The Efficiency of a Homology Algorithm based on Discrete Morse Theory and Coreductions (Extended Abstract)

Shaun Harker[1], Konstantin Mischaikow[1], Marian Mrozek[2]
Vidit Nanda[1], Hubert Wagner[2], Mateusz Juda[2] and Paweł Dłotko[2]

[1] Rutgers University, Department of Mathematics, Piscataway, New Jersey, USA
[2] Uniwersytet Jagiellonski, Instytut Informatyki, Kraków, Poland.
sharker@math.rutgers.edu, mischaik@math.rutgers.edu, mrozek@ii.uj.edu.pl,
velnias@gmail.com, pawel.dlotko@ii.uj.edu.pl,
mateusz.juda@gmail.com, hub.wag@gmail.com

Abstract Two implementations of a homology algorithm based on the Forman's discrete Morse theory combined with the coreduction method are presented. Their efficiency is compared with other implementations of homology algorithms.

Keywords: efficient homology algorithms, Discrete Morse theory, coreductions

1 Introduction

Homology theory is already more than 100 years old but, despite its origins in the theory of differential equations, until late 20th century it was considered a purely theoretical part of mathematics. However, in recent years homological methods found their way into the applied sciences. This causes growing demand for efficient homology algorithms. In principle, homology is an algebraic method and Smith diagonalization of integer matrices is the classical tool for computing homology. Unfortunately, because the size of inputs in many practical problems is hundred thousands and more, Smith diagonalization, which is supercubical, is to slow for many applications. Among such applications are in particular many problems in material science [6, 13, 21, 26] and imaging [5, 12, 24, 25, 28, 27]. Based on these limitations of the Smith normal form algorithm, it is not surprising that more efficient algorithms for the computation of homology have been developed over the years, see for example [7, 8, 9, 11, 16, 17, 15, 19, 20, 22], as well as the references therein.

Among the many ways to overcome this problem the fundamental one is to try to improve the efficiency of Smith diagonalization [8]. However, when computing homology of spaces there is an alternative based on the so called reduction methods. The idea is to transform the original input into one which is considerably smaller and still has the same homology, and then to apply the Smith normal form algorithm as a last step. A variety of different reduction methods have been proposed such as elementary reductions, acyclic subspace reductions [20], as well as coreductions [19]. Implementations of these algorithms are available from [1, 2].

Several researchers agree that the discrete Morse theory for finite, regular CW complexes, developed by R. Forman [10], may potentially be used as a powerful reduction method (see in particular [23]). However, since the ideas are not algebraic but geometric, their implementation is not straightforward. Maybe this is the reason that although the discrete Morse theory is already 12 years old, we know of no implementation, and particularly no competitive implementation, of a homology algorithm based on the discrete Morse theory.

The aim of this paper is to show that in practice applying the discrete Morse theory to computing homology of spaces indeed leads to very efficient algorithms. We present two recently

coded implementations of such homology algorithms in which the construction of the discrete Morse function utilizes the concept of coreductions introduced in [19]. Here we briefly describe the algorithm and show the efficiency of the two implementations. Theoretical foundations of the presented method, including a computational approach to maps induced in homology based on discrete Morse theory will be presented in [14].

2   Discrete Morse Theory

We begin with briefly recalling the fundamentals of the discrete Morse theory as developed by R. Forman [10].

Let $\mathcal{K}$ be the collection of cells of a finite, regular, CW complex which we will also denote by $\mathcal{K}$. We denote by $\dim \sigma$ the dimension of the cell $\sigma \in \mathcal{K}$ and we write $\mathcal{K}_n$ for the subcollection of cells of dimension n. The *adherence relation* is defined by

$$\tau \prec \sigma \iff \tau \subset \sigma \text{ and } \dim \tau = \dim \sigma - 1$$

and the *adherence digraph* by

$$(\mathcal{K}, \{ (\sigma, \tau) \mid \tau \prec \sigma \}).$$

We also write

$$\begin{aligned}
\mathrm{bd}_{\mathcal{K}}\, \sigma &:= \{ \tau \mid \tau \prec \sigma \}, \\
\mathrm{cbd}_{\mathcal{K}}\, \sigma &:= \{ \rho \mid \sigma \prec \rho \}.
\end{aligned}$$

