Computational topology with Gudhi library.

Paweł Dłotko with Jean-Daniel Boissonat, Marc Glisse, François Godi, Clément Jamin, Siargey Kachanovich, Clément Maria, Vincent Rouvreau and David Salinas

Swansea University *and DataShape, Inria*

What does Gudhi mean?

What does Gudhi should mean (to us)?

What is Gudhi project?

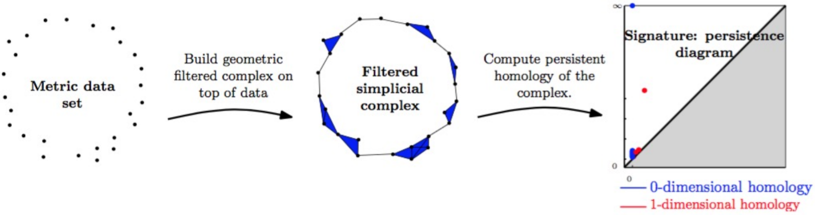
गुढी GUDHI

Geometric Understanding in Higher Dimensions

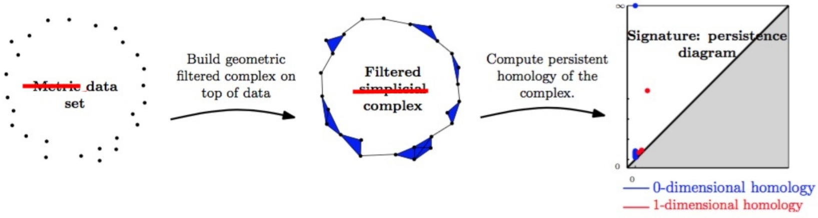
Inria

- GUDHI – project supported by ERC and hosted by INRIA.
- We aim in:
  - developing new data structures in computational topology and geometry and
  - developing associated statistical, geometrical and topological algorithms.

Standard computational topology pipeline.



Standard computational topology pipeline, redo 1.



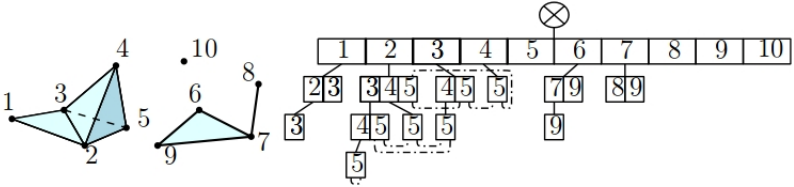
Extended computational topology pipeline.

- Input: Topological space / point cloud / grid–based data.
- Discretization: Simplicial complex (Rips, alpha, witness), cubical complex (all with or without filtration).
- Topological summary: (co)homology, persistent homology, zig-zag persistence.
- Later statistical and machine learning processing of topological information.
- All in C++ of Python (via cython).

The data structures

- General containers:
  - Simplex tree.
  - Simplex array list.
  - Simplicial complex via skeleton blockers.
  - Cubical complexes.
- Specific implementations:
  - Rips complexes
  - Alpha complexes.
  - Witness complexes.
  - Cubical complexes.

Simplicial complexes, simplex tree.



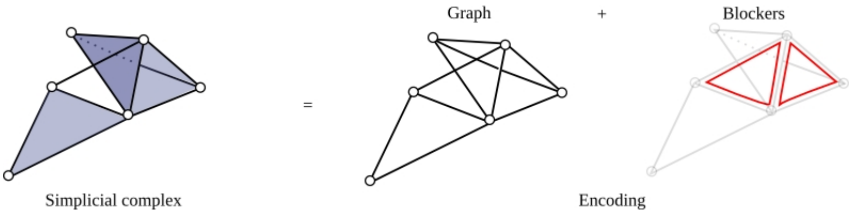
- Memory and time–efficient data structure to store simplicial complexes.
- Every simplex is a word stored in the tree.
- Nodes at each level corresponds to simplices of the same dimension.
- It is a base of all algorithms to compute persistence of weighted simplicial complexes in Gudhi.

by Clément Maria

Simpliex array list.

1. We store a list of 0 dimensional vertices.
2. And link them to the list of maximal/critical simplices.
3. The complexity of basic operations is comparable to the ones in Simplex Tree.
4. The space complexity of the structure is way superior to ST.
5. Most algorithms are currently being translated to SAL.

Simplicial complexes, skeleton blockers.



- A data structure for very large simplicial complexes.
- We store the $1-$skeleton and the minimal simplices which are not present in the complex.
- The rest is generated from cliques in the $1-$skeleton.
- Used in edge contraction toolbox (details later).

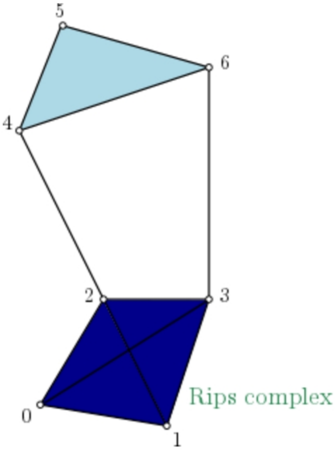
by David Salinas

Specific simplicial complexes.

