Computational Homology Applied to Discrete Objects

Aldo Gonzalez-Lorenzo[1,2]

[1]Aix-Marseille Université, CNRS, LSIS UMR 7296 (France)
[2]Universidad de Sevilla, IMUS (Spain)

November 24, 2016

Structure

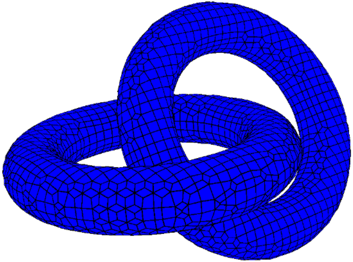
1 Introduction and Preliminaries

2 The Homological Discrete Vector Field

3 Fast Computation of Betti Numbers on 3D Cubical Complexes

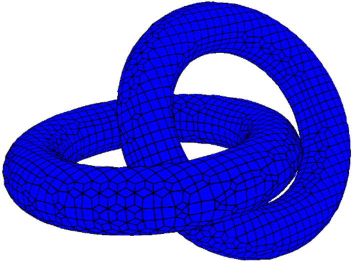
4 Measuring Holes

5 Conclusion



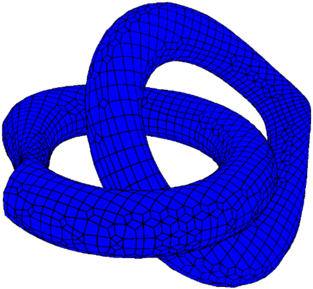
Geometry:

- Volume
- Diameter
- Curvature
- Holes!



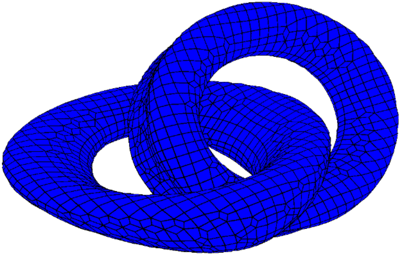
Topology:

- Volume
- Diameter
- Curvature
- Holes!



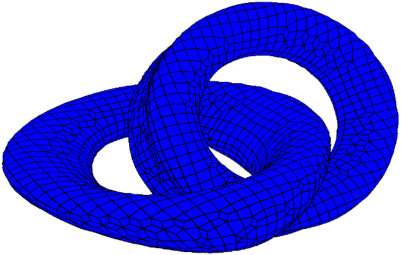
Topology:

- Volume
- Diameter
- Curvature
- Holes!



Topology:

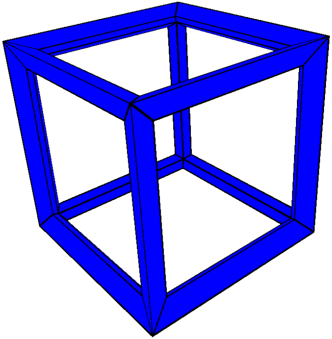
- Volume
- Diameter
- Curvature
- Holes!



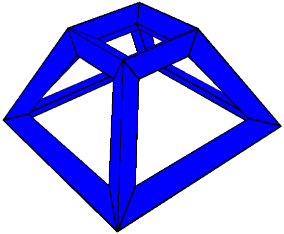
Topology:

- Volume
- Diameter
- Curvature
- Holes!

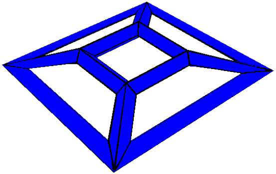
Number of holes: 6? 3?



Number of holes: 6? 3?



Number of holes: 5



Sections

1 Introduction and Preliminaries
- Complexes
- Homology
- Reduction

2 The Homological Discrete Vector Field

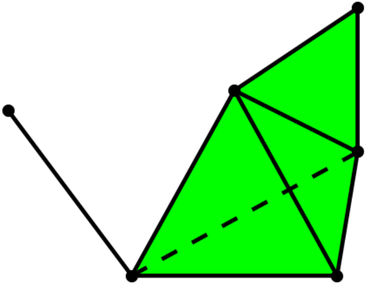
3 Fast Computation of Betti Numbers on 3D Cubical Complexes

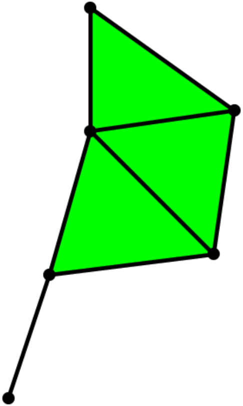
4 Measuring Holes

5 Conclusion

Simplicial complex

Union[1] of points, edges, triangles, tetrahedra, ... (simplices)

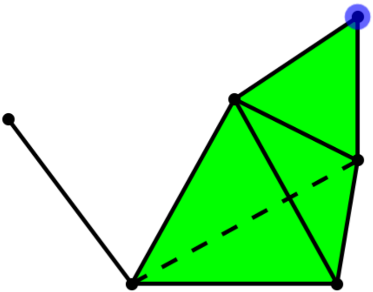


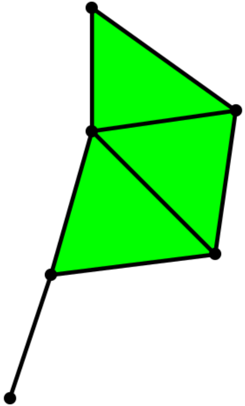


[1] with some conditions (cf. Definition 2.14)

Simplicial complex

Union[1] of **points**, edges, triangles, tetrahedra, ... (simplices)

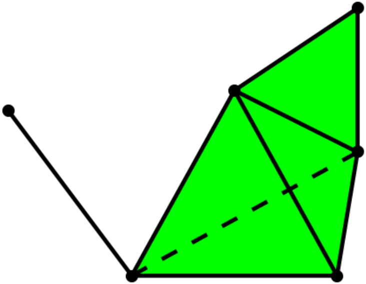


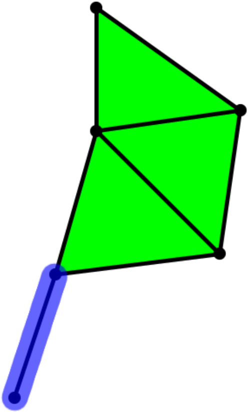


[1] with some conditions (cf. Definition 2.14)

Simplicial complex

Union[1] of points, **edges**, triangles, tetrahedra, ... (simplices)

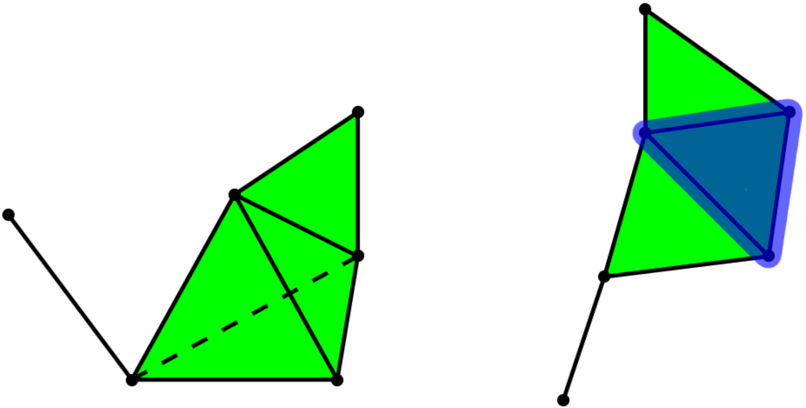




[1]with some conditions (cf. Definition 2.14)

Simplicial complex

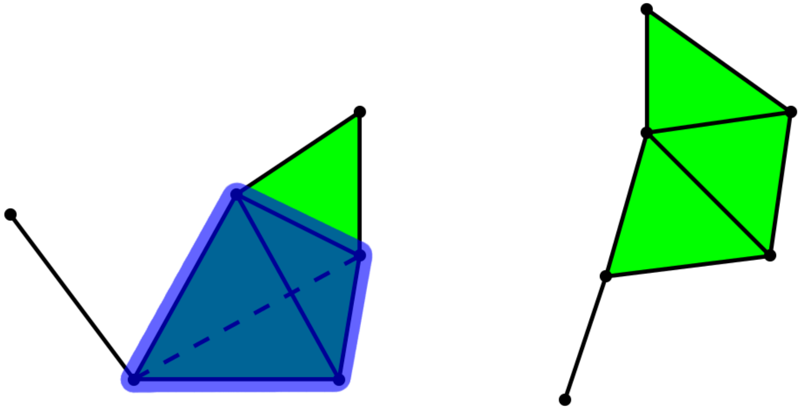
Union[1] of points, edges, **triangles**, tetrahedra, ... (simplices)



[1] with some conditions (cf. Definition 2.14)

Simplicial complex

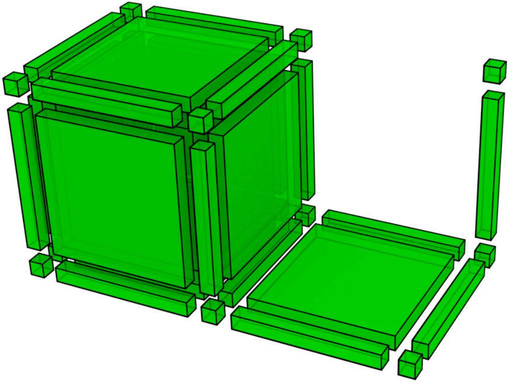
Union[1] of points, edges, triangles, **tetrahedra**, ... (simplices)



[1] with some conditions (cf. Definition 2.14)

Cubical complex

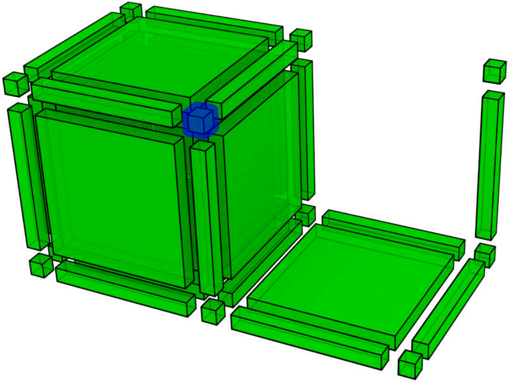
Union[2] of points, edges, squares, cubes, ... (cubes)



[2]with some conditions (cf. Definition 2.17)

Cubical complex

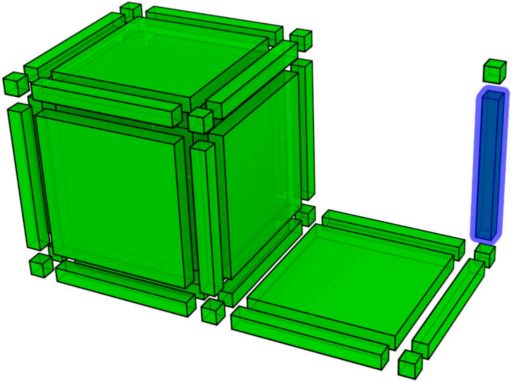
Union[2] of **points**, edges, squares, cubes, ... (cubes)



[2] with some conditions (cf. Definition 2.17)

Cubical complex

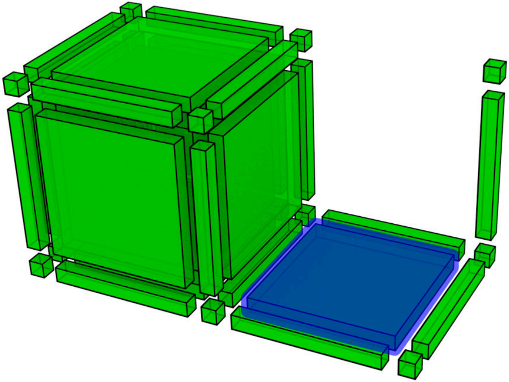
Union[2] of points, **edges**, squares, cubes, ... (cubes)



[2] with some conditions (cf. Definition 2.17)

Cubical complex

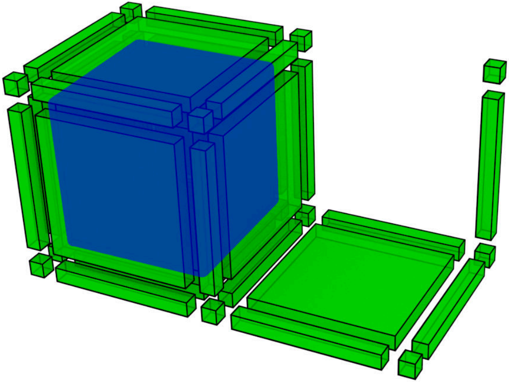
Union[2] of points, edges, **squares**, cubes, ... (cubes)



[2]with some conditions (cf. Definition 2.17)

Cubical complex

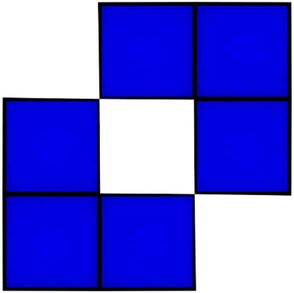
Union[2] of points, edges, squares, **cubes**, ... (cubes)

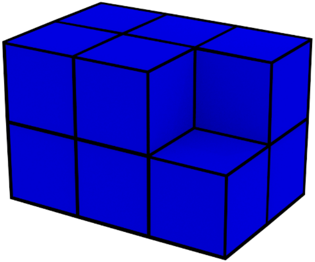


[2]with some conditions (cf. Definition 2.17)

Discrete object

A nD discrete object is a subset of $\mathbb{Z}^n$

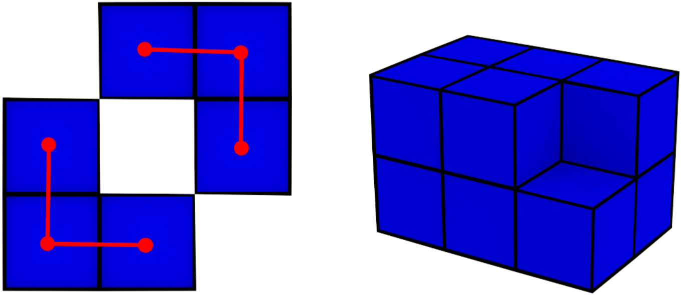




We usually choose a connectivity relation such as the $2n$ or the $(3^n - 1)$-connectivity.

Discrete object

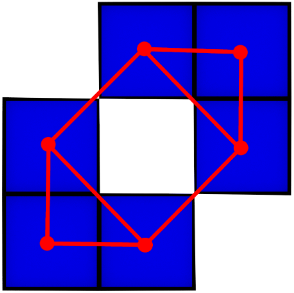
A nD discrete object is a subset of $\mathbb{Z}^n$

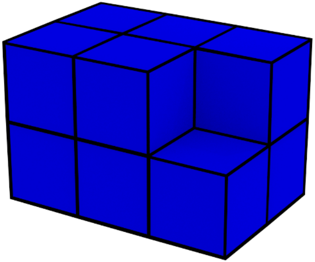


We usually choose a connectivity relation such as the $2n$ or the $(3^n - 1)$-connectivity.

Discrete object

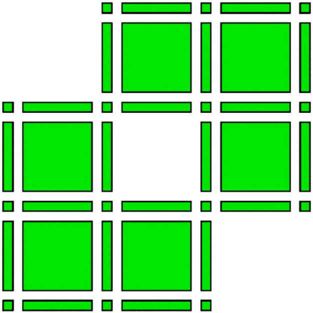
A nD discrete object is a subset of $\mathbb{Z}^n$

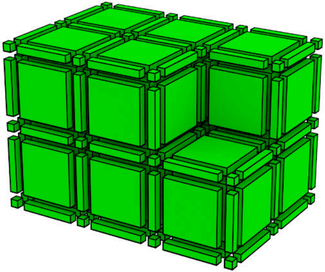




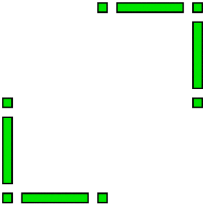
We usually choose a connectivity relation such as the $2n$ or the $(3^n - 1)$-connectivity.

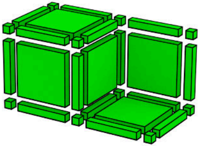
We can transform a discrete object into a cubical complex in two ways, one for each connectivity relation.





We can transform a discrete object into a cubical complex in two ways, one for each connectivity relation.