Forman [10] defines $f : \mathcal{K} \to \mathbb{Z}$ to be a *discrete Morse function* if for every $\sigma \in \mathcal{K}$

$$c^+(\sigma) := \mathrm{card} \{ \rho \in \mathrm{cbd}\, \sigma \mid f(\rho) \leq f(\sigma) \} \leq 1$$

and

$$c^-(\sigma) := \mathrm{card} \{ \tau \in \mathrm{bd}\, \sigma \mid f(\tau) \geq f(\sigma) \} \leq 1.$$

A cell $\sigma \in \mathcal{K}$ is *critical* if $c^+(\sigma) = c^-(\sigma) = 0$. A *discrete vector field V* on $\mathcal{K}$ is a collection of pairs $(\tau, \sigma) \in \mathcal{K}^2$ such that $\tau \prec \sigma$ and each element of $\mathcal{K}$ is in at most one pair in V. The V-digraph of $\mathcal{K}$ is the adherence graph of $\mathcal{K}$ with the direction reversed on the elements of V.

Theorem 2.1 *(Forman, 1998) If $f : \mathcal{K} \to \mathbb{Z}$ is a discrete Morse function then*

$$\{ (\tau, \sigma) \in \mathcal{K}^2 \mid \tau \prec \sigma \text{ and } f(\tau) \geq f(\sigma) \}$$

is a discrete vector field on $\mathcal{K}$, called the gradient vector field *of f.*

Theorem 2.2 *(Forman, 1998) A discrete vector field V is the gradient vector field of a discrete Morse function if and only if the V-digraph of $\mathcal{K}$ is acyclic, ie. contains no non-trivial directed loops.*

We now assume that a discrete Morse function f on $\mathcal{K}$ is given and V is the associated gradient vector field. We introduce the following sets

$$\begin{aligned}
Q(\mathcal{K}) := Q(\mathcal{K}, V) &:= \{ \tau \in \mathcal{K} \mid \exists \sigma \in \mathcal{K} \, (\tau, \sigma) \in V \}, \\
K(\mathcal{K}) := K(\mathcal{K}, V) &:= \{ \sigma \in \mathcal{K} \mid \exists \tau \in \mathcal{K} \, (\tau, \sigma) \in V \}, \\
A(\mathcal{K}) := A(\mathcal{K}, V) &:= \mathcal{K} \setminus Q(\mathcal{K}, V) \setminus K(\mathcal{K}, V).
\end{aligned}$$

Note that $A(\mathcal{K})$ consists precisely of critical elements of $\mathcal{K}$. The fact that V is a gradient vector field causes that the elements of $Q(\mathcal{K})$ and $K(\mathcal{K})$ appear in pairs, which allows us to define the pairing:

$$Q(\mathcal{K}) \cup K(\mathcal{K}) \ni \sigma \to \sigma^* \in Q(\mathcal{K}) \cup K(\mathcal{K})$$

such that if $\sigma \in Q(\mathcal{K})$ then $\sigma^* \in K(\mathcal{K})$ and if $\sigma \in K(\mathcal{K})$ then $\sigma^* \in Q(\mathcal{K})$.

Let $C(\mathcal{K})$ be the chain complex of $\mathcal{K}$ (see [18, Chapter IX.5]) and let $\langle \cdot, \cdot \rangle$ denote the scalar product associated with the free basis $\mathcal{K}$. If $\tau \prec \sigma$, then τ has a non-zero coefficient in the boundary chain of σ, called the incidence number of the pair (σ, τ). We denote it by $\kappa(\sigma, \tau)$. The *weight* of a pair $(\sigma, \tau) \in \mathcal{K}^2$ is defined by

$$w(\sigma, \tau) := \begin{cases} \kappa(\sigma, \tau) & \tau \prec \sigma \\ -\kappa(\tau, \sigma)^{-1} & \tau \succ \sigma \\ 0 & \text{otherwise.} \end{cases}$$

A *path* from $\sigma \in \mathcal{K}$ to $\tau \in \mathcal{K}$ is a sequence $\alpha = (\alpha_0, \alpha_1, \ldots, \alpha_n) \in \mathcal{K}^{n+1}$ such that $(\alpha_{i-1}, \alpha_i) \in V$, $\alpha_0 = \sigma$ and $\alpha_n = \tau$. The path α is *alternating* if

$$\dim \alpha_i = \dim \alpha_0 - (i \mod 2).$$

We denote the set of all paths from $\sigma \in \mathcal{K}$ to $\tau \in \mathcal{K}$ by $P_V(\sigma, \tau)$ and the set of alternating paths from $\sigma \in \mathcal{K}$ to $\tau \in \mathcal{K}$ by $P_V^a(\sigma, \tau)$. The *weight of a path* $\alpha = (\alpha_0, \alpha_a, \ldots, \alpha_n)$ is given by

