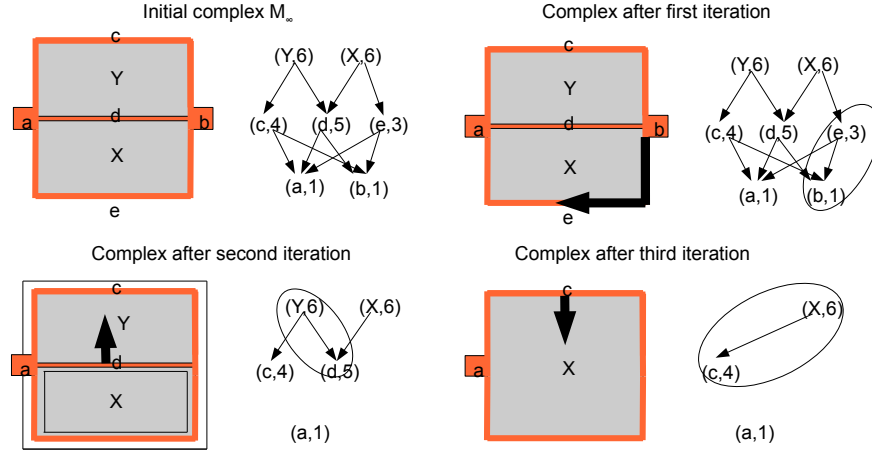


FIGURE 7. First complex on the top left – complex $M^\infty(\mathcal{K})$ from Figure 5. The complex on the top right – on filtration value 3 the first possible Morse matching between b and e appears (indicated on the left with arrow, and with ellipse on the right) is made. When the matching is constructed, the persistence interval $[1, 3]$ in dimension zero is reported. Third complex on the bottom left is obtained as a result. There, on the level 6 a matching between Y and d can be made (we want to point out that there is also possible matching between Y and c , however filtration value of d is higher than filtration value of c). After the matching, the persistent interval $[5, 6]$ in dimension one is reported. The final complex on the bottom right – the last possible matching between X and c is made. After the matching, the persistent interval $[4, 6]$ in dimension 1 is reported. The remaining cell in the complex is the vertex a it correspond to infinite persistence interval $[1, \infty]$ in dimension zero.

Summarizing, in this paper a novel technique of homology and persistent homology computations has been presented. It indicates that problem of computing (persistent) homology can be solved by iteratively applying standard graph algorithms. We hope that this approach will lead to a scalable implementation for (persistent) homology computations.

ACKNOWLEDGMENTS

The authors would like to thank Herbert Edelsbrunner and Ulrich Bauer for their valuable comments and suggestions. Both authors are supported by Google Research Awards program. We thank Marian Mrozek for the supervision of this project. PD. is partially supported by grant IP 2010 046370. HW. is partially supported by Foundation for Polish Science IPP Programme "Geometry and Topology in Physical Models".

10. APPENDIX

10.1. Example of matrix reduction computations. A simple example of persistence intervals computations with the Algorithm 1 are presented in Figure 8. On the left, the initial complex. We assume that the filtration value for every vertex is 0. The filtration value of edges are given in the picture. On the upper left, the initial boundary matrix. As one can see, the only collision is between columns cd and bd . Therefore we have column $cd = cd + bd$ on the upper right. Then a collision between cd and ac appears and we set $cd = cd + ac$ on the lower left. There again we have a collision cd with ab which is removed in the lower right by setting $cd = cd + ab$. On the matrix in lower right there are no more collisions, therefore we can read persistence intervals out of it. Lowest one in column ac indicates that edge ac kills connected component created by c , which gives an interval $[0, 1]$ in dimension 0. Analogously lowest one in column bd induces an interval $[0, 1]$ in dimension 0. Lowest one in column ab indicates that edge ab kills a connected component created in b , which gives an interval $[0, 2]$ in dimension 0. Zero column cd induces an infinite interval $[3, \infty]$ in dimension one.

10.2. Relaxing the tolerance. In Figure 9 we show that if one makes a Morse matchings between elements $(A, M(A))$ such that $|g(A) - g(M(A))| \leq \epsilon$, one may get arbitrarily large differences in the output persistence intervals.

10.3. Processing order. In Figure 10 it is shown what happens if we do not proceed in Algorithm 7 in the filtration order.

REFERENCES

- [1] R. Ayala, D. Fernandez-Ternero, J. A. Vilches, *Perfect discrete Morse functions on 2-complexes*, Pattern Recognition Letters - PRL, DOI: 10.1016/j.patrec.2011.08.011.