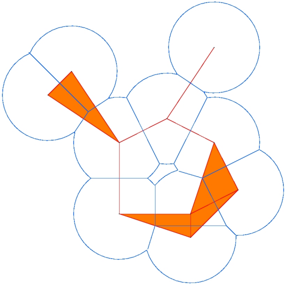
- So far we have seen the general containers to keep simplicial complexes.
- Now we will discuss specific simplicial complexes that can be stored in those containers.

The Rips complex.

- Rips complex from point cloud.
- Rips complex from distance / similarity matrix.



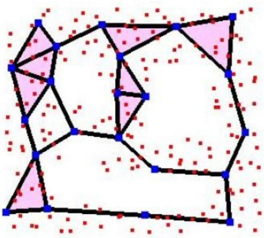
Weighted Alpha complexes.



- Constructed from point-clouds, based on CGAL data structures.
- Filtered alpha complexes in any dimension.
- Periodic alpha complexes in dimension 3.
- Weighted alpha complexes coming up soon.

by Vincent Rouvreau

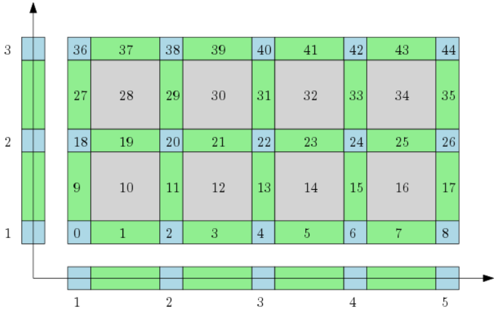
Witness complexes.



- For large point clouds, select small, representative collection of points L called landmarks.
- Build a complex on landmark points. Add a simplex if a witness exists.
- Version with and without filtration.

by Siargey Kachanovich

Cubical complexes, bitmaps.



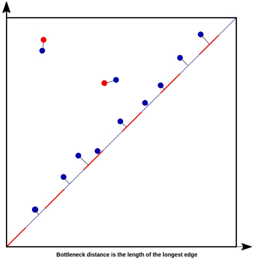
- Represented as a vector of filtration values.
- (Co)boundary computed based on the position in this vector.
- Used in analysis of grid-type data.

by Paweł Dłotko

(Persistent) (co)homology.

- Standard persistence cohomology computations by using compressed annotation matrix (by Clément Maria).
- Multi–field persistence (detection of torsion coefficients) (by Clément Maria).
- Computing persistence with Phat (Phat by Ulrich Bauer, Michael Kerber, Jan Reininghaus and Hubert Wagner) – coming in the next release.
- Zig-zag persistence (coming soon).

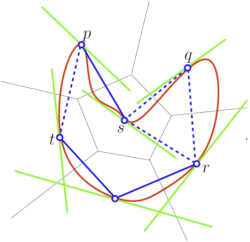
Standard metrics.



- Bottleneck distances.
- The geometric algorithm (Efart et al).
- At the moment we do not have p-th Wasserstein distances.
- Nice alternative Hera: `bitbucket.org/grey_narn/hera`.

by Francois Godi

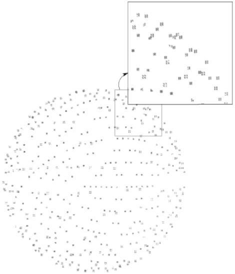
Manifold reconstruction with tangential complexes.



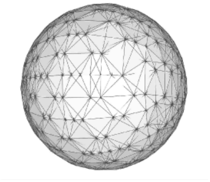
- Suppose we have a set of points sampled from a manifold.
- For every point construct tangent space at that every $p \in L$.
- For every $p \in L$, construct its star and glue the stars of neighbouring points if they agree.
- Based on Jean-Daniel Boissonnat and Arijit Ghosh *Manifold reconstruction using Tangential Delaunay Complexes*.

by Clément Jamin

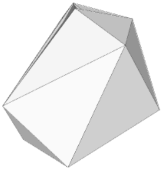
Simplicial complex simplification with edge contraction.



Point cloud sampling SO3
(points are in R^9 but projected into R^3
for vizualization)



Rips complex built uppon these points
20 millions simplices



Simplicial complex obtained after simplification
714 simplices

by David Salinas

Finally.

Let us have some goodies!

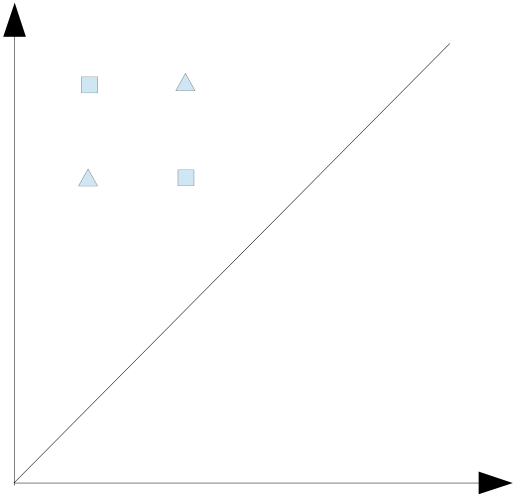
Diagrams, diagrams, what next?