$$w(\alpha) := \prod_{i=1}^{n} w(\alpha_{i-1}, \alpha_i).$$

We define the *Morse complex* $(M, \Delta) := \left(M_q(\mathcal{K}, V), \Delta_q(\mathcal{K}, V)_{q \in \mathbb{Z}} \right)$ of a gradient vector field V on $\mathcal{K}$ by

$$M_q := \{ \sigma \in A(\mathcal{K}) \mid \dim \sigma = q \} \tag{1}$$

$$\langle \Delta_q \sigma, \tau \rangle := \sum_{\alpha \in P_V^a(\sigma, \tau)} w(\alpha). \tag{2}$$

The central theorem of the discrete Morse theory is the following theorem.

Theorem 2.3 *(Forman, 1998)*

$$H_*(\mathcal{K}) \cong H_*(M, \Delta).$$

Now, in many applied situations one expects the Morse complex (M, Δ) to be much smaller than the original complex $\mathcal{K}$. Therefore, computing the homology of the Morse complex instead of the original complex may substantially reduce the amount of necessary algebraic computations and consequently lead to a particularly efficient homology algorithm. However, for this to work one needs an efficient method of constructing a discrete Morse function.

3 Discrete Morse Function Construction based on coreductions as a tool for computing homology.

The coreduction homology algorithm, in the sequel referred to as CR homology algorithm, presented in [19] is among the most efficient homology algorithms for cubical sets available so far. Recall that a pair $(\sigma, \tau) \in \mathcal{K}^2$ is called a coreduction pair if $\mathrm{bd}_{\mathcal{K}} s = \{t\}$. It is proved in [19] that in the context of so called S-complexes (see [19, Section 2]) a coreduction pair may be removed from $\mathcal{K}$ without changing the homology of $\mathcal{K}$. Roughly speaking, the CR homology algorithm consists in removing coreducton pairs as long as they are available. However, one easily sees that in a compact CW complex no coreduction pairs exists. Nevertheless, if $\mathcal{K}$ is a CW complex and v is a vertex of $\mathcal{K}$ then $\mathcal{K} \setminus \{v\}$ is an S-complex which admits coreduction pairs. Since we have

$$H(\mathcal{K} \setminus \{v\}) \cong H(\mathcal{K}, \{v\}) \cong \bar{H}(\mathcal{K}) \tag{3}$$

```
Algorithm 3.1 discreteMorseFunction(CW complex 𝒦)
  V := L := A := K := Q := dom f := ∅;
 while 𝒦 ≠ ∅  do
     if L = ∅ then
         n := min { q | 𝒦_q ≠ ∅ };
         move an element r from 𝒦_n to A;
         define f(r) as 1 + max { f(r') | r' ∈ dom f, r' ≺ r };
         enqueue(L, cbd_𝒦 r);
     else
         s:=dequeue(L);
         if bd_𝒦 s = {t}  then
             move s from 𝒦 to K and t from 𝒦 to Q;
             insert (s, t) to V;
             define f(s) and f(t) as 1 + max { f(s') | s' ∈ dom f, s' ≺ s };
             enqueue(L, cbd_𝒦 t);
         else if bd_𝒦 s = ∅  then
             enqueue(L, cbd_𝒦 s);
         endif;
     endif;
 endwhile ;
```

Table 1: Discrete Morse Function by coreduction method

with $\bar{H}(\mathcal{K})$ standing for the reduced homology of $\mathcal{K}$ and

$$H_n(\mathcal{K}) = \begin{cases} \mathbb{Z} \oplus \bar{H}_0(\mathcal{K}) & \text{if } n = 0, \\ \bar{H}_n(\mathcal{K}) & \text{otherwise,} \end{cases} \tag{4}$$

it is sufficient to compute $H(\mathcal{K} \setminus \{v\})$ instead of $H(\mathcal{K})$. We refer the reader to [19] for details.

The efficiency of the CR homology algorithm crucially depends on the amount of cells left after all available coreduction pairs are removed. One can show that if after removing a vertex from a connected CW-complex $\mathcal{K}$ all available coreduction pairs have been removed then the remaining complex has no vertices. Then, often removing an edge causes that the coreduction pairs reappear. However, in this case there is no simple relation between the homology before and after removing an edge as in the case of formulas (3) and (4). But the coreduction technique may be used to construct a discrete Morse function on $\mathcal{K}$. The respective algorithm is presented in Table 1.