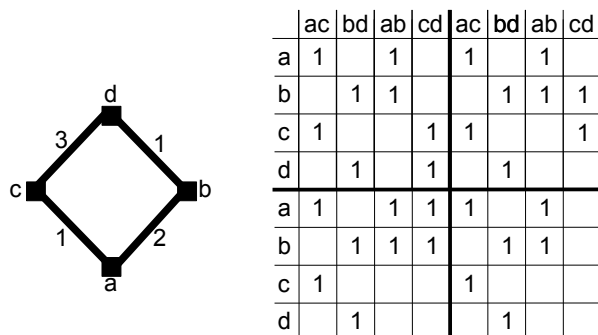


FIGURE 8. Example of matrix reduction computations

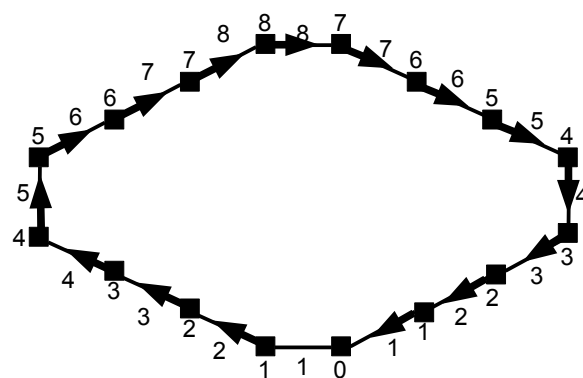


FIGURE 9. Let us assume the matchings are made with tolerance $\epsilon = 1$ (i.e. one is allowed to do Morse matchings between elements having the absolute value of difference of filtration values not greater than ϵ). In the original complex in dimension 1 we have a persistence interval $[8, \infty]$, while in the Morse complex (matchings are indicated with arrows) we have a persistence interval $[1, \infty]$. It is clear that by enlarging the circle, one may get arbitrary large difference in persistence intervals.

- [2] U. Bauer, C. Lange, M. Wardetzky, *Optimal topological simplification of discrete functions on surfaces*, Discrete & Computational Geometry 47:2 (2012), 347377.
- [3] The CAPD library, capd.i.i.uj.edu.pl
- [4] C. Chen, M. Kerber, *An output-sensitive algorithm for persistent homology*, 27th Annual Symposium on Computational Geometry (SoCG 2011).

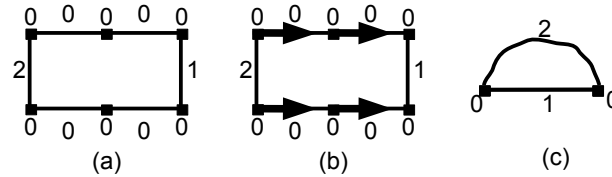


FIGURE 10. (a) The initial complex \mathcal{K} . Numbers indicate filtration values of elements. (b) Directed path on \mathcal{K} compatible with filtration. (c) The reduced complex $\mathbb{M}^\infty(\mathcal{K})$. Suppose $\mathbb{M}_2^\infty(\mathcal{K}) \subseteq \mathbb{M}^\infty(\mathcal{K})$ is considered first in the described procedure. Then a Morse matching can be made between one dimensional cell having filtration value 2 and one of vertices of filtration value 0 (and an interval $[0, 2]$ in dimension 0 is reported). Consequently no more matching can be made at any level of filtration and an interval $[0, \infty]$ in dimension 0 and $[1, \infty]$ in dimension 1 are reported. This is wrong, since the true output is $[0, 1]$, $[0, \infty]$ in dimension 0 and $[2, \infty]$ in dimension 1. Therefore indeed the complexes have to be processed in the order of filtration values.

- [5] C. Chen, M. Kerber, *Persistent homology computation with a twist*, 27th European Workshop on Computational Geometry (EuroCG 2011), 2011.
- [6] Chomp library, chomp.rutgers.edu
- [7] C. J. A. Delfinado, H. Edelsbrunner, *An incremental algorithm for Betti numbers of simplicial complexes*, Proceeding SCG '93 Proceedings of the ninth annual symposium on Computational geometry. pp. 232-239.
- [8] Dionysus library <http://www.mrzv.org/software/dionysus/>
- [9] P. Dłotko, M. Robinson, *Distributed homology computations by local Morse pairings*, in preparation.
- [10] H. Edelsbrunner, J. Harer, *Computational Topology. An Introduction*. Amer. Math. Soc., Providence, Rhode Island, 2010.
- [11] H. Edelsbrunner, D. Letscher and A. Zomorodian. Topological persistence and simplification. *Discrete Comput. Geom.* 28 (2002), 511-533.
- [12] G. Even, J. Noar, B. Schieber and M. Sudan, *Approximating minimum feedback sets and multicuts in directed graphs*, *Algorithmica*, 20, (1998) 151-174.
- [13] R. Forman, *Morse theory for cell complexes*, *Advances in Mathematics*, 134:90-145, 1998.
- [14] P. Frosini, *A distance for similarity classes of submanifolds of a Euclidean space*, *Bulletin of the Australian Mathematical Society*, 42, 3 (1990), 407-416.
- [15] R. Gonzalez-Daz, M. J. Jimenez, B. Medrano, P. Real *A tool for integer homology computation: lambda-AT-model*, *Image Vision Comput.* 27(7): 837-845 (2009).