Sections

1 Introduction and Preliminaries
- Complexes
- Homology
- Reduction

2 The Homological Discrete Vector Field

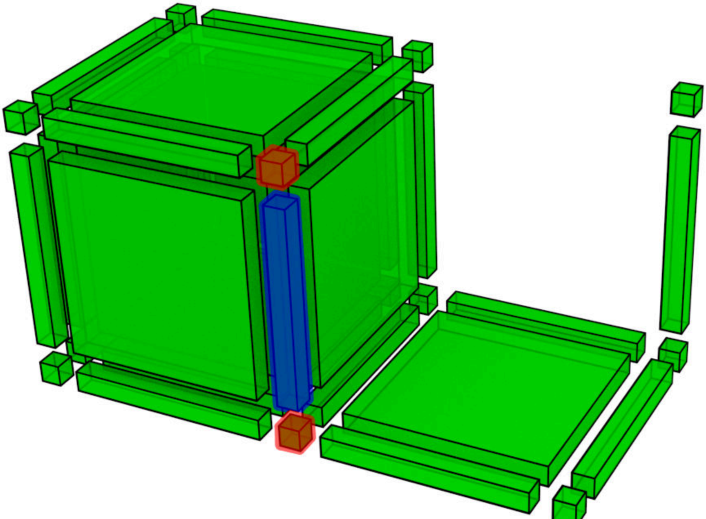
3 Fast Computation of Betti Numbers on 3D Cubical Complexes

4 Measuring Holes

5 Conclusion

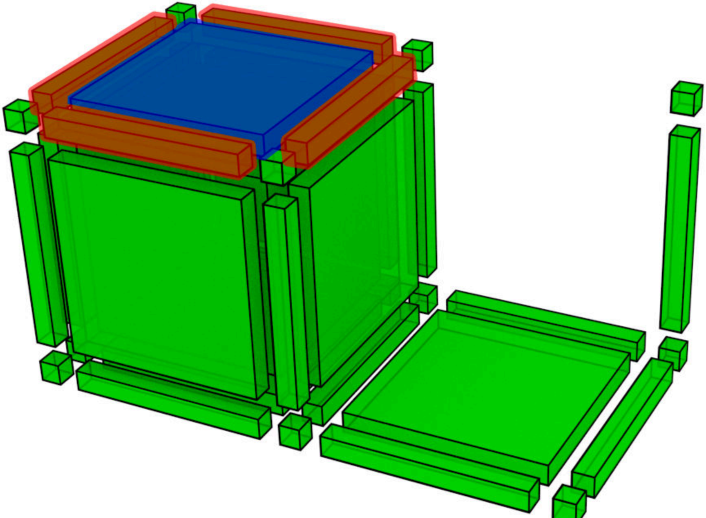
Blue: 1-cube

Red: its boundary (faces)



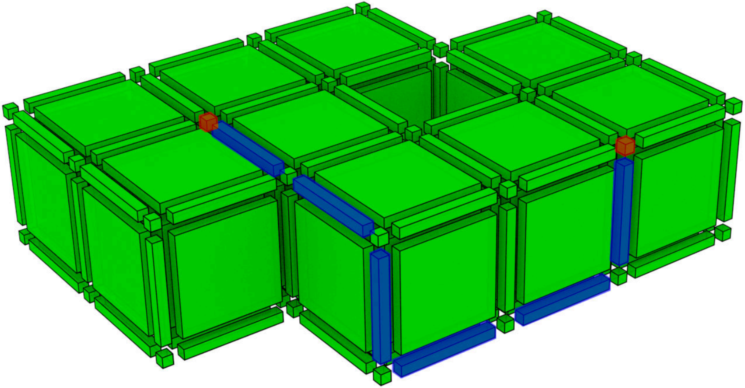
Blue: 2-cube

Red: its boundary (faces)



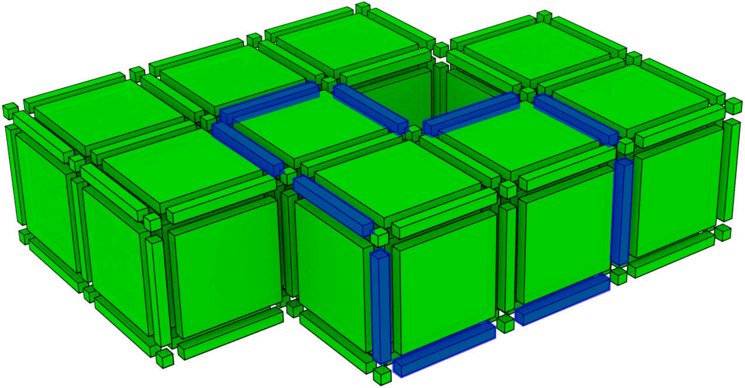
Blue: 1-chain

Red: its boundary



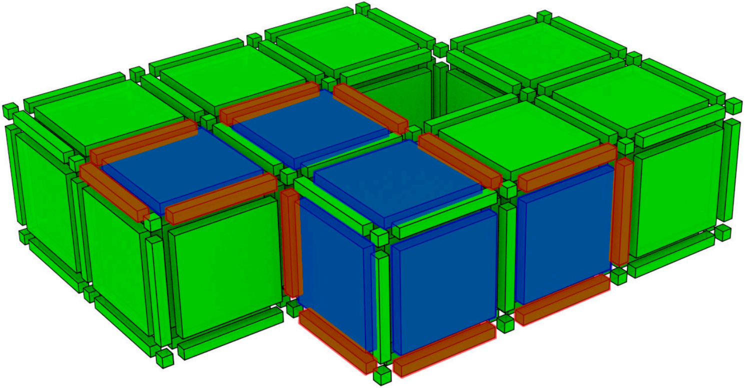
Blue: 1-chain (1-cycle)

Red: its boundary ($= \emptyset$)



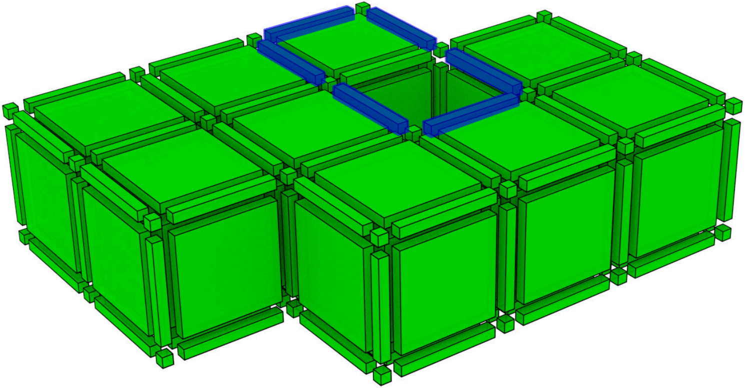
Blue: 2-chain

Red: its boundary (1-cycle)



Blue: 1-chain (1-cycle, but not boundary)

Red: its boundary ($= \emptyset$)



- K cubical complex, $\mathfrak{R}$ ring (e.g., $\mathbb{Z}$, $\mathbb{Z}_2$)
- Chain complex of K

$$\cdots C_3 \xrightarrow{d_3} C_2 \xrightarrow{d_2} C_1 \xrightarrow{d_1} C_0 \xrightarrow{d_0} 0$$

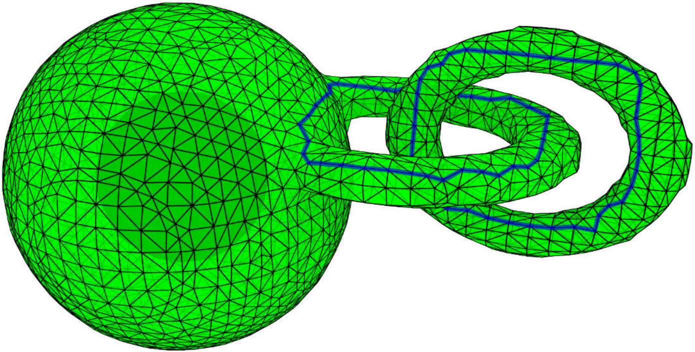
where $d_q d_{q+1} = 0 \Rightarrow \operatorname{im}(d_{q+1}) \subset \ker(d_q)$

- q-dimensional homology group
$H_q(K) := \ker(d_q)/\operatorname{im}(d_{q+1}) = \mathbb{Z}^{\beta_q} \oplus \mathbb{T}$
- q-dimensional Betti number: β_q

- $\beta_0 = \#$ connected components (0-holes)
- $\beta_1 = \#$ tunnels or handles (1-holes)
- $\beta_2 = \#$ cavities (2-holes)

Betti numbers are

- Topological invariants $\rightarrow$ classification
- Shape descriptors $\rightarrow$ understanding



$\beta_0 = 2,\ \beta_1 = 2,\ \beta_2 = 1,\ \beta_3 = 0,\ \ldots$

Sections

1 Introduction and Preliminaries
- Complexes
- Homology
- Reduction

2 The Homological Discrete Vector Field

3 Fast Computation of Betti Numbers on 3D Cubical Complexes

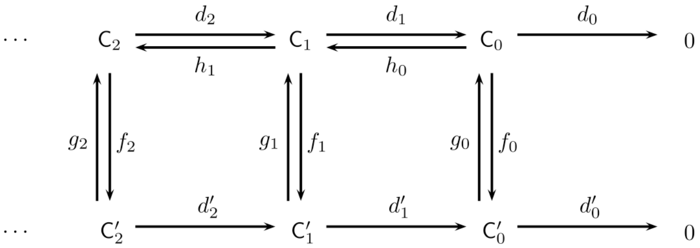
4 Measuring Holes

5 Conclusion

Effective homology theory [Sergeraert, 1992]

Reduction

Triplet $\rho = (h, f, g)$ of graded homomorphisms[3] between two chain complexes (C, d) and (C', d')



Both chain complexes have isomorphic homology groups

[3] with some conditions (cf. Definition 2.18)

A reduction is perfect if $d' = 0$. Hence

- $C' \cong H(C)$
- $g(C') =$ homology generators
- $f^*(C') =$ cohomology generators
- $d(x) = 0 \Rightarrow d(y) = x$ for $y = h(x)$

Sections

1 Introduction and Preliminaries

2 The Homological Discrete Vector Field
- Introduction
- Definitions and theorems
- Computing a HDVF
- Deforming a HDVF
- Relation with other methods in computational homology
- Conclusion

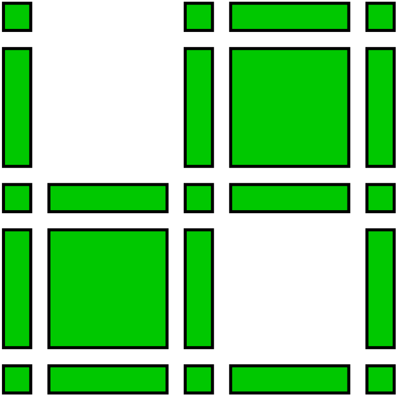
3 Fast Computation of Betti Numbers on 3D Cubical Complexes

4 Measuring Holes

5 Conclusion

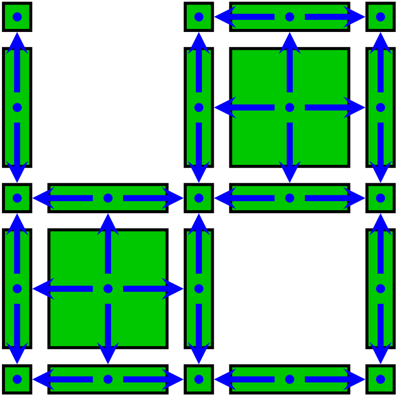
Discrete Morse theory [Forman, 1998]

- (CW-) complex
- Connectivity graph
- Matching $\mathcal{V}$
- Morse graph (no cycles)
- $\mathcal{V}$ is a discrete gradient vector field (DGVF)



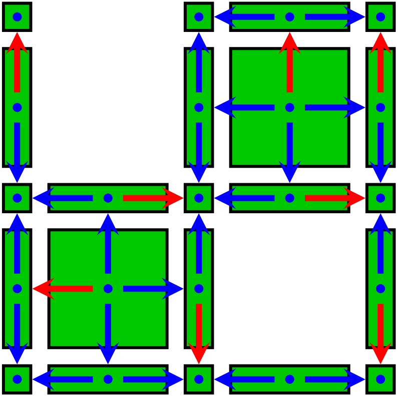
Discrete Morse theory [Forman, 1998]

- (CW-) complex
- Connectivity graph
- Matching $\mathcal{V}$
- Morse graph (no cycles)
- $\mathcal{V}$ is a discrete gradient vector field (DGVF)



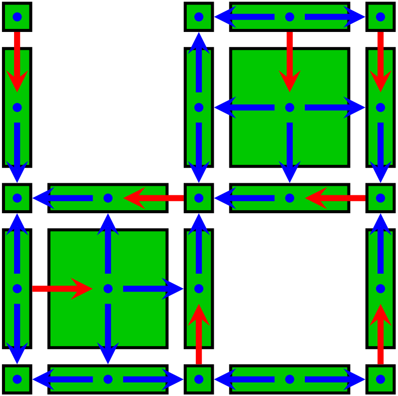
Discrete Morse theory [Forman, 1998]

- (CW-) complex
- Connectivity graph
- Matching $\mathcal{V}$
- Morse graph (no cycles)
- $\mathcal{V}$ is a discrete gradient vector field (DGVF)



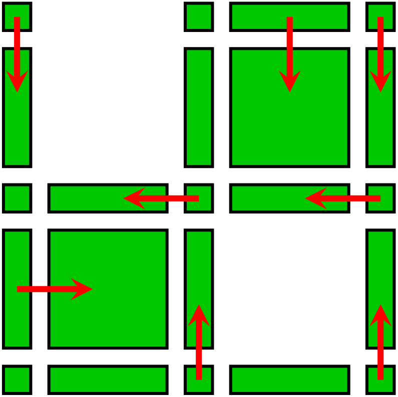
Discrete Morse theory [Forman, 1998]

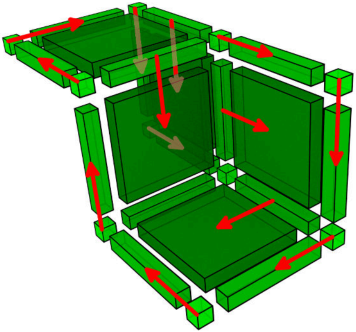
- (CW-) complex
- Connectivity graph
- Matching $\mathcal{V}$
- Morse graph (no cycles)
- $\mathcal{V}$ is a discrete gradient vector field (DGVF)



Discrete Morse theory [Forman, 1998]

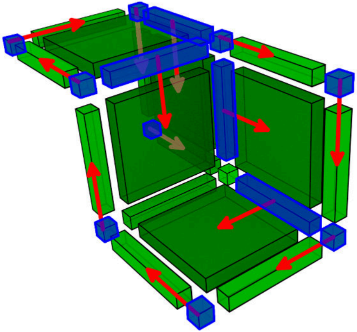
- (CW-) complex
- Connectivity graph
- Matching $\mathcal{V}$
- Morse graph (no cycles)
- $\mathcal{V}$ is a discrete gradient vector field (DGVF)





A DGVF

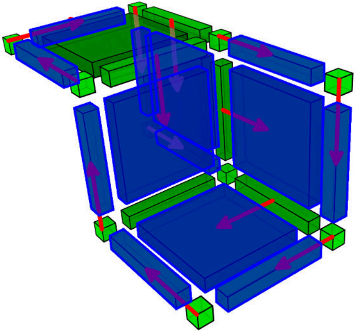
- Acyclic matching
- The arrows can be deduced from P and S
- It induces a reduction
- $|C_q| \geq \beta_q$



P: primary cells

A DGVF

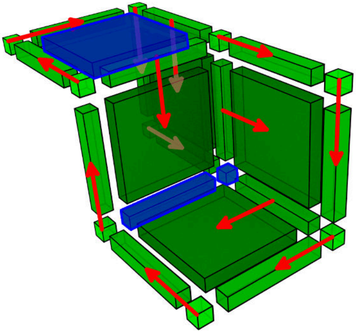
- Acyclic matching
- The arrows can be deduced from P and S
- It induces a reduction
- $|C_q| \geq \beta_q$



S: secondary cells

A DGVF

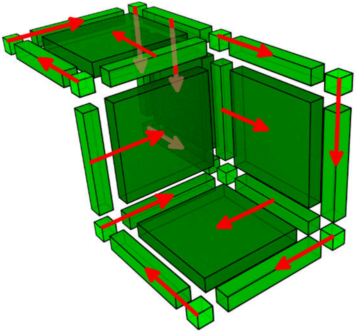
- Acyclic matching
- The arrows can be deduced from P and S
- It induces a reduction
- $|C_q| \geq \beta_q$



C: critical cells

A DGVF

- Acyclic matching
- The arrows can be deduced from P and S
- It induces a reduction
- $|C_q| \geq \beta_q$



A perfect DGVF

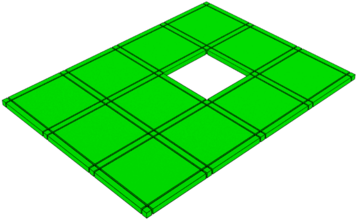
- Acyclic matching
- The arrows can be deduced from P and S
- It induces a perfect reduction
- $|C_q| = \beta_q$