Theorem 3.2 *Algorithm 3.1 runs in linear time.*

The coreduction based, discrete Morse theory homology algorithm, referred in the sequel as *DMT algorithm*, consists in applying Algorithm 3.1 to a a finite, regular, CW complex $\mathcal{K}$ followed by the construction of the respective Morse complex and computation of its homology by algebraic methods. Note that a straightforward, naive implementation of formula (2) may lead to heavy computations and consequently the gains from the approach may not show up. An algorithm for the boundary operator Δ used in our implementations is considerably more efficient than Forman's sum over paths formula. It will be presented elsewhere.

4   Numerical experiments

We have two independent implementations of CR algorithm and DMT algorithm. The first is by S. Harker and will be available soon from CHOMP [2]. The other is by H. Wagner, M. Juda, P.

	$T \times S^1$	$(S^1)^3$	$S^1 \times K$	$T \times T$
dim	5	6	6	6
size in millions	0.07	0.10	0.40	2.36
H_0	$\mathbb{Z}$	$\mathbb{Z}$	$\mathbb{Z}$	$\mathbb{Z}$
H_1	$\mathbb{Z}^3$	0	$\mathbb{Z}^2 + \mathbb{Z}_2$	$\mathbb{Z}^4$
H_2	$\mathbb{Z}^3$	0	$\mathbb{Z} + \mathbb{Z}_2$	$\mathbb{Z}^6$
H_3	$\mathbb{Z}$	$\mathbb{Z}$		$\mathbb{Z}^4$
H_4				$\mathbb{Z}$
Linbox::Smith	130	350	> 600	> 600
RedHom::Shave+**Linbox**::Smith	0.5	0.1	2.2	> 600
ChomP	1.3	1.7	10	56
RedHom::CR	0.03	0.04	0.26	2.5
ChomP::DMT	0.06	0.15	1.6	5.9
ChomP::CR+DMT	0.04	0.16	1.7	3
RedHom::CR+DMT	0.02	0.08	0.5	1.1

Table 2: The homology and the timings in seconds for a collection of cubical manifolds. The consecutive rows contain: the manifolds, the embedding dimension, the number of cells counted in millions, the homology groups and the timings for various implementations of homology algorithms.

	P0001	P0050	P0100
dim	3	3	3
size in millions	75.56	73.36	71.64
H_0	$\mathbb{Z}^7$	$\mathbb{Z}^2$	$\mathbb{Z}$
H_1	$\mathbb{Z}^{6554}$	$\mathbb{Z}^{2962}$	$\mathbb{Z}^{1057}$
H_2	$\mathbb{Z}^2$		
Linbox::Smith	> 600	> 600	> 600
RedHom::Shave+**Linbox**::Smith	> 600	> 600	> 600
ChomP	400	360	310
RedHom::CR	36	34	33
ChomP::DMT	110	110	100
ChomP::CR+DMT	45	43	42
RedHom::CR+DMT	26	25	24

Table 3: The homology and the timings in seconds for a collection of cubical sets arising from the numerical simulation of Cahn-Hillard partial differential equation. The consecutive rows contain: the names of sets, the embedding dimension, the number of cells counted in millions, the homology groups and the timings for various implementations of homology algorithms.

Dłotko and M. Mrozek and will be available soon as RedHom sublibrary [3] of CAPD [1]. Both implementations are written in C++.

We have compared the performance of these implementations, as well as the combination of CR followed by DMT, with our earlier best algorithms and with the classical approach utilizing Smith diagonalization available form LinBox [4].

We chose four collections of sets for our tests: a collection of five and six dimensional cubical manifolds, a collection of huge cubical sets arising from numerical simulations of Cahn-Hillard partial differential equation and a collection of random cubical sets in dimension four.

For comparison purposes we chose the following implementations of homology algorithms