1. So, we have this whole machinery to compute diagrams.
2. What shall we do next once have them?
3. How can we make a statistics or machine learning on diagrams?
4. Wait... did you said *diagrams*?

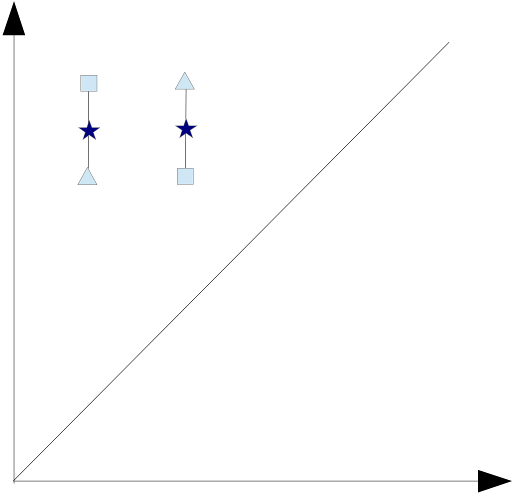
The Gudhi Stat

1. Statistical toolbox for Gudhi.
2. Pre-available as a set of executables.
3. Link standard operations in statistics and machine learning with persistence.
4. Uses various representations of persistence. Not only diagrams.

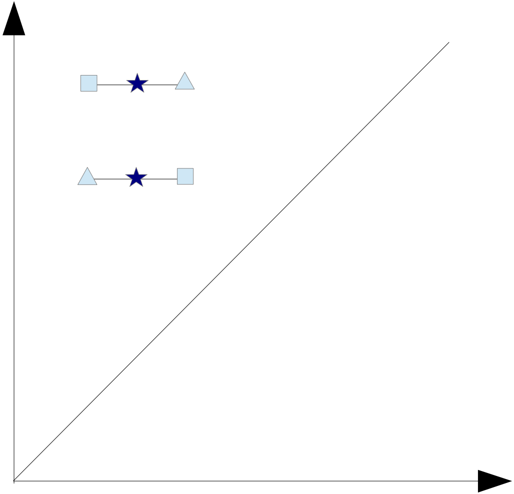
Why persistence diagrams are not sufficient?



Why persistence diagrams are not sufficient?



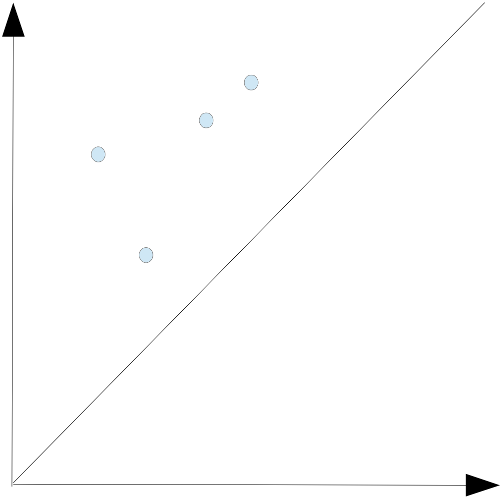
Why persistence diagrams are not sufficient?



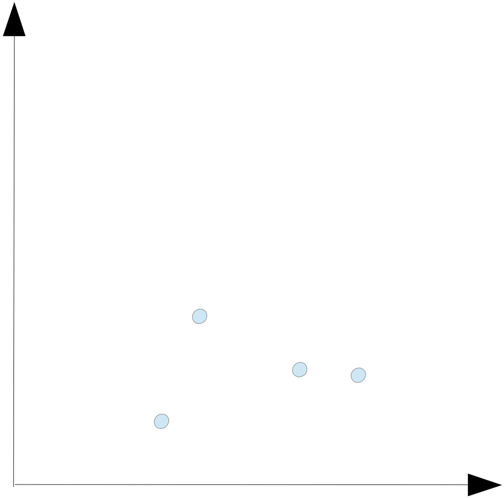
Persistence landscapes.

- Idea by Peter Bubenik.
- Lift persistence diagrams to Banach space of functions.
- This space is large enough to have well defined averages and scalar products.

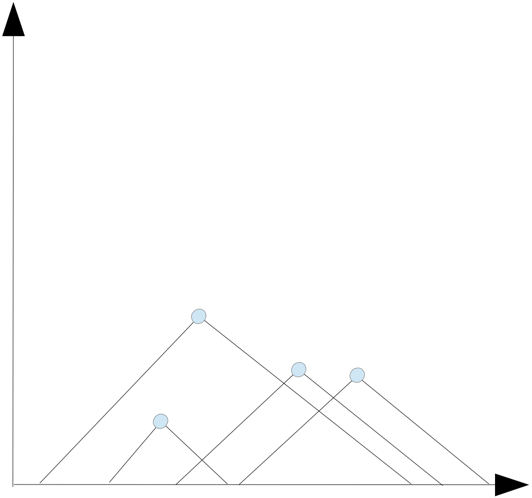
Persistence landscapes.