So

- Algebra → graph theory
- Homology computation → optimization problem

But

- Finding optimal DGVF is NP
- No possible perfect DGVF always

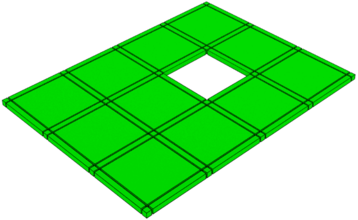


So

- Algebra → graph theory
- Homology computation → optimization problem

But

- Finding optimal DGVF is NP
- No possible perfect DGVF always

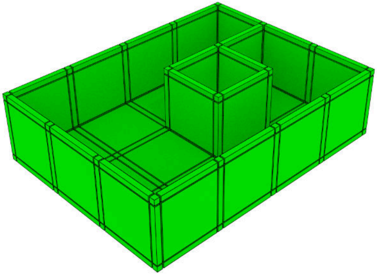


So

- Algebra → graph theory
- Homology computation → optimization problem

But

- Finding optimal DGVF is NP
- No possible perfect DGVF always

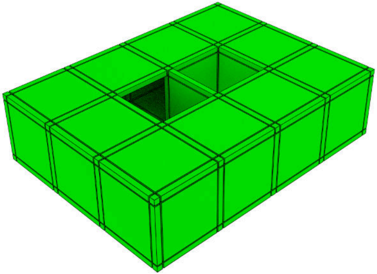


So

- Algebra → graph theory
- Homology computation → optimization problem

But

- Finding optimal DGVF is NP
- No possible perfect DGVF always

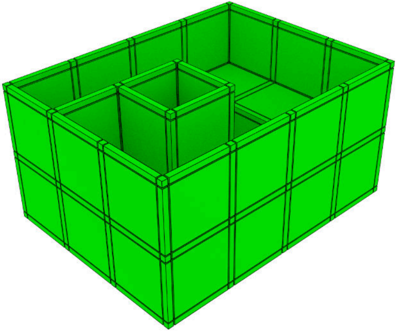


So

- Algebra → graph theory
- Homology computation → optimization problem

But

- Finding optimal DGVF is NP
- No possible perfect DGVF always

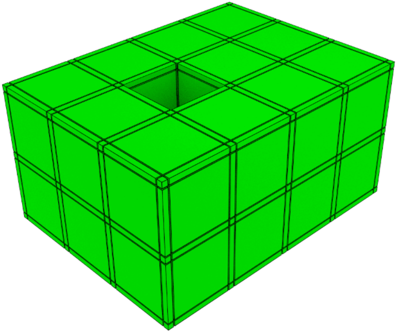


So

- Algebra → graph theory
- Homology computation → optimization problem

But

- Finding optimal DGVF is NP
- No possible perfect DGVF always



Sections

1 Introduction and Preliminaries

2 The Homological Discrete Vector Field
- Introduction
- Definitions and theorems
- Computing a HDVF
- Deforming a HDVF
- Relation with other methods in computational homology
- Conclusion

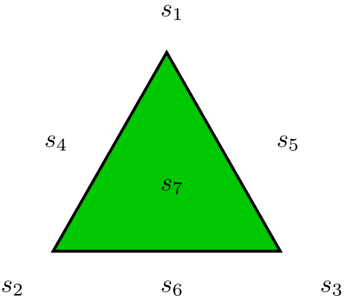
3 Fast Computation of Betti Numbers on 3D Cubical Complexes

4 Measuring Holes

5 Conclusion

Boundary matrix

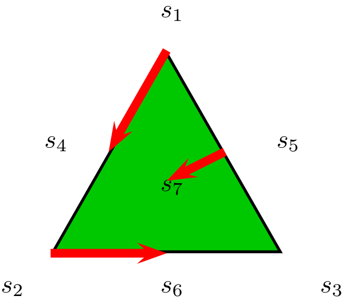
Matrix of the (linear) boundary operator d



$$d = \begin{array}{c} \\ s_1 \\ s_2 \\ s_3 \\ s_4 \\ s_5 \\ s_6 \\ s_7 \end{array} \begin{array}{c} \begin{array}{ccccccc} s_1 & s_2 & s_3 & s_4 & s_5 & s_6 & s_7 \end{array} \\ \begin{pmatrix} 0 & 0 & 0 & 1 & 1 & 0 & 0 \\ 0 & 0 & 0 & 1 & 0 & 1 & 0 \\ 0 & 0 & 0 & 0 & 1 & 1 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 1 \\ 0 & 0 & 0 & 0 & 0 & 0 & 1 \\ 0 & 0 & 0 & 0 & 0 & 0 & 1 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 \end{pmatrix} \end{array}$$

Boundary matrix

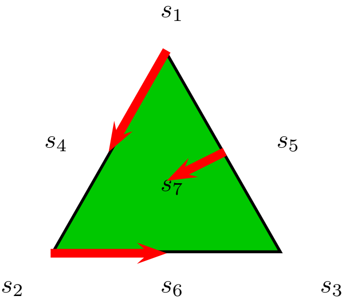
Matrix of the (linear) boundary operator *d*



$$d = \begin{array}{c|ccccccc} & s_1 & s_2 & s_3 & s_4 & s_5 & s_6 & s_7 \\ \hline s_1 & 0 & 0 & 0 & 1 & 1 & 0 & 0 \\ s_2 & 0 & 0 & 0 & 1 & 0 & 1 & 0 \\ s_3 & 0 & 0 & 0 & 0 & 1 & 1 & 0 \\ s_4 & 0 & 0 & 0 & 0 & 0 & 0 & 1 \\ s_5 & 0 & 0 & 0 & 0 & 0 & 0 & 1 \\ s_6 & 0 & 0 & 0 & 0 & 0 & 0 & 1 \\ s_7 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \end{array}$$

Boundary matrix

Matrix of the (linear) boundary operator *d*



$$d(S)_{|P} = \begin{array}{c} \\ s_1 \\ s_2 \\ s_5 \end{array} \begin{array}{c} \begin{array}{ccc} s_4 & s_6 & s_7 \end{array} \\ \begin{pmatrix} 1 & 0 & 0 \\ 1 & 1 & 0 \\ 0 & 0 & 1 \end{pmatrix} \end{array}$$

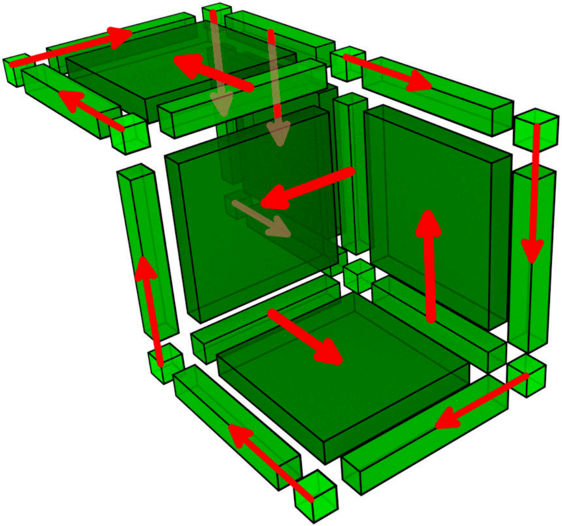
"Forget the cycles, focus on the reduction"

HDVF (Definition 3.1)

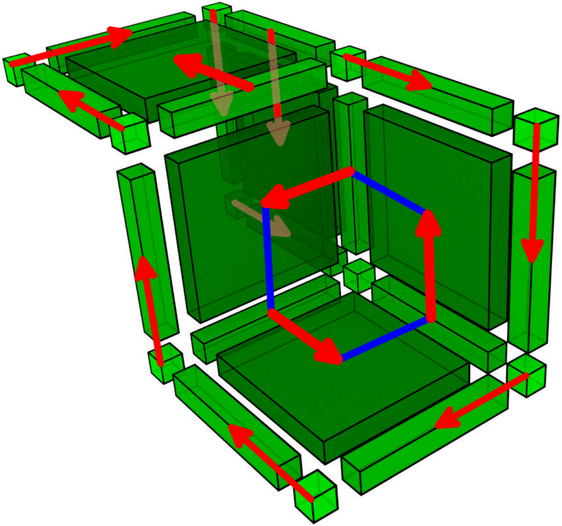
A homological discrete vector field (HDVF) $X = (P, S)$ on a CW complex K is a partition $K = P \sqcup S \sqcup C$ such that $d(S)_{|P}$ is an invertible matrix (in $\mathfrak{R}$)

We can always represent a HDVF as a discrete vector field (cf. Proposition 3.8)

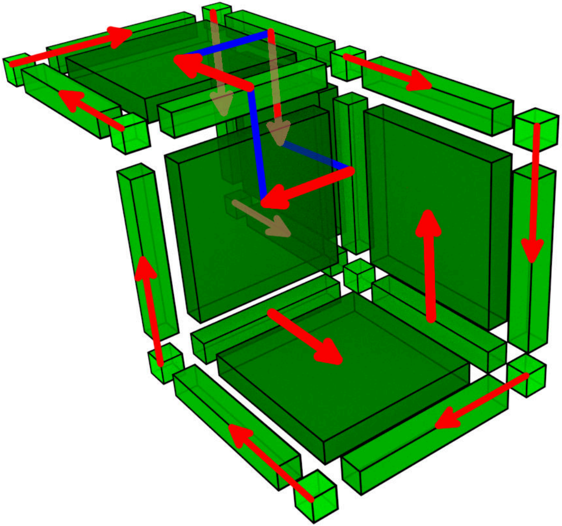
Example: a HDVF with two cycles in the Morse graph



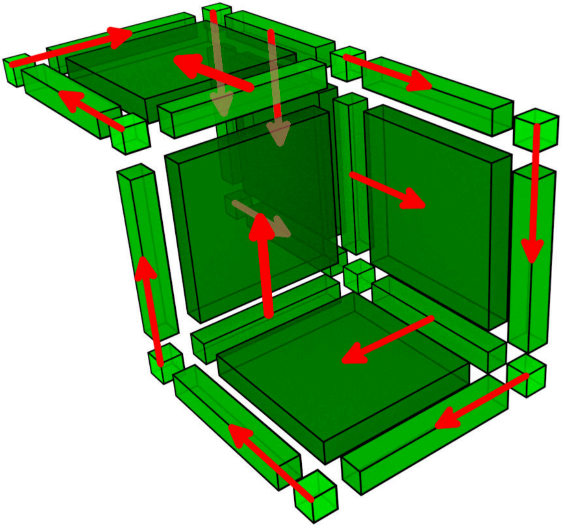
Example: a HDVF with two cycles in the Morse graph



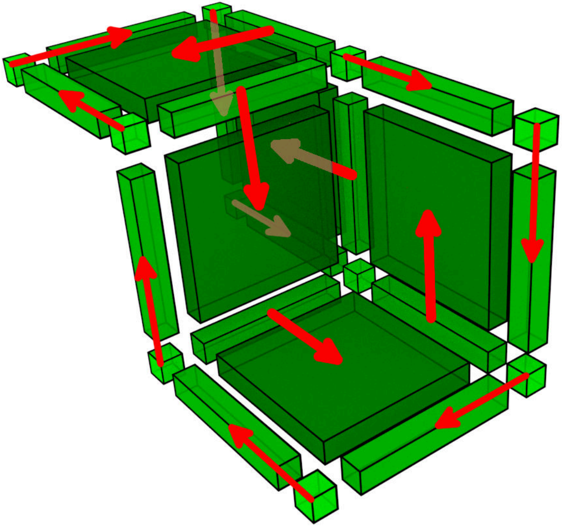
Example: a HDVF with two cycles in the Morse graph



Example: a HDVF with two cycles in the Morse graph



Example: a HDVF with two cycles in the Morse graph



Theorem 3.9

Let K be a CW complex endowed with a HDVF X. Then X induces the reduction

$$(h, f, g) : (C, d) \Rightarrow (\mathfrak{R}[C], d')$$

where the operators h, f, g and the reduced boundary d' are given by

$$h = \begin{array}{c|ccc} & P & S & C \\ \hline P & 0 & 0 & 0 \\ S & H & 0 & 0 \\ C & 0 & 0 & 0 \end{array} \quad f = \begin{array}{c|ccc} & P & S & C \\ \hline C & F & 0 & I \end{array} \quad g = \begin{array}{c|c} & C \\ \hline P & 0 \\ S & G \\ C & I \end{array} \quad d' = \begin{array}{c|c} & C \\ \hline C & D \end{array}$$

$$H = (d(S)_{|P})^{-1}$$
$$F = -d(S)_{|C} \cdot H$$
$$G = -H \cdot d(C)_{|P}$$
$$D = d(C)_{|C} + F \cdot d(C)_{|P}$$

Sections

1 Introduction and Preliminaries

2 The Homological Discrete Vector Field
- Introduction
- Definitions and theorems
- Computing a HDVF
- Deforming a HDVF
- Relation with other methods in computational homology
- Conclusion

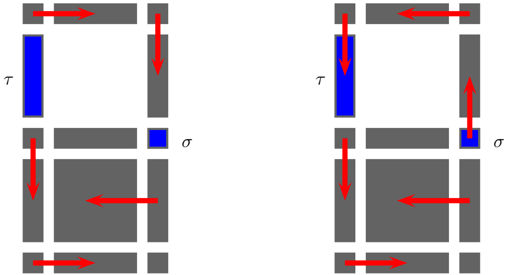
3 Fast Computation of Betti Numbers on 3D Cubical Complexes

4 Measuring Holes

5 Conclusion

Proposition 3.12

K CW complex, $X = (P, S)$ HDVF, σ, τ critical cells. If $\langle d'(\tau), \sigma \rangle$ is a unit then $X' = (P \cup \{\sigma\}, S \cup \{\tau\})$ is a HDVF.



Algorithm 1: Compute a HDVF

Input: A CW complex K

Output: A HDVF X

```
repeat
    Find two critical cells σ, τ such that ⟨d′(τ), σ⟩ is a unit;
    Add (σ, τ) to X;
    Update the reduced boundary matrix D;
until idempotency;
```

Theorem 3.15

Algorithm 1 can be computed within $\mathcal{O}(n^3)$ operations.

Algorithm 2: Compute a HDVF

Input: A CW complex K

Output: A HDVF X

```
repeat
    Find two critical cells σ, τ such that ⟨d'(τ), σ⟩ is a unit;
    Add (σ, τ) to X;
    Update the reduced boundary matrix D;
until idempotency;
```

Theorem 3.15

Algorithm 1 can be computed within $\mathcal{O}(n^3)$ operations.