Linbox::Smith - the classical homology algorithm based on the LinBox [4] implementation of Smith diagonalization

	d4s8f50	d4s12f50	d4s16f50	d4s20f50
dim	4	4	4	4
size in millions	0.07	0.34	1.04	2.48
H_0	$\mathbb{Z}^2$	$\mathbb{Z}^2$	$\mathbb{Z}^2$	$\mathbb{Z}^2$
H_1	$\mathbb{Z}^2$	$\mathbb{Z}^{17}$	$\mathbb{Z}^{30}$	$\mathbb{Z}^{51}$
H_2	$\mathbb{Z}^{174}$	$\mathbb{Z}^{1389}$	$\mathbb{Z}^{5510}$	$\mathbb{Z}^{15401}$
H_3	$\mathbb{Z}^2$	$\mathbb{Z}^{15}$	$\mathbb{Z}^{71}$	$\mathbb{Z}^{179}$
Linbox::Smith	120	> 600	> 600	> 600
RedHom::Shave+**Linbox**::Smith	4	> 600	> 600	> 600
ChomP	1	8.3	41	170
RedHom::CR	0.08	1.4	15	140
ChomP::DMT	0.05	0.38	1.8	5.3
ChomP::CR+DMT	0.03	0.16	0.56	1.4
RedHom::CR+DMT	0.03	0.16	0.58	2.9

Table 4: The homology and the timings in seconds for a collection of random cubical sets. The consecutive rows contain: the names of sets, the embedding dimension, the number of cells counted in millions, the homology groups and the timings for various implementations of homology algorithms.

RedHom::Shave+Linbox::Smith - as **Linbox::Smith** but preprocessed by the method of shaving in the spirit of [20, Section 7] in order to speed up the computations

ChomP - the main homology engine available from CHOMP [2] based on the implementation by P. Pilarczyk

RedHom::CR - the implementation by M. Mrozek of the CR homology algorithm [19] for Red-Hom sublibrary of CAPD [1] and CHOMP [2].

ChomP::DMT - the implementation by. S. Harker of DMT homology algorithm for CHOMP [2].

ChomP::CR+DMT - a combination of the CR homology algorithm followed by DMT homology algorithm implemented by S. Harker for CHOMP [2].

RedHom::CR+DMT - a combination of the CR homology algorithm followed by DMT homology algorithm implemented by H. Wagner, M. Juda, P. Dłotko and M. Mrozek for RedHom sublibrary [3] of CAPD [1].

The homology of the examples and the timings for the selected implementations are presented in Tables 2, 3 and 4. The tests were performed on a 3.0GHz Intel Xeon processor with 16GB of RAM running Linux. Note that in the case of Linbox based implementations we present only the timing of the Smith diagonalization of the matrices of boundary maps. In this case the actual time needed to compute the homology is longer, since the time needed to generate the matrices of boundary maps and (in the case of **RedHom::Shave+Linbox::Smith**) the preprocessing time should be added.

Tables 2, 3 and 4 indicate that DMT homology algorithm may be a valuable alternative to the existing methods, although a combination of CR homology algorithm followed by DMT homology algorithm may be even better. Most likely this is because DMT homology algorithm, unlike CR homology algorithm, requires storing all the coreduction pairs processed, because they are needed in the construction of the Morse complex. On the other hand, in the case of CR homology algorithm each spotted coreduction pair may be immediately removed.

The implementations are primarily aimed at cubical sets but they are written using generic methods so we tested them also on a collection of three simplicial complexes. The results are

	random set	Björner set	S^5
dim	4	2	5
size in millions	4.8	1.9	4.3
H_0	$\mathbb{Z}$	$\mathbb{Z}$	$\mathbb{Z}$
H_1	$\mathbb{Z}^{39}$	0	0
H_2	$\mathbb{Z}^{84}$	$\mathbb{Z}$	0
H_3			0
H_4			$\mathbb{Z}$
ChomP	830	310	2100
RedHom::CR+DMT	65	11	100

Table 5: The homology and the timings in seconds for a collection of simplicial complexes. The consecutive rows contain: the names of sets, the embedding dimension, the number of cells counted in millions, the homology groups and the timings for various implementations of homology algorithms.

available in Table 5. We also tested these examples with the classical Smith diagonalization method based on [4] but in each case we failed to get results either due to lack of memory of very long waiting times. However, these results are very preliminary and DMT homology algorithm must be tested on a larger collection of simplicial complexes to draw some conclusions.

Summing up, allthough we cannot claim that discrete Morse theory provides a universal solution for fast homology computations, we believe that DMT homology algorithm, and particularly the combination of CR and DMT homology algorithms may be very helpful in computing homology of many huge sets appearing in concrete problems.

References

[1] CAPD: Computer assisted proofs in dynamics, http://capd.ii.uj.edu.pl/.

[2] CHOMP: Computational homology project, http://chomp.rutgers.edu/.

[3] CAPD::RedHom: Reduction homology algorithms, http://redhom.ii.uj.edu.pl/.

[4] Project linbox: Exact computational linear algebra, http://www.linalg.org/.