- [16] D. Günther, J. Reininghaus, H. Wagner, I. Hotz, *Efficient Computation of 3D Morse-Smale Complexes and Persistent Homology using Discrete Morse Theory*, The Visual Computer 28(10), 959-969 (2012).
- [17] A. Gyulassy, V. Natarajan, V. Pascucci, B. Hamann, *Efficient Computation of Morse-Smale Complexes for Three-dimensional Scalar Functions*, IEEE Transactions on Visualization and Computer Graphics, Vol. 13 , Issue: 6, pp. 1440 - 1447, 2007.
- [18] S. Harker, K. Mischaikow, M. Mrozek, V. Nanda, H. Wagner, M. Juda, P. Dlotko, *The Efficiency of a Homology Algorithm based on Discrete Morse Theory and Coreductions*, 3rd International Workshop on Computational Topology in Image Context, November 2010, Chipiona, Spain Proceedings (Roco Gonzalez Daz Pedro Real Jurado (Eds.)) ISSN: 1885-4508.
- [19] S. Harker, K. Mischaikow, M. Mrozek, V. Nanda, *Discrete Morse Theoretic Algorithms for Computing Homology of Complexes and Maps*, submitted to Foundations of Computational Mathematics.
- [20] J. Heras, M. Dénès, G. Mata, A. Mörtberg, M. Poza, V. Siles, *Towards a certified computation of homology groups for digital images*, Lecture Notes in Computer Science Volume 7309, 2012, pp 49-57.
- [21] P. Hersh, *On optimizing discrete Morse functions*, (English summary) Adv. in Appl. Math. 35 (2005), no. 3, 294322.
- [22] M. Joswig , M. E. Pfetsch , *Computing optimal Morse matchings*, SIAM Journal on Discrete Mathematics, Vol. 20 Issue 1, pp. 11 – 25, 2006.
- [23] Jplex library <http://comptop.stanford.edu/u/programs/jplex/>
- [24] Kenzo program <http://www-fourier.ujf-grenoble.fr/~sergerar/Kenzo/>
- [25] D. Kozlov, *Combinatorial Algebraic Topology*, Springer-Verlag, 2007.
- [26] T. Lewiner, H. Lopes, G. Tavares, *Optimal discrete Morse functions for 2-manifolds*, Computational Geometry: Theory and Applications 26(3): pp. 221-233.
- [27] G. Malewicz, M. Austern, A. Bik, J. Dehnert, I. Horn, N. Leiser, G. Czajkowski, *Pregel: a system for large-scale graph processing*, Proceedings of the 2010 ACM SIGMOD International Conference on Management of data, pp. 135-146, 2010.
- [28] N. Milosavljevic, D. Morozov, P. Skraba, *Zigzag Persistent Homology in Matrix Multiplication Time*, Proceedings of the 27th Annual Symposium on Computational Geometry (SCG'11).
- [29] K. Mischaikow, V. Nanda, *Morse theory for filtrations and efficient computation of persistent homology*, Discrete & Computational Geometry submitted.
- [30] J. R. Munkres, *Elements of Algebraic Topology*, Addison-Wesley, Reading, MA, 1984.
- [31] S. Peltier, A. Ion, Y. Haxhimusa, W. G. Kropatsch, G. Damiand, *Computing Homology Group Generators of Images Using Irregular Graph Pyramids*, GbRPR 2007: 283-294.
- [32] Perseus library, <http://www.math.rutgers.edu/~vidit/perseus.html>
- [33] V. Robins, *Towards computing homology from finite approximations*, Topology Proceedings, 24:503-532, (1999).
- [34] V. Robins, P.J. Wood, A.P. Sheppard, *Theory and algorithms for constructing discrete Morse complexes from grayscale digital images*, IEEE Transactions on pattern analysis and machine intelligence, (2010) accepted.

- [35] V. de Silva, R. Ghrist, *Coverage in sensor networks via persistent homology*, Algebraic and Geometric Topology, 2007.
- [36] A. Storjohann, *Near Optimal Algorithms for Computing Smith Normal Form of Integer Matrices*, Proceedings of the 1996 international symposium on symbolic and algebraic computation, ISAAC 1996, (1996), 267-274.
- [37] H. Wagner, C. Chen, E. Vuçini, *Efficient computation of persistent homology for cubical data*, Proceedings of the 4th Workshop on Topology-based Methods in Data Analysis and Visualization (TopoInVis 2011).
- [38] H. Wagner, P. Dłotko, M. Mrozek, *Computational topology in text mining*, in M. Ferri et al. (Eds.): CTIC 2012, LNCS 7309, pp. 117127, 2012.
- [39] A. Verri, C. Uras, P. Frosini, M. Ferri, *On the use of size functions for shape analysis*, Biological Cybernetics, 70, (1993), 99-107.

INSTITUTE OF COMPUTER SCIENCE, JAGIELLONIAN UNIVERSITY, KRAKOW,
POLAND

E-mail address: pawel.dlotko@ii.uj.edu.pl

INSTITUTE OF COMPUTER SCIENCE, JAGIELLONIAN UNIVERSITY, KRAKOW,
POLAND

E-mail address: hubert.wagner@ii.uj.edu.pl