Sections

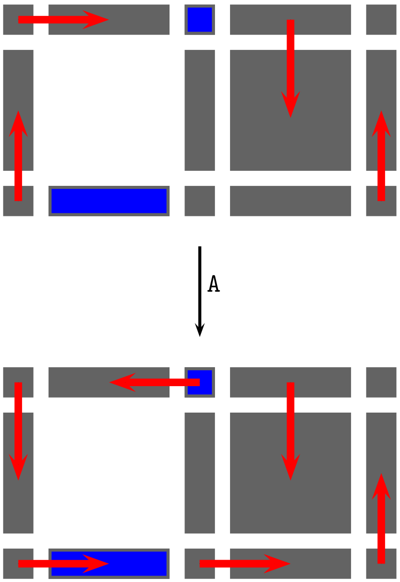
1 Introduction and Preliminaries

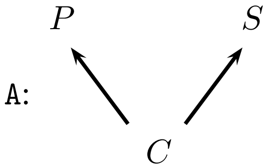
2 The Homological Discrete Vector Field
- Introduction
- Definitions and theorems
- Computing a HDVF
- Deforming a HDVF
- Relation with other methods in computational homology
- Conclusion

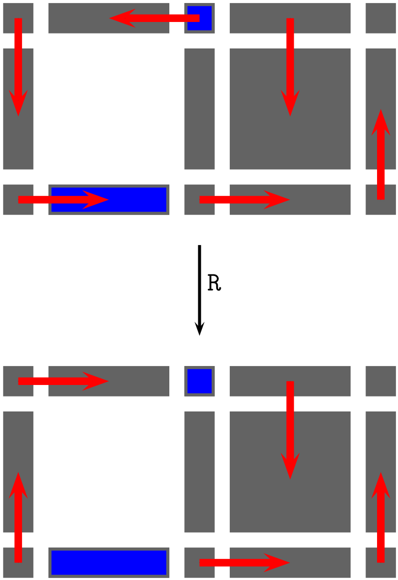
3 Fast Computation of Betti Numbers on 3D Cubical Complexes

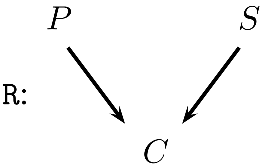
4 Measuring Holes

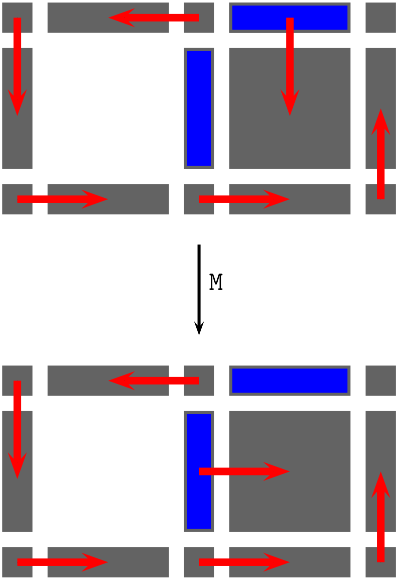
5 Conclusion

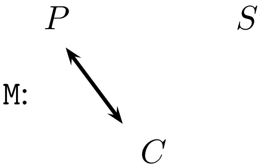

A

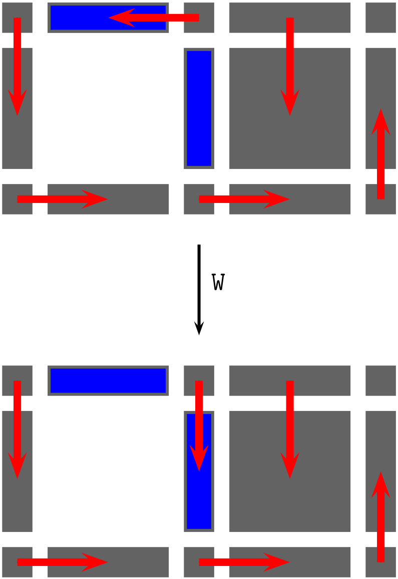

A:
P
S
C

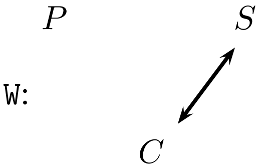

R

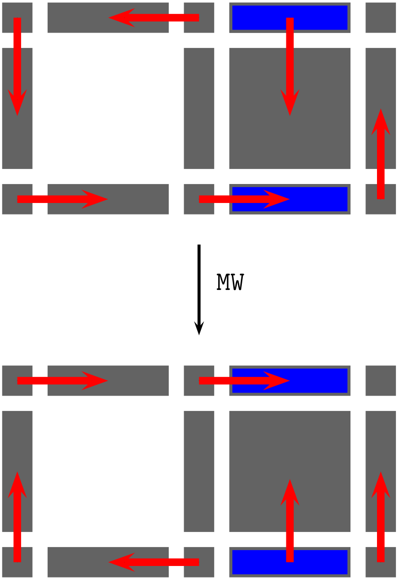

P
S
R:
C


M


P
S
M:
C


W


W:
P
S
C



$$P \longleftrightarrow S$$

MW:

$$C$$

Proposition 3.19

Let K be a CW complex endowed with a HDVF X. Let $\sigma \in P$, $\tau \in S$ and $\gamma, \gamma' \in C$. Thus,

1. $\mathtt{A}(X, \gamma, \gamma')$ is a HDVF if $\langle d'(\gamma'), \gamma \rangle$ is a unit
2. $\mathtt{R}(X, \sigma, \tau)$ is a HDVF if $\langle h(\sigma), \tau \rangle$ is a unit
3. $\mathtt{M}(X, \sigma, \gamma)$ is a HDVF if $\langle f(\sigma), \gamma \rangle$ is a unit
4. $\mathtt{W}(X, \tau, \gamma)$ is a HDVF if $\langle g(\gamma), \tau \rangle$ is a unit
5. $\mathtt{MW}(X, \sigma, \tau)$ is a HDVF if $\langle dh(\sigma), \tau \rangle$ and $\langle hd(\tau), \sigma \rangle$ are units

Sections

1 Introduction and Preliminaries

2 The Homological Discrete Vector Field
- Introduction
- Definitions and theorems
- Computing a HDVF
- Deforming a HDVF
- Relation with other methods in computational homology
- Conclusion

3 Fast Computation of Betti Numbers on 3D Cubical Complexes

4 Measuring Holes

5 Conclusion

Proposition 3.21

Every DGVF is a HDVF.

Proposition 3.22

Every iterated DGVF[4] is a HDVF.

Proposition 3.23

Let K be a CW complex. Then,

1. Algorithm 1 performs a partial diagonalization of the boundary matrices of K;
2. Algorithm 1 computes a perfect HDVF whenever $\mathfrak{R}$ is a field.

Thus, we can compute persistent homology with the HDVF

[4] [Dlotko and Wagner, 2012]

Proposition 3.21

Every DGVF is a HDVF.

Proposition 3.22

Every iterated DGVF[4] is a HDVF.

Proposition 3.23

Let K be a CW complex. Then,

1. Algorithm 1 performs a partial diagonalization of the boundary matrices of K;
2. Algorithm 1 computes a perfect HDVF whenever $\mathfrak{R}$ is a field.

Thus, we can compute persistent homology with the HDVF

[4] [Dlotko and Wagner, 2012]

Proposition 3.21

Every DGVF is a HDVF.

Proposition 3.22

Every iterated DGVF[4] is a HDVF.

Proposition 3.23

Let K be a CW complex. Then,

1. Algorithm 1 performs a partial diagonalization of the boundary matrices of K;
2. Algorithm 1 computes a perfect HDVF whenever $\mathfrak{R}$ is a field.

Thus, we can compute persistent homology with the HDVF

[4] [Dlotko and Wagner, 2012]

Sections

1 Introduction and Preliminaries

2 The Homological Discrete Vector Field
- Introduction
- Definitions and theorems
- Computing a HDVF
- Deforming a HDVF
- Relation with other methods in computational homology
- Conclusion

3 Fast Computation of Betti Numbers on 3D Cubical Complexes

4 Measuring Holes

5 Conclusion

Conclusion:

- HDVF: combinatorial structure for computing homology
- Visually representable
- More powerful than DGVF
- Cubical complexity

Conclusion:

- HDVF: combinatorial structure for computing homology
- Visually representable
- More powerful than DGVF
- Cubical complexity

Conclusion:

- HDVF: combinatorial structure for computing homology
- Visually representable
- More powerful than DGVF
- Cubical complexity

Conclusion:

- HDVF: combinatorial structure for computing homology
- Visually representable
- More powerful than DGVF
- Cubical complexity

Sections

1 Introduction and Preliminaries

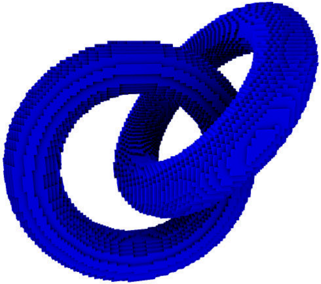
2 The Homological Discrete Vector Field

3 Fast Computation of Betti Numbers on 3D Cubical Complexes
- Introduction
- Algorithm
- Results
- Conclusion

4 Measuring Holes

5 Conclusion

We want to compute the Betti numbers of binary volumes



It seems that:

- $\beta_0 = \#$ connected components
- $\beta_2 = \#$ bounded connected components of the complement
- β_1 ?

[Delfinado and Edelsbrunner, 1995], [Dey and Guha, 1998]: 3D simplicial complexes

Sections

1 Introduction and Preliminaries

2 The Homological Discrete Vector Field

3 Fast Computation of Betti Numbers on 3D Cubical Complexes
- Introduction
- Algorithm
- Results
- Conclusion

4 Measuring Holes

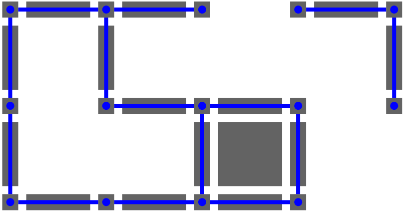
5 Conclusion

Ingredients

(1) β_0 is the number of connected components
(2) Duality
(3) Euler-Poincaré formula

Ingredient (1)

Let K be a 3D cubical complex. Consider the graph $G_0(K)$



Proposition 4.3

$\beta_0(K) =$ number of connected components in $G_0(K)$

Ingredient (2)

Proposition 4.4

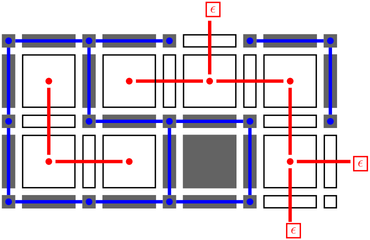
Let $K \subset L$ be two 3D cubical complexes such that $\beta(L) = (1, 0, 0, 0)$. Then,

$$\beta_q(K) = \begin{cases} \beta_1(L - K) + 1 & \text{if } q = 0 \\ \beta_{q+1}(L - K) & \text{else} \end{cases}$$

Thus, $\beta_2(K) = \beta_3(L - K)$

Ingredient (2)

Let $K \subset L$ be two 3D cubical complexes. Consider the graph $G_3(L - K)$



Proposition 4.5

$\beta_3(L - K) =$ number of connected components in $G_3(L - K)$ minus one.

Ingredient (3)

Euler-Poincaré Formula

$$\begin{aligned}\chi(K) &= |K_0| - |K_1| + |K_2| - |K_3| \\ &= \beta_0(K) - \beta_1(K) + \beta_2(K)\end{aligned}$$

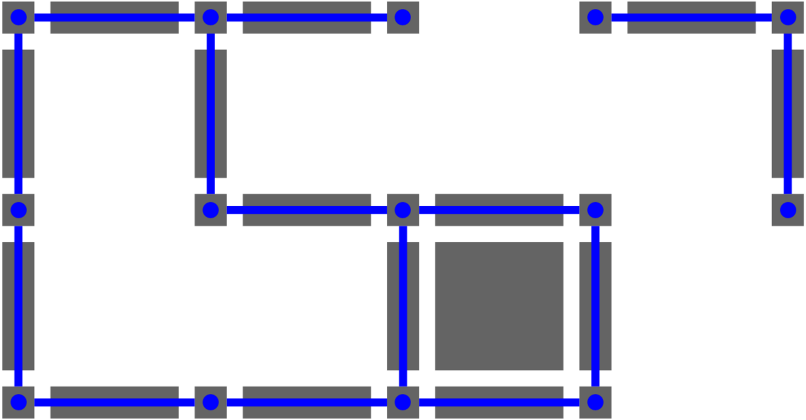
Thus, $\beta_1(K) = \beta_0(K) + \beta_2(K) - \chi(K)$

Computing the Betti numbers

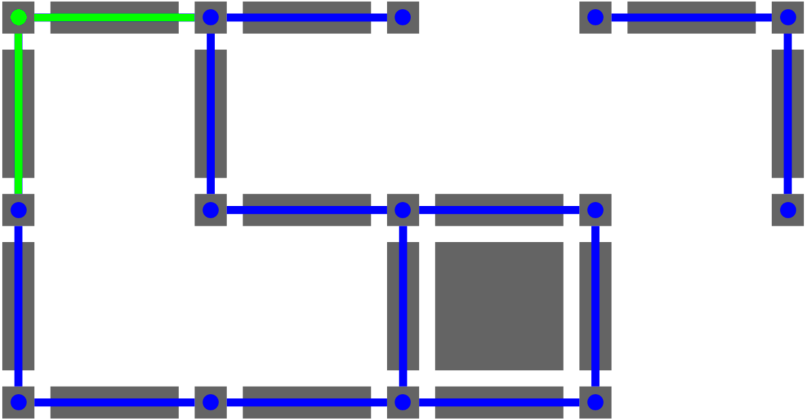
1. $\beta_0 \leftarrow$ number of connected components of $G_0(K)$
2. $\beta_2 \leftarrow$ number of connected components of $G_3(L - K)$ - 1
3. $\beta_1 \leftarrow \beta_0 + \beta_2 - \chi(K)$

- Linear time and space complexity
- We propose two versions for implementing this method

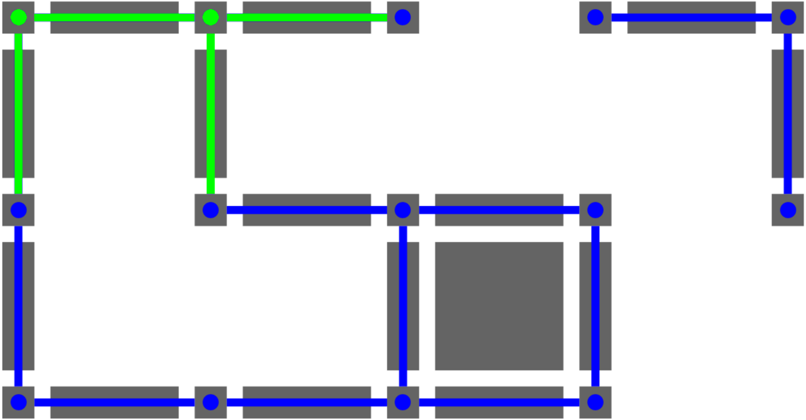
Sequential algorithm: BFS, iterative



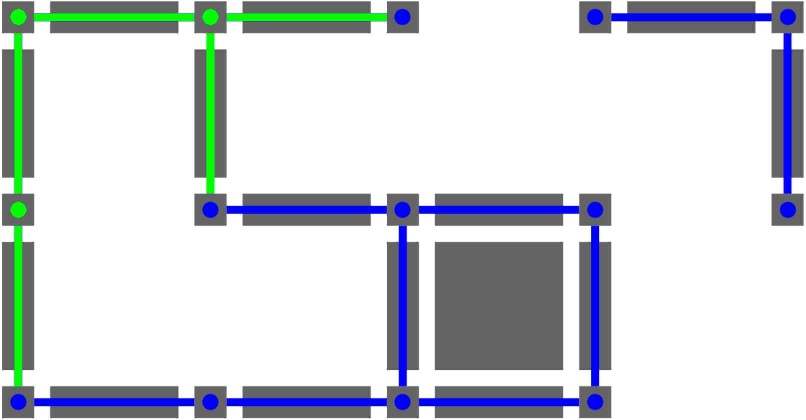
Sequential algorithm: BFS, iterative



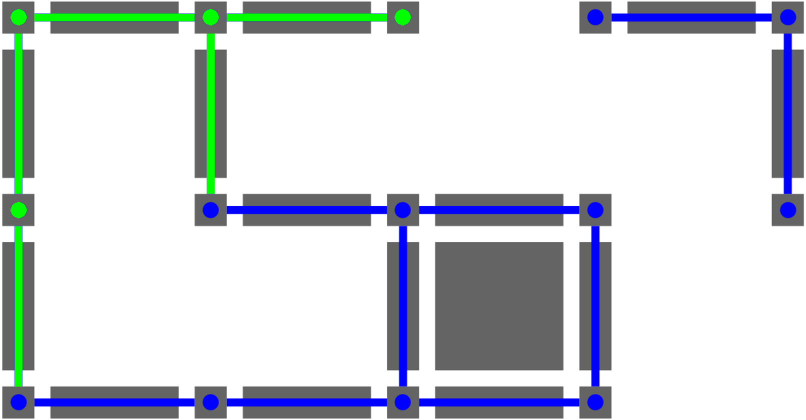
Sequential algorithm: BFS, iterative



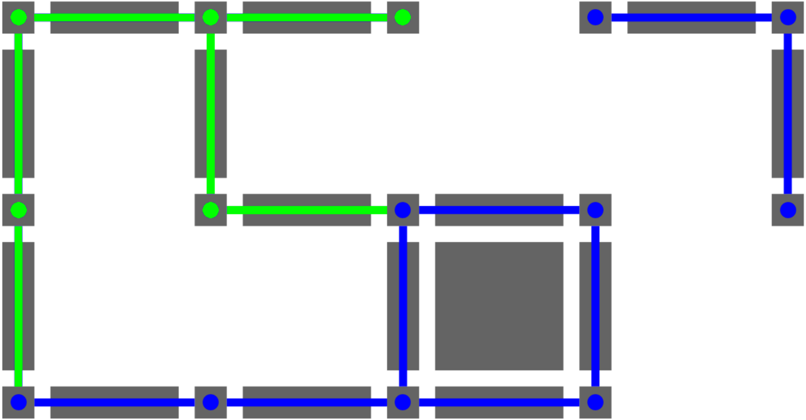
Sequential algorithm: BFS, iterative



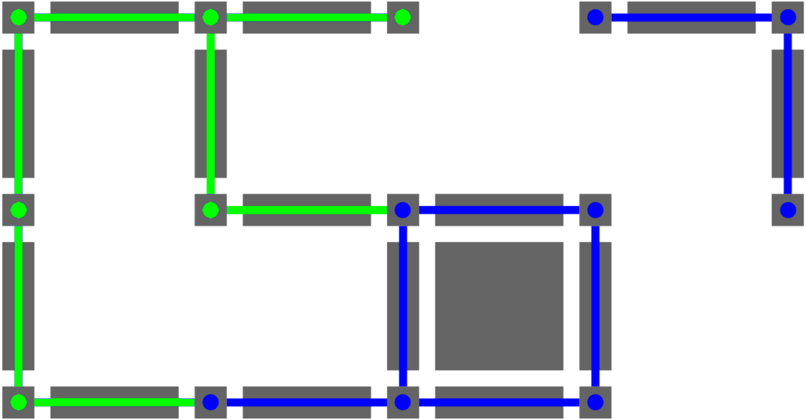
Sequential algorithm: BFS, iterative



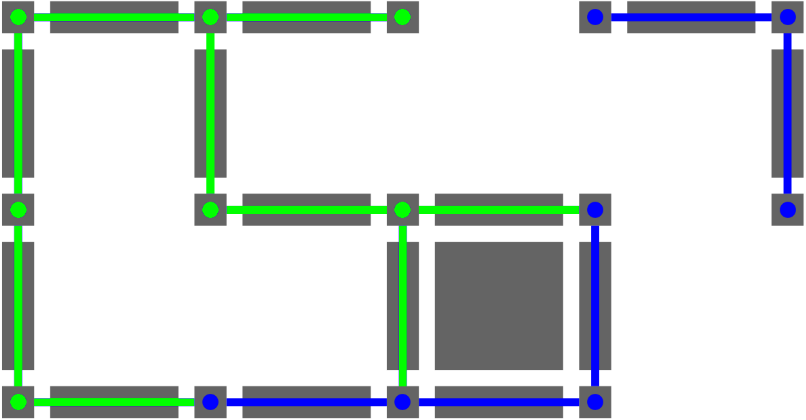
Sequential algorithm: BFS, iterative



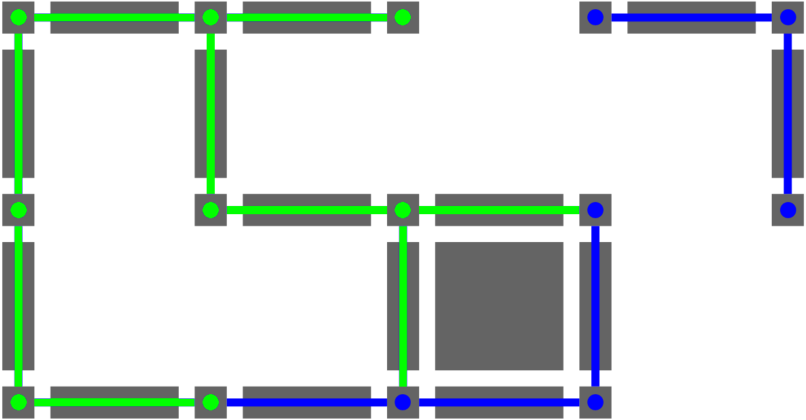
Sequential algorithm: BFS, iterative



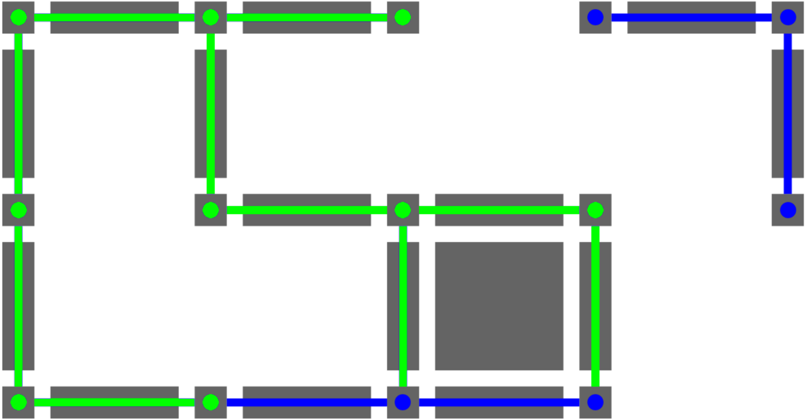
Sequential algorithm: BFS, iterative



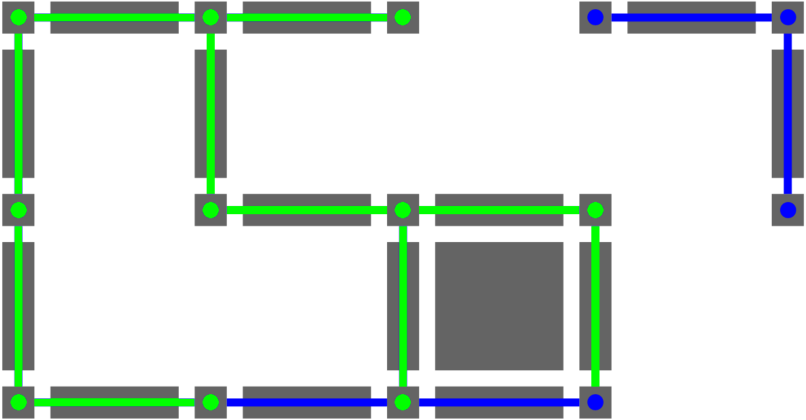
Sequential algorithm: BFS, iterative



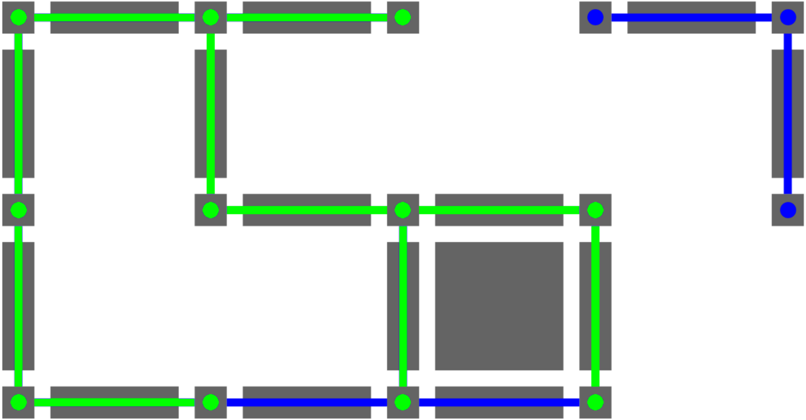
Sequential algorithm: BFS, iterative



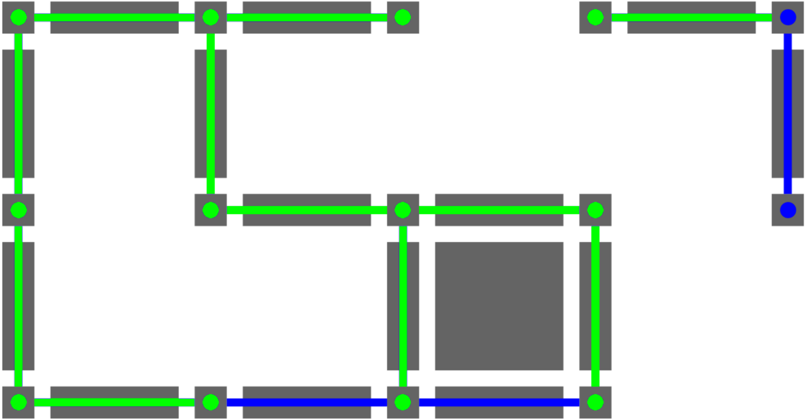
Sequential algorithm: BFS, iterative



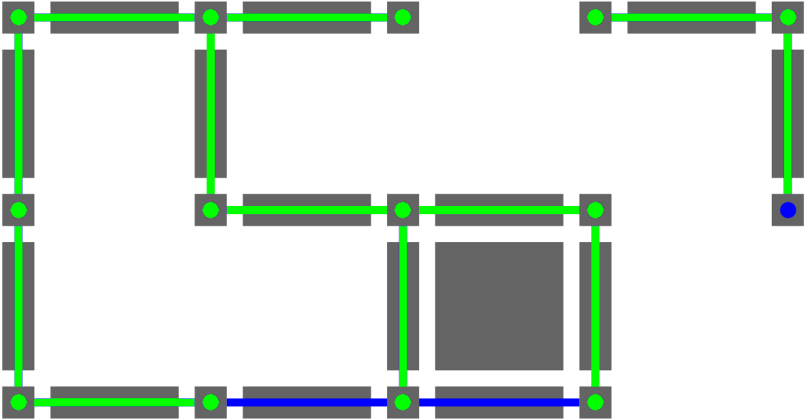
Sequential algorithm: BFS, iterative



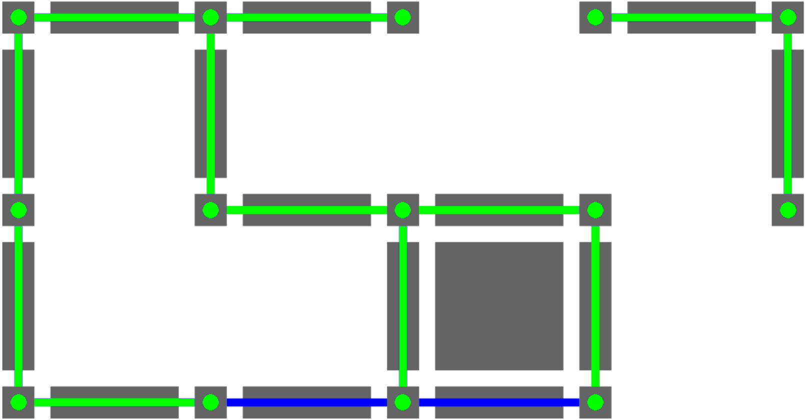
Sequential algorithm: BFS, iterative



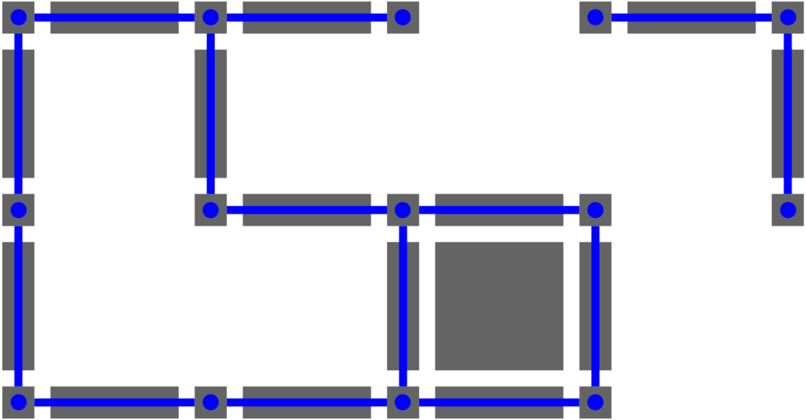
Sequential algorithm: BFS, iterative



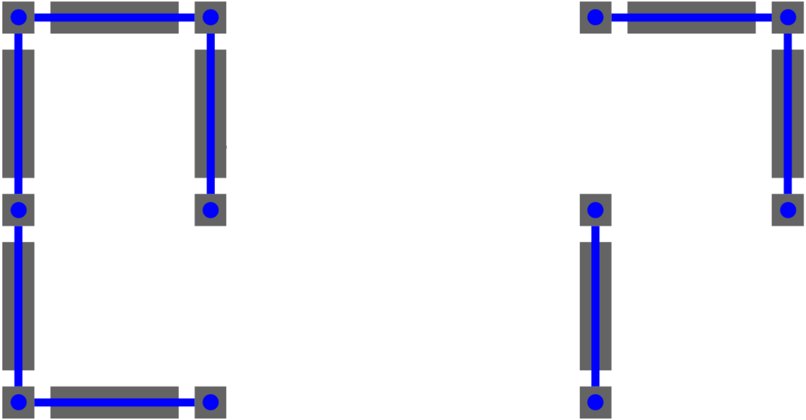
Sequential algorithm: BFS, iterative



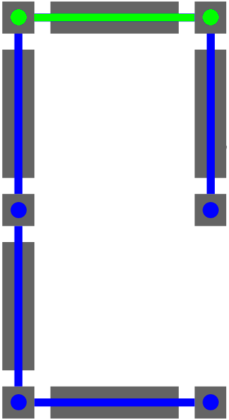
Recursive algorithm: divide-and-conquer, union-find data set, partial parallelization

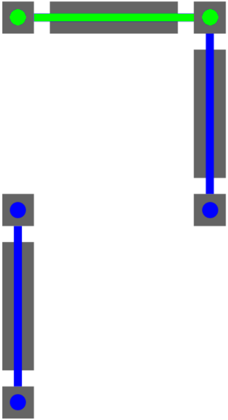


Recursive algorithm: divide-and-conquer, union-find data set, partial parallelization

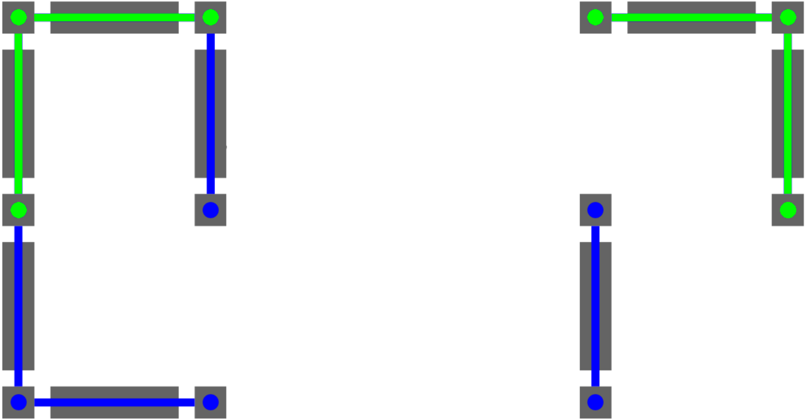


Recursive algorithm: divide-and-conquer, union-find data set, partial parallelization

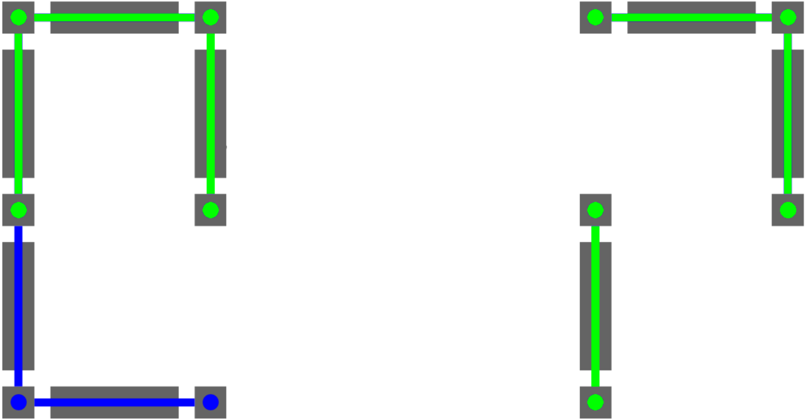




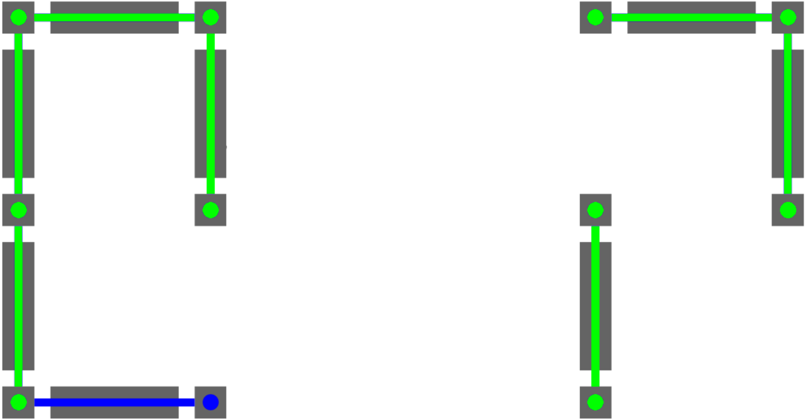
Recursive algorithm: divide-and-conquer, union-find data set, partial parallelization



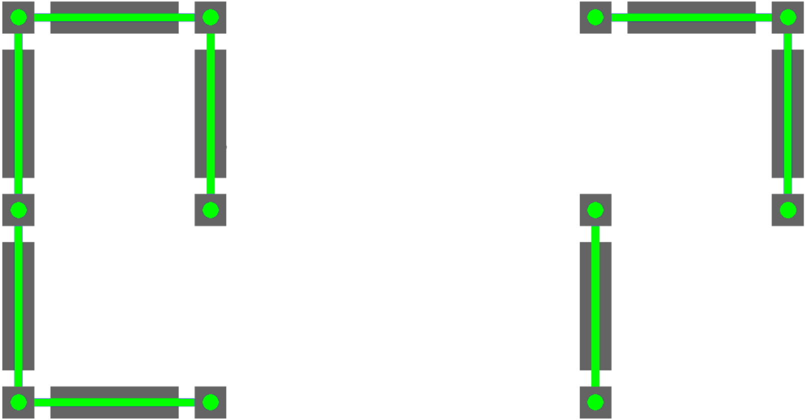
Recursive algorithm: divide-and-conquer, union-find data set, partial parallelization



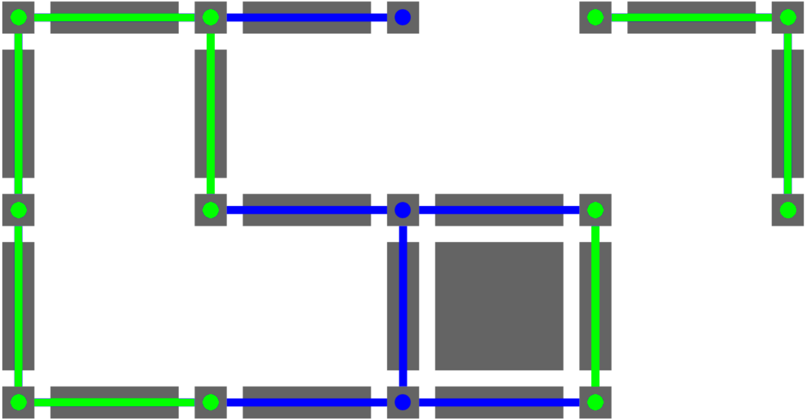
Recursive algorithm: divide-and-conquer, union-find data set, partial parallelization



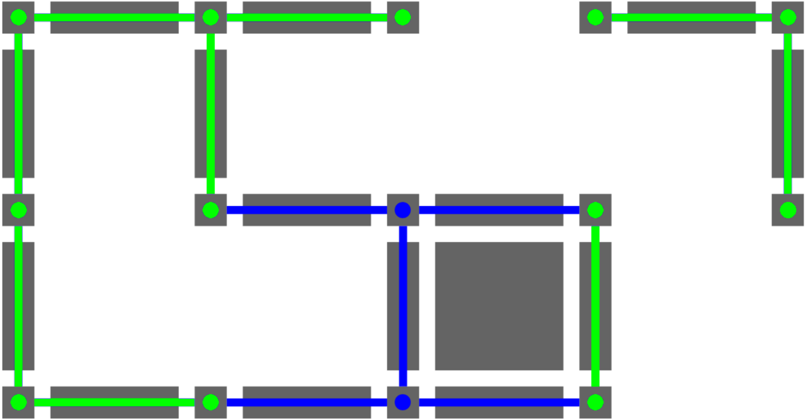
Recursive algorithm: divide-and-conquer, union-find data set, partial parallelization



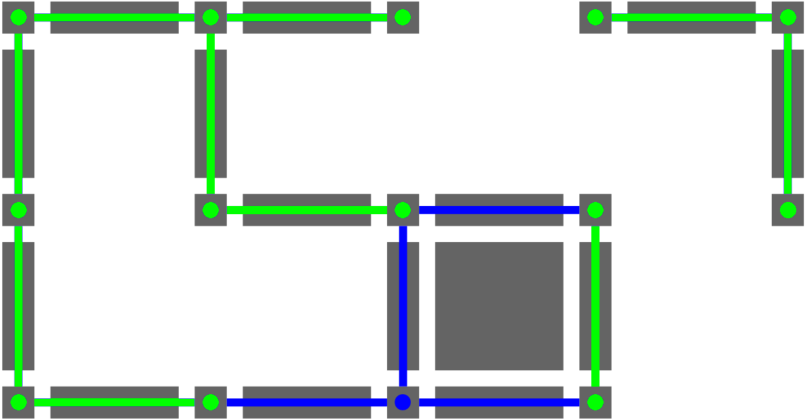
Recursive algorithm: divide-and-conquer, union-find data set, partial parallelization



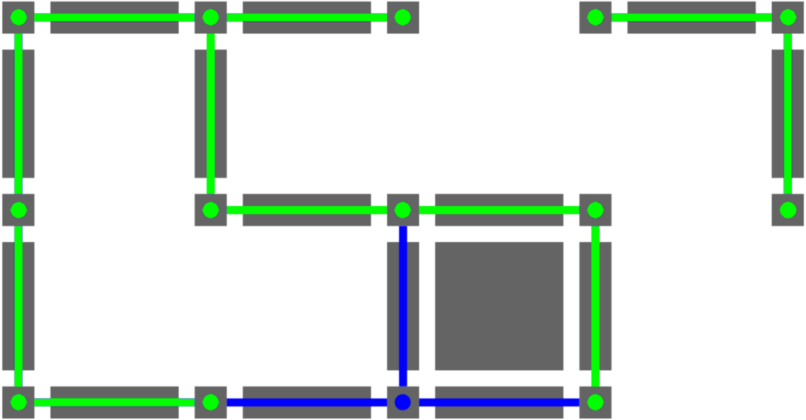
Recursive algorithm: divide-and-conquer, union-find data set, partial parallelization



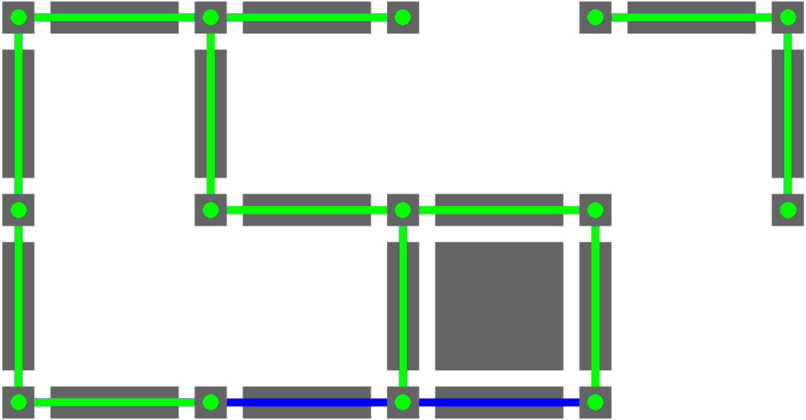
Recursive algorithm: divide-and-conquer, union-find data set, partial parallelization



Recursive algorithm: divide-and-conquer, union-find data set, partial parallelization



Recursive algorithm: divide-and-conquer, union-find data set, partial parallelization



Sections

1 Introduction and Preliminaries

2 The Homological Discrete Vector Field

3 Fast Computation of Betti Numbers on 3D Cubical Complexes
- Introduction
- Algorithm
- Results
- Conclusion

4 Measuring Holes

5 Conclusion

Comparison against `CAPD::RedHom`[5] library

Size	RedHom	VB-s	VB-r	VB-rp
51^3	0.1842	0.0026	0.0026	0.0023
101^3	1.268	0.0142	0.0148	0.0091
201^3	10.78	0.1309	0.1232	0.0552
301^3	40.89	0.4303	0.4176	0.1583
401^3	101.26	1.436	0.983	0.3092
501^3	—	3.609	1.977	0.5494

Table: Execution time (in seconds) versus the size of the cubical complex.

Space is the problem, not time.

[5] [Juda and Mrozek, 2014]

Sections

1 Introduction and Preliminaries

2 The Homological Discrete Vector Field

3 Fast Computation of Betti Numbers on 3D Cubical Complexes
- Introduction
- Algorithm
- Results
- Conclusion

4 Measuring Holes

5 Conclusion

Conclusion:

- Simple algorithm for Betti numbers relying on connected components computation
- Linear time complexity
- Combinatorial and constructive proofs
- Implementation published under GNU GPL v3
- More effective than available algorithms

Conclusion:

- Simple algorithm for Betti numbers relying on connected components computation
- Linear time complexity
- Combinatorial and constructive proofs
- Implementation published under GNU GPL v3
- More effective than available algorithms

Conclusion:

- Simple algorithm for Betti numbers relying on connected components computation
- Linear time complexity
- Combinatorial and constructive proofs
- Implementation published under GNU GPL v3
- More effective than available algorithms

Conclusion:

- Simple algorithm for Betti numbers relying on connected components computation
- Linear time complexity
- Combinatorial and constructive proofs
- Implementation published under GNU GPL v3
- More effective than available algorithms

Conclusion:

- Simple algorithm for Betti numbers relying on connected components computation
- Linear time complexity
- Combinatorial and constructive proofs
- Implementation published under GNU GPL v3
- More effective than available algorithms

Sections

1 Introduction and Preliminaries

2 The Homological Discrete Vector Field

3 Fast Computation of Betti Numbers on 3D Cubical Complexes

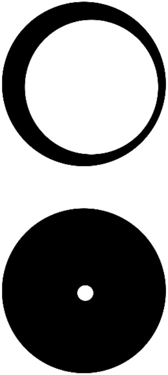
4 Measuring Holes
- Introduction
- Definition
- Thickness-breadth balls
- Applications
- Conclusion

5 Conclusion

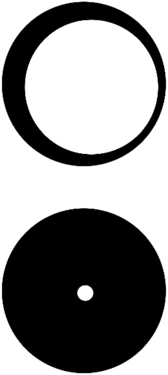
- We can know **how many** holes there are in an object
- We cannot know **where** or **how** they are

Size of a hole

The 1st one is *bigger* than the 2nd one

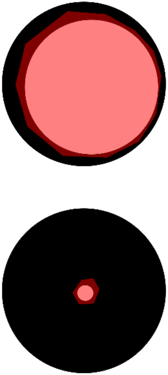


The 2nd one is *thicker* than the 1st one

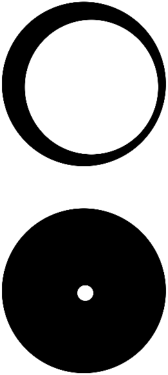


Size of a hole

The 1st one is *bigger* than the 2nd one

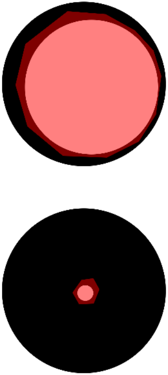


The 2nd one is *thicker* than the 1st one

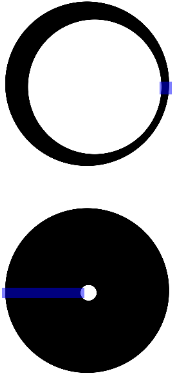


Size of a hole

The 1st one is *bigger* than the 2nd one

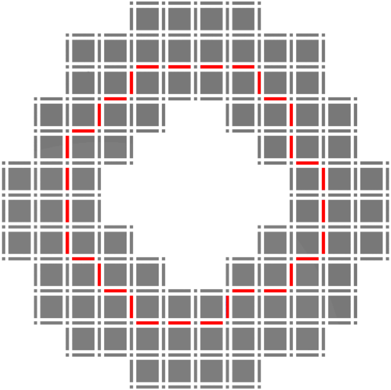


The 2nd one is *thicker* than the 1st one

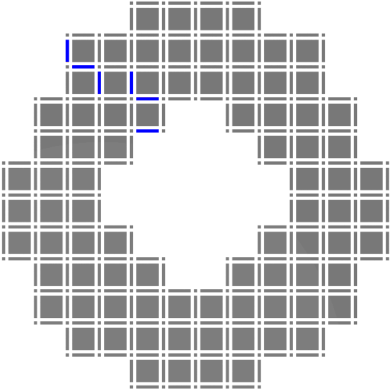


Representing a hole

Homology

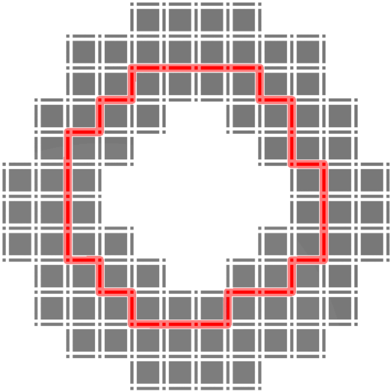


Cohomology

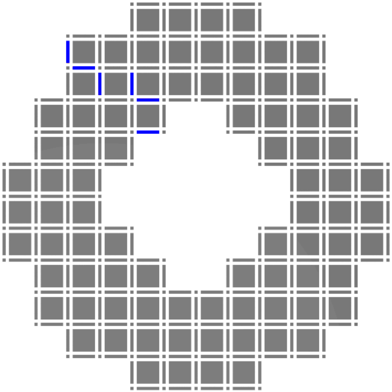


Representing a hole

Homology

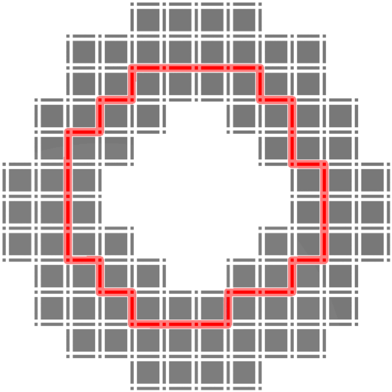


Cohomology

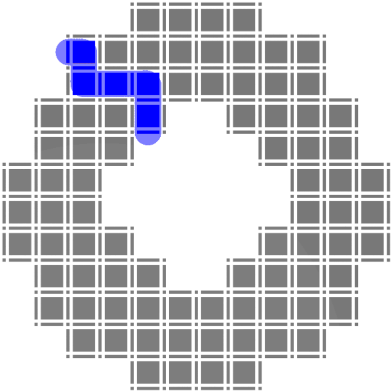


Representing a hole

Homology

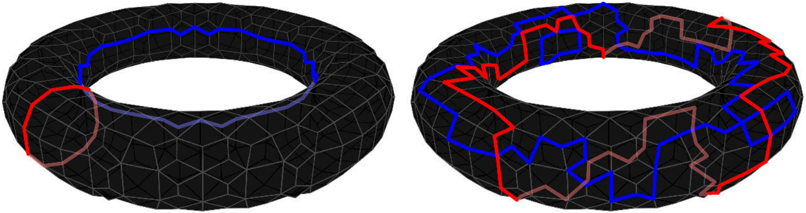


Cohomology



Representing a hole

Do homology generators really represent holes?



Geometry	+	Topology
↓		↓
Signed distance transform		Persistent homology

Sections

1 Introduction and Preliminaries

2 The Homological Discrete Vector Field

3 Fast Computation of Betti Numbers on 3D Cubical Complexes

4 Measuring Holes
- Introduction
- Definition
- Thickness-breadth balls
- Applications
- Conclusion

5 Conclusion

Signed distance transform

Let O be a discrete object,

$$sdt_O(x) = \begin{cases} -d(x, O^c) & \text{if } x \in O \\ d(x, O) & \text{if } x \notin O \end{cases}$$

Figure: Sublevel sets of the signed distance form

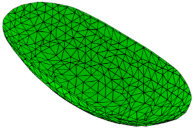
Persistent homology

Given a filtration F, we can define its set of persistence intervals through its persistent homology groups.

- These intervals tell the lifetime of the holes in the filtration
- They are represented as a set of points $PD(F)$ in $\mathbb{R}^2$

Example

- height: 5
- β_0: **1**
- β_1: 0
- β_2: 0

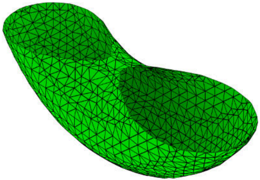


Persistence intervals:

- Dimension 0:
- Dimension 1:
- Dimension 2:

Example

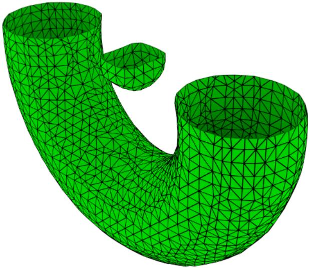
- height: 7
- β_0: 1
- β_1: **1**
- β_2: 0



Persistence intervals:

- Dimension 0:
- Dimension 1:
- Dimension 2:

Example

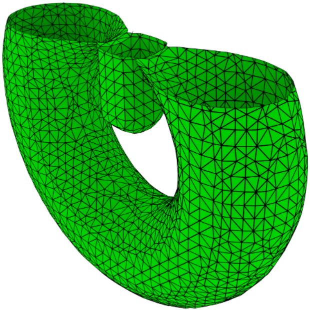


- height: 11
- β_0: **2**
- β_1: 1
- β_2: 0

Persistence intervals:

- Dimension 0:
- Dimension 1:
- Dimension 2:

Example

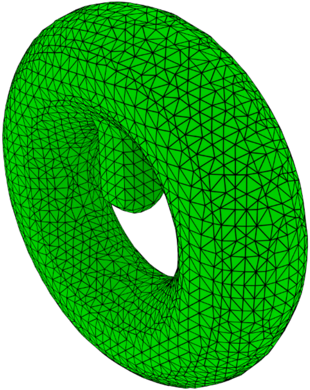


- height: 13
- β_0: **1**
- β_1: **2**
- β_2: 0

Persistence intervals:

- Dimension 0: $(10, 13)$
- Dimension 1:
- Dimension 2:

Example

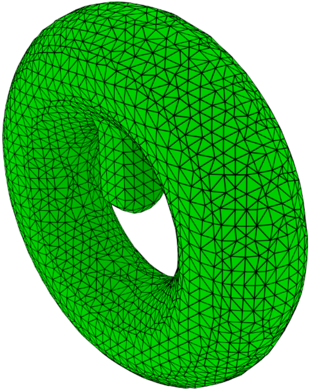


- height: 17
- β_0: 1
- β_1: 2
- β_2: **1**

Persistence intervals:

- Dimension 0: $(10, 13)$
- Dimension 1:
- Dimension 2:

Example

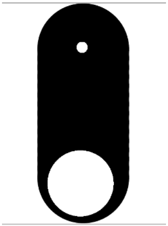


- height:
- β_0: 1
- β_1: 2
- β_2: 1

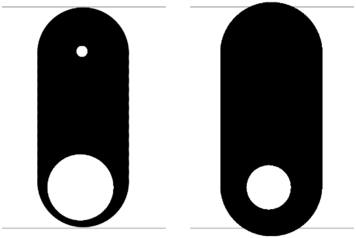
Persistence intervals:

- Dimension 0: $(10, 13)$, $(0, \infty)$
- Dimension 1: $(7, \infty)$, $(13, \infty)$
- Dimension 2: $(17, \infty)$

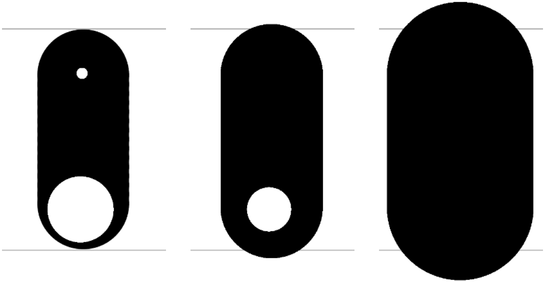
Persistent homology with signed distance transform



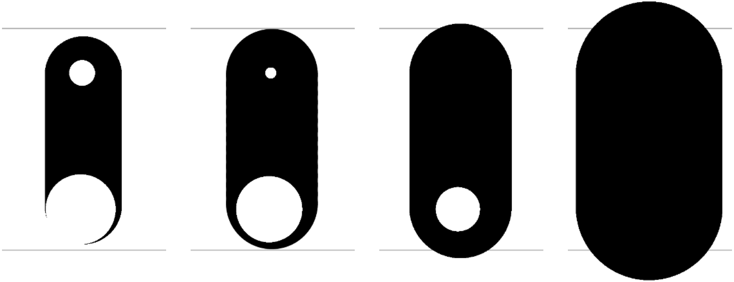
Persistent homology with signed distance transform



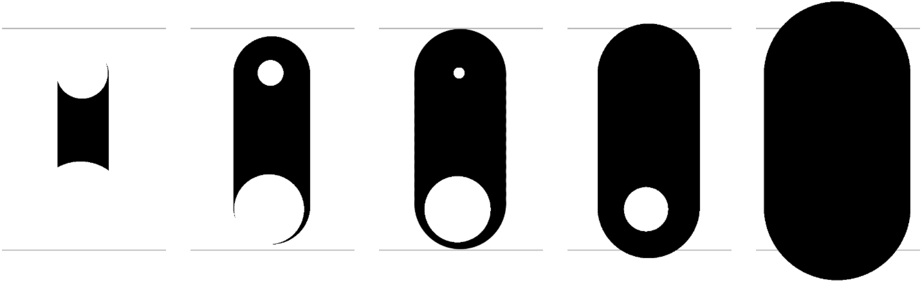
Persistent homology with signed distance transform



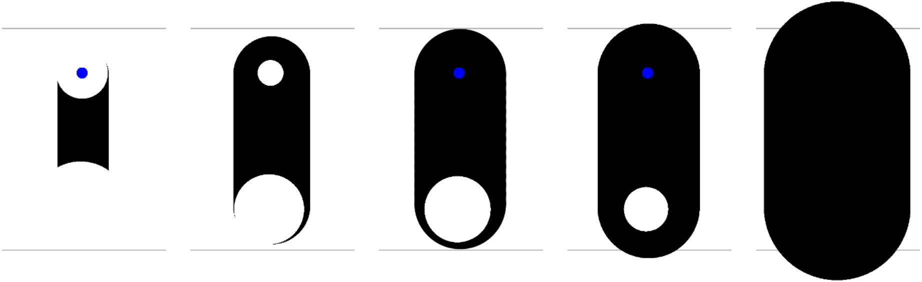
Persistent homology with signed distance transform



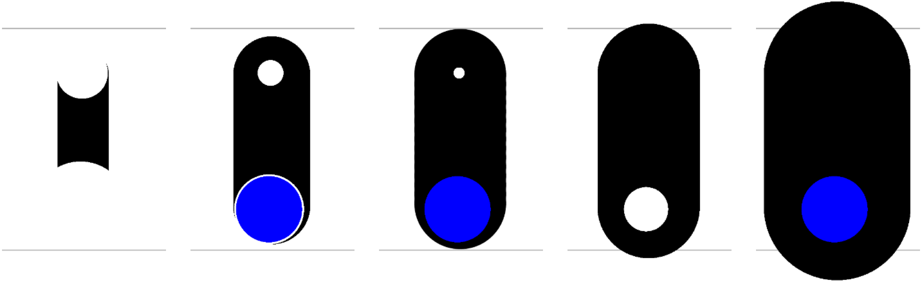
Persistent homology with signed distance transform



Persistent homology with signed distance transform



Persistent homology with signed distance transform



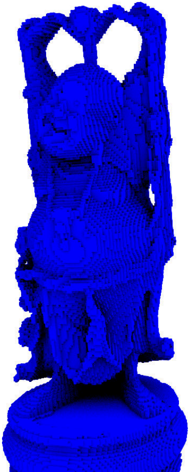
Thickness and breadth (Definition 5.1)

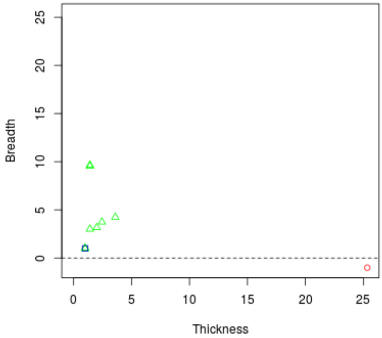
Let O be a discrete object and F the filtration defined by the sublevel sets of its signed distance transform. Let $TB(O) = \{(-x, y) \in PD(F) \mid x \leq 0, y \geq 0\}$. Its intervals are the thickness-breadth pairs of O

- There is a thickness-breadth pair (t, b) for each hole of O
- t is the thickness of the hole and b, its breadth

Thickness-breadth diagram

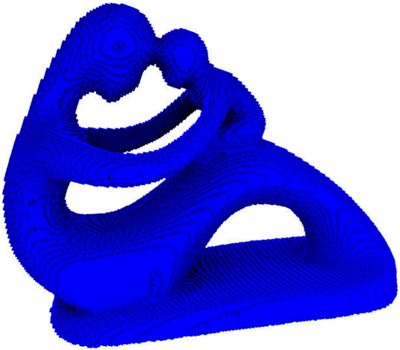
Thickness-breadth pairs can be represented like persistence diagrams

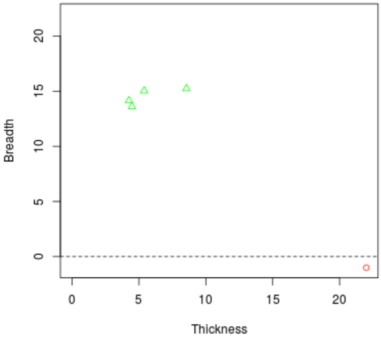




Thickness-breadth diagram

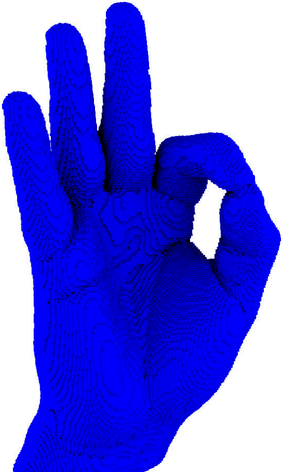
Thickness-breadth pairs can be represented like persistence diagrams

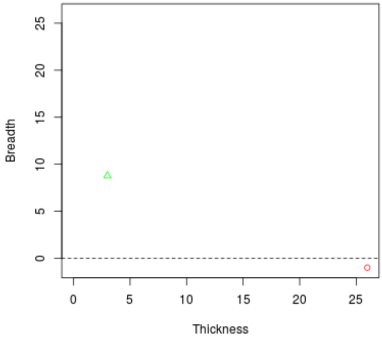




Thickness-breadth diagram

Thickness-breadth pairs can be represented like persistence diagrams





Theorem 5.2

Let X and Y be two 3D discrete objects. Let us call

$$\delta = d_H(X, Y) + d_H(\mathbb{Z}^3 \setminus X, \mathbb{Z}^3 \setminus Y) + 2\sqrt{3}$$

Thus, for every thickness-breadth pair $p_X = (x, y)$ of X such that $x, y > \delta$, there exists another thickness-breadth pair $p_Y = (x', y')$ of Y such that

$$||p_X - p_Y||_\infty \leq \delta$$

Sections

1 Introduction and Preliminaries

2 The Homological Discrete Vector Field

3 Fast Computation of Betti Numbers on 3D Cubical Complexes

4 Measuring Holes
- Introduction
- Definition
- Thickness-breadth balls
- Applications
- Conclusion

5 Conclusion

Thickness and breadth ball

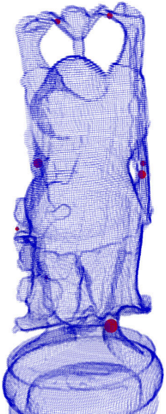
Let be (t, b) a TB-pair and (σ, τ) its pair of cells

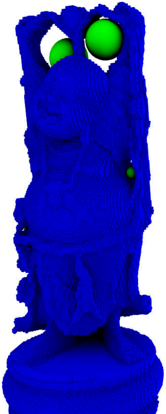
- The thickness ball of (t, b) is the ball of radius t centered at σ
- The breadth ball of (t, b) is the ball of radius b centered at τ

Thickness and breadth ball

Let be (t, b) a TB-pair and (σ, τ) its pair of cells

- The thickness ball of (t, b) is the ball of radius t centered at σ
- The breadth ball of (t, b) is the ball of radius b centered at τ

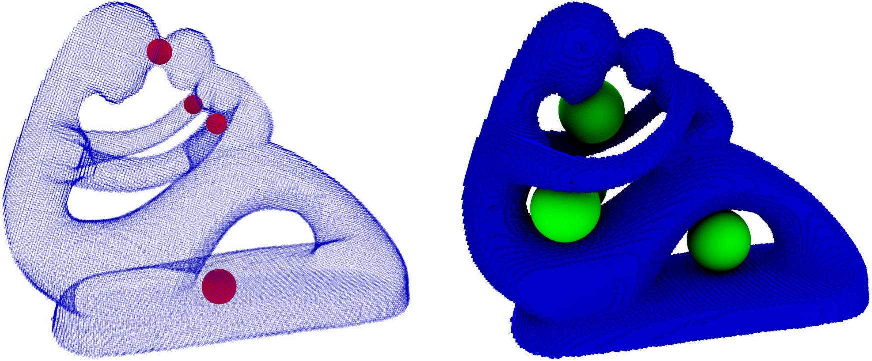




Thickness and breadth ball

Let be (t, b) a TB-pair and (σ, τ) its pair of cells

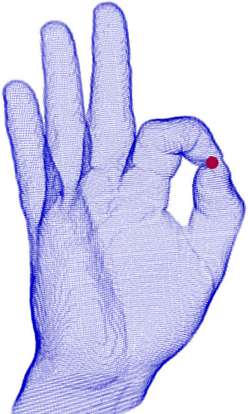
- The thickness ball of (t, b) is the ball of radius t centered at σ
- The breadth ball of (t, b) is the ball of radius b centered at τ

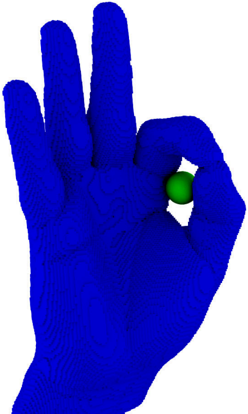


Thickness and breadth ball

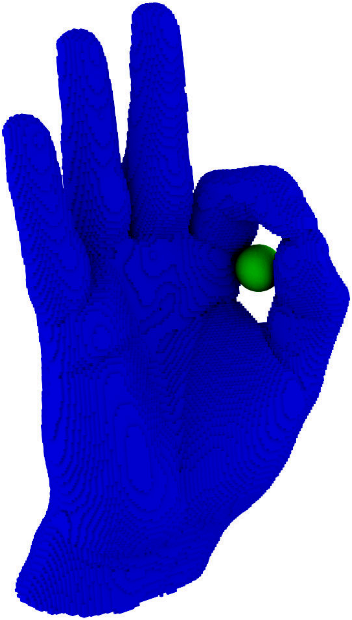
Let be (t, b) a TB-pair and (σ, τ) its pair of cells

- The thickness ball of (t, b) is the ball of radius t centered at σ
- The breadth ball of (t, b) is the ball of radius b centered at τ

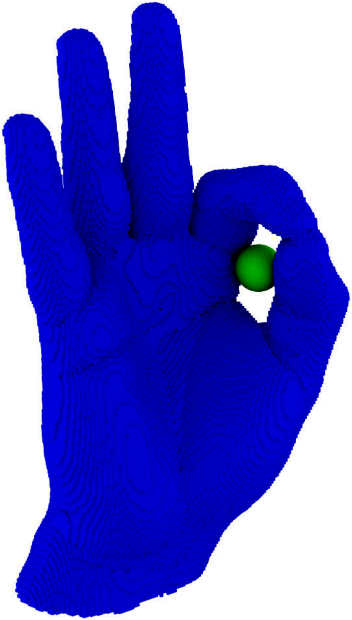




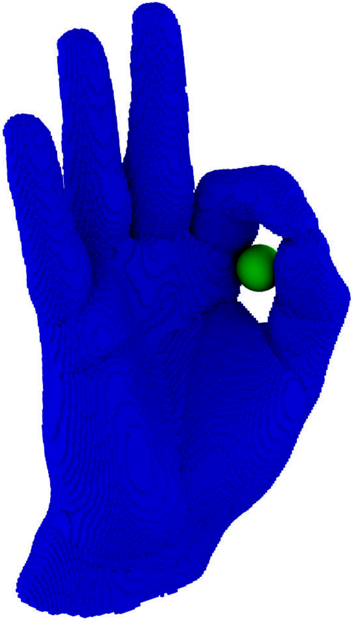
- Breadth ball
- Homology generator
- Close hole

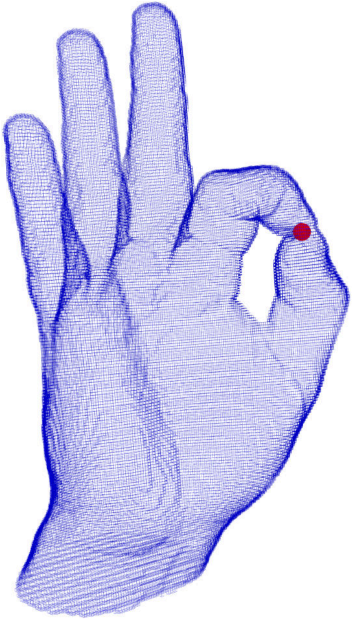


- Breadth ball
- Homology generator
- Close hole



- Breadth ball
- Homology generator
- Close hole





- Thickness ball
- Cohomology generator
- Open hole

Sections

1 Introduction and Preliminaries

2 The Homological Discrete Vector Field

3 Fast Computation of Betti Numbers on 3D Cubical Complexes

4 Measuring Holes
- Introduction
- Definition
- Thickness-breadth balls
- Applications
- Conclusion

5 Conclusion

(co)homology generators

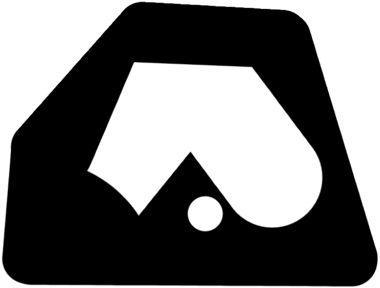
"A good homology generator should be close to a breadth ball"

Algorithms

- Algorithm 6: TB pair $\mapsto$ homology generator
- Algorithm 7: TB pair $\mapsto$ cohomology generator

(co)homology generators

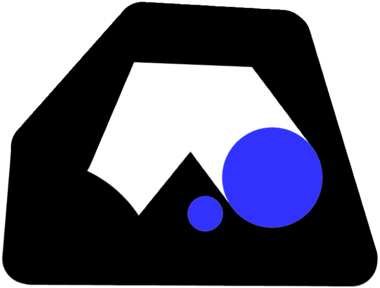
Algorithm 6 (homology generator)



- Discrete object
- Breadth balls
- Filtration

(co)homology generators

Algorithm 6 (homology generator)



- Discrete object
- Breadth balls
- Filtration

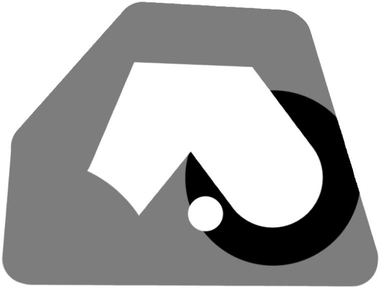
(co)homology generators

Algorithm 6 (homology generator)

- Discrete object
- Breadth balls
- Filtration

(co)homology generators

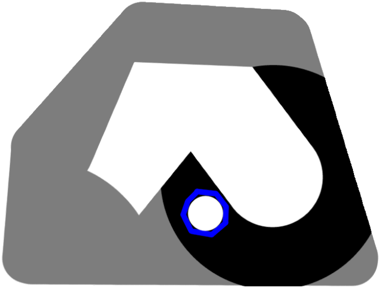
Algorithm 6 (homology generator)



- Discrete object
- Breadth balls
- Filtration

(co)homology generators

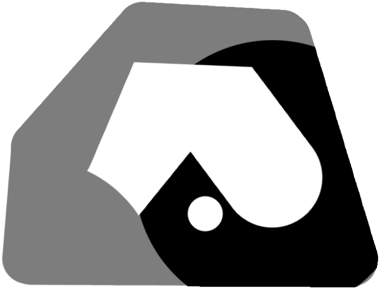
Algorithm 6 (homology generator)



- Discrete object
- Breadth balls
- Filtration

(co)homology generators

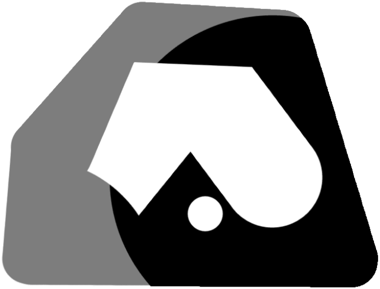
Algorithm 6 (homology generator)



- Discrete object
- Breadth balls
- Filtration

(co)homology generators

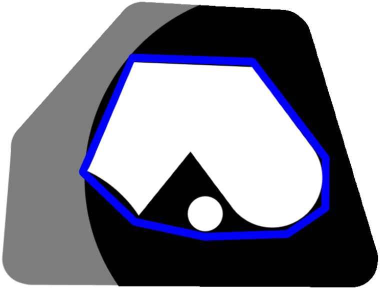
Algorithm 6 (homology generator)



- Discrete object
- Breadth balls
- Filtration

(co)homology generators

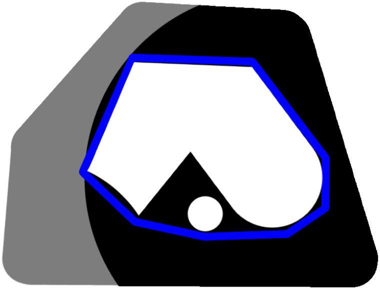
Algorithm 6 (homology generator)



- Discrete object
- Breadth balls
- Filtration

(co)homology generators

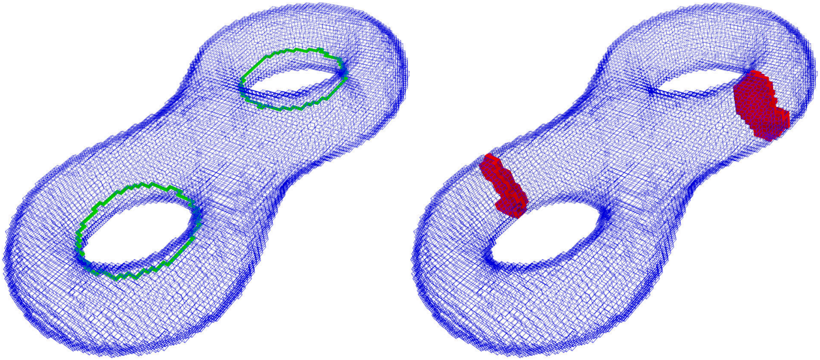
Algorithm 6 (homology generator)



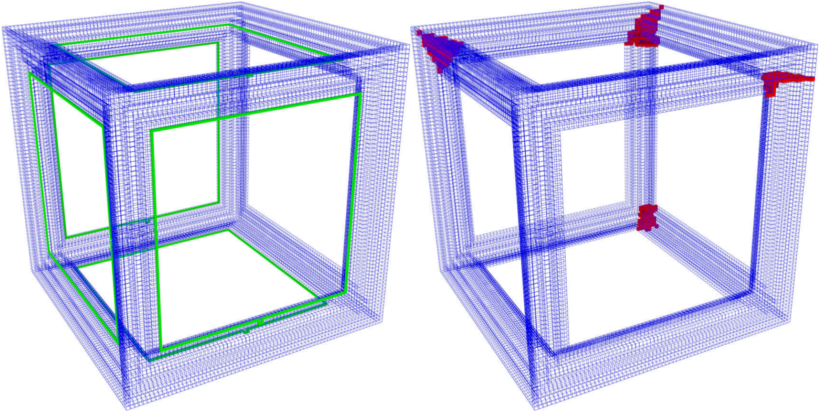
- Discrete object
- Breadth balls
- Filtration

A similar (dual) approach produces **co**homology generators!

(co)homology generators — examples



(co)homology generators — examples



Opening and closing holes

- Thickness balls (and cohomology generators) seem to tell where to **break** a hole
- Breadth balls (and homology generators) seem to tell where to **fill** a hole

Opening and closing holes

- Thickness balls (and cohomology generators) seem to tell where to **break** a hole
- Breadth balls (and homology generators) seem to tell where to **fill** a hole

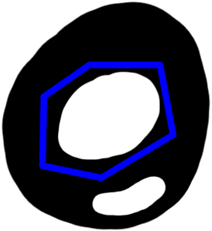
Opening and closing holes

K cubical complex, x cycle, S set of cubes

Opening the cycle x

S opens the cycle x if

1. $K - S$ is a cubical complex
2. $[x] \notin \text{im}(\iota)$
3. $\iota : H(K - S) \to H(K)$ is injective



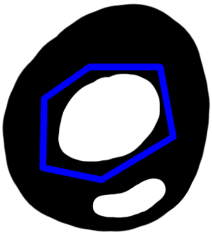
Opening and closing holes

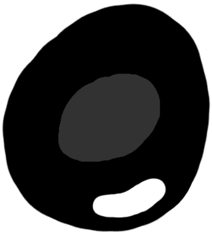
K cubical complex, x cycle, S set of cubes

Closing the cycle x

S closes the cycle x if

1. $K \cup S$ is a cubical complex
2. $[x] \in \ker(\iota)$
3. $\iota : H(K) \to H(K \cup S)$ is surjective



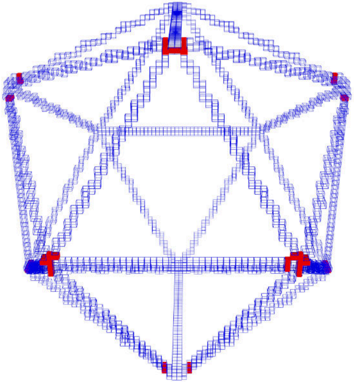


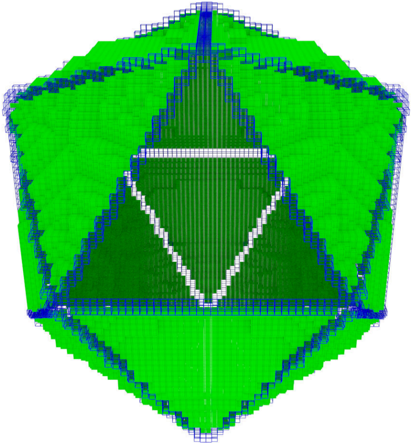
Opening and closing holes

Algorithms

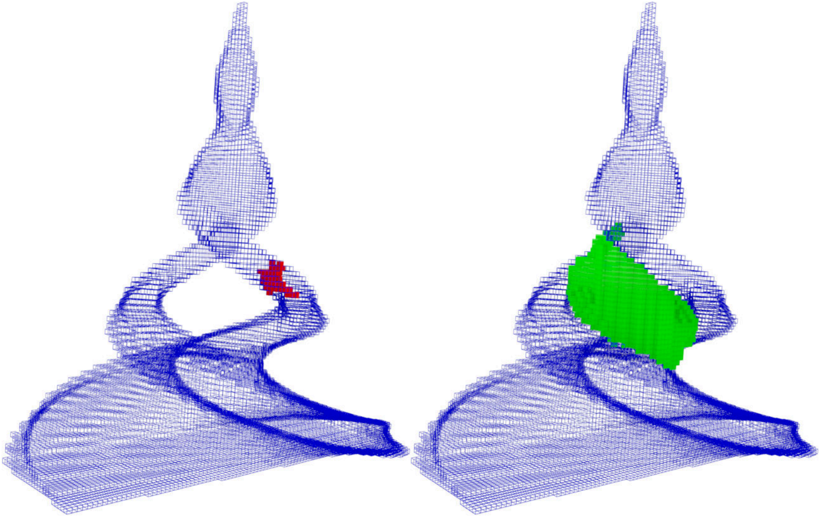
- Algorithm 8: TB pair $\mapsto$ hole opening
- Algorithm 9: TB pair $\mapsto$ hole closing **without surjectivity condition**

Opening and closing holes — examples

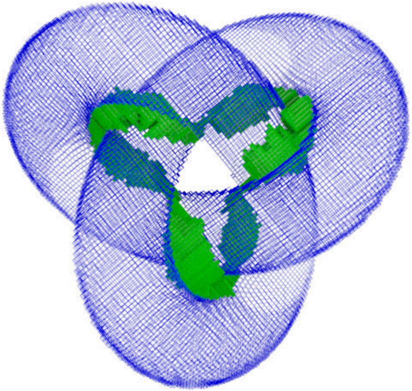




Opening and closing holes — examples



Opening and closing holes — examples



Sections

1 Introduction and Preliminaries

2 The Homological Discrete Vector Field

3 Fast Computation of Betti Numbers on 3D Cubical Complexes

4 Measuring Holes
- Introduction
- Definition
- Thickness-breadth balls
- Applications
- Conclusion

5 Conclusion

Conclusion:

- Topological-geometrical signature of objects
- Robust to noise ⇒ suitable for real applications
- Alternative visualization of holes
- Heuristics for small homology and cohomology generators
- Heuristics for opening and closing holes

Conclusion:

- Topological-geometrical signature of objects
- Robust to noise ⇒ suitable for real applications
- Alternative visualization of holes
- Heuristics for small homology and cohomology generators
- Heuristics for opening and closing holes

Conclusion:

- Topological-geometrical signature of objects
- Robust to noise $\Rightarrow$ suitable for real applications
- Alternative visualization of holes
- Heuristics for small homology and cohomology generators
- Heuristics for opening and closing holes

Conclusion:

- Topological-geometrical signature of objects
- Robust to noise ⇒ suitable for real applications
- Alternative visualization of holes
- Heuristics for small homology and cohomology generators
- Heuristics for opening and closing holes

Sections

1 Introduction and Preliminaries

2 The Homological Discrete Vector Field

3 Fast Computation of Betti Numbers on 3D Cubical Complexes

4 Measuring Holes

5 Conclusion

Context

Contributions

Discrete object

↓

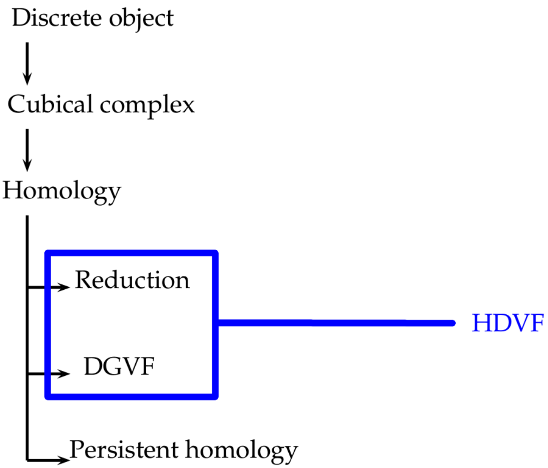
Cubical complex

↓

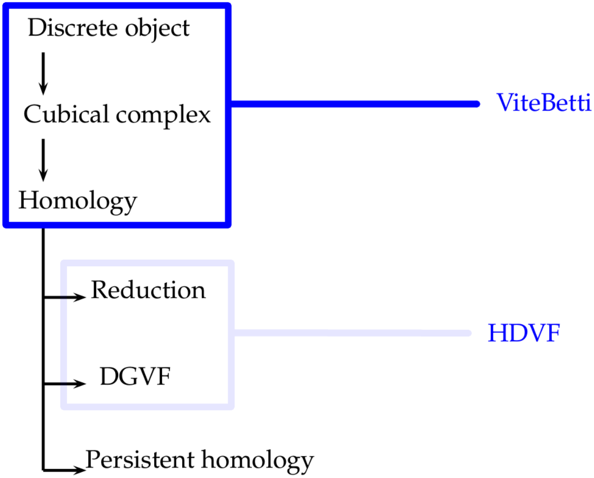
Homology

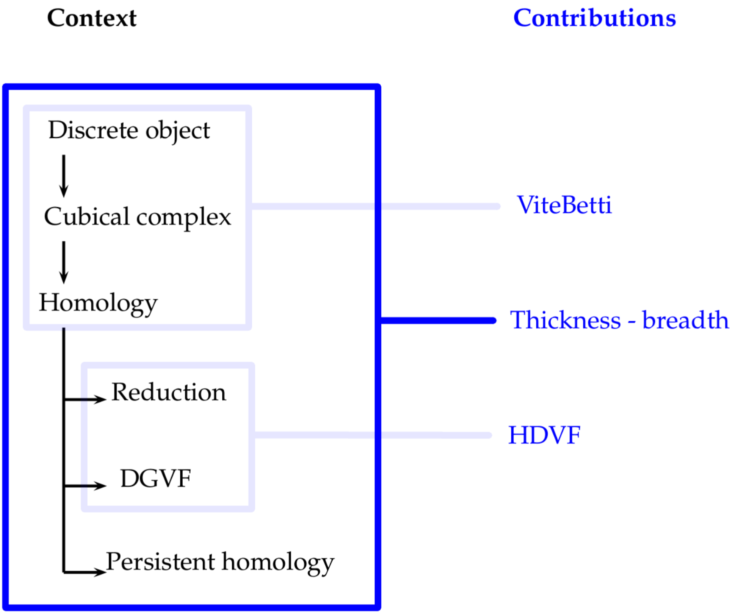
- → Reduction
- → DGVF
- → Persistent homology

Context **Contributions**



Context **Contributions**




Context
Contributions
Discrete object
Cubical complex
Homology
Reduction
DGVF
Persistent homology
ViteBetti
Thickness - breadth
HDVF

Perspectives:

1. HDVF
   - Every CW complex admits a perfect HDVF?
   - Use the operations for comparing HDVFs
   - Compute zigzag persistent homology with HDVFs
2. ViteBetti
   - Apply it directly on the discrete object
   - Try component labeling algorithms from digital geometry context
   - Process complex by slices
3. Thickness and breadth
   - To find real world applications
   - To formalize the geometric intuition
   - Algorithm for closing holes
   - Simplicial complexes

Perspectives:

1. HDVF
    - Every CW complex admits a perfect HDVF?
    - Use the operations for comparing HDVFs
    - Compute zigzag persistent homology with HDVFs
2. ViteBetti
    - Apply it directly on the discrete object
    - Try component labeling algorithms from digital geometry context
    - Process complex by slices
3. Thickness and breadth
    - To find real world applications
    - To formalize the geometric intuition
    - Algorithm for closing holes
    - Simplicial complexes

Perspectives:

1. HDVF
   - Every CW complex admits a perfect HDVF?
   - Use the operations for comparing HDVFs
   - Compute zigzag persistent homology with HDVFs
2. ViteBetti
   - Apply it directly on the discrete object
   - Try component labeling algorithms from digital geometry context
   - Process complex by slices
3. Thickness and breadth
   - To find real world applications
   - To formalize the geometric intuition
   - Algorithm for closing holes
   - Simplicial complexes

Perspectives:

1. HDVF
   - Every CW complex admits a perfect HDVF?
   - Use the operations for comparing HDVFs
   - Compute zigzag persistent homology with HDVFs
2. ViteBetti
   - Apply it directly on the discrete object
   - Try component labeling algorithms from digital geometry context
   - Process complex by slices
3. Thickness and breadth
   - To find real world applications
   - To formalize the geometric intuition
   - Algorithm for closing holes
   - Simplicial complexes

Thanks **Merci** **Grazie** **Gracias**