[5] M. Allili, K. Mischaikow, and A. Tannenbaum. Cubical homology and the topological classification of 2d and 3d imagery. *IEEE International Conference on Image Processing*, 2:173–176, 2001.

[6] Dirk Blömker, Stanislaus Maier-Paape, and Thomas Wanner. Phase separation in stochastic Cahn-Hilliard models. In Alain Miranville, editor, *Mathematical Methods and Models in Phase Transitions*, pages 1–41. Nova Science Publishers, New York, 2005.

[7] Cecil Jose A. Delfinado and Herbert Edelsbrunner. An incremental algorithm for betti numbers of simplicial complexes. In *SCG '93: Proceedings of the ninth annual symposium on Computational geometry*, pages 232–239, New York, NY, USA, 1993. ACM.

[8] Jean-Guillaume Dumas, Frank Heckenbach, David Saunders, and Volkmar Welker. Computing simplicial homology based on efficient Smith normal form algorithms. In *Algebra, geometry, and software systems*, pages 177–206. Springer, Berlin, 2003.

[9] Herbert Edelsbrunner, David Letscher, and Afra Zomorodian. Topological persistence and simplification. *Discrete & Computational Geometry*, 28(4):511–533, 2002.

[10] Robin Forman. Morse theory for cell complexes. *Advances in Mathematics*, 134:90–145, 1998.

[11] J. Friedman. Computing Betti numbers via combinatorial Laplacians. *Algorithmica*, 21(4):331–346, 1998.

[12] P. Frosini and C. Landi. Size theory as a topological tool for computer vision. *Pattern Recognition And Image Analysis*, 9(4):596–603, 1999.

[13] Marcio Gameiro, Konstantin Mischaikow, and Thomas Wanner. Evolution of pattern complexity in the Cahn-Hilliard theory of phase separation. *Acta Materialia*, 53(3):693–704, 2005.

[14] Shaun Harker, Konstantin Mischaikow, Marian Mrozek, and Vidit Nanda. Computing homology via Morse coreductions. in preparation.

[15] T. Kaczyński, M. Mrozek, and M. Ślusarek. Homology computation by reduction of chain complexes. *Computers & Mathematics with Applications*, 35(4):59–70, 1998.

[16] Tomasz Kaczynski, Konstantin Mischaikow, and Marian Mrozek. Computing homology. *Homology, Homotopy and Applications*, 5(2):233–256, 2003.

[17] Tomasz Kaczynski, Konstantin Mischaikow, and Marian Mrozek. *Computational Homology*, volume 157 of *Applied Mathematical Sciences*. Springer-Verlag, New York, 2004.

[18] W.S. Massey. *A Basic Course in Algebraic Topology Applied Mathematical Sciences, **157***. Springer-Verlag, New York – Berlin – Heidelberg, 1991.

[19] Marian Mrozek and Bogdan Batko. Coreduction homology algorithm. *Discrete & Computational Geometry*, 41(1):96–118, 2009.

[20] Marian Mrozek, Paweł Pilarczyk, and Natalia Żelazna. Homology algorithm based on acyclic subspace. *Computers & Mathematics with Applications*, 55(11):2395–2412, 2008.

[21] Marian Mrozek and Thomas Wanner. Coreduction homology algorithm for inclusions and persistent homology. *Computers and Mathematics with Applications*, accepted.

[22] Samuel Peltier, Adrian Ion, Yll Haxhimusa, Walter Kropatsch, and Guillaume Damiand. Computing homology group generators of images using irregular graph pyramids. *Lecture Notes in Computer Science*, 4538(3):283–294, 2007.

[23] Francis Sergeraert. W-reductions. personal communication.

[24] A. Szymczak, A. Stillman, A. Tannenbaum, and K. Mischaikow. Coronary vessel trees from 3d imagery: A topological approach. *Med Image Anal.*, 10(4):548–559, 2006.

[25] A. Verri and C. Uras. Metric-topological approach to shape representation and recognition. *Image Vision Comput.*, 14:189–207, 1996.

[26] Thomas Wanner, Edwin R. Fuller Jr., and David M. Saylor. Homological characterization of microstructure response fields in polycrystals. *Acta Materialia*, 58(1):102–110, 2010.

[27] M. Żelawski. Pattern recognition based on homology theory. *Machine Graphics & Vision*, 14(3):309–324, 2005.

[28] D. Ziou and M. Allili. Generating cubical complexes from image data and computation of the euler number. *Pattern Recognition*, 35(12):2833–2839, 2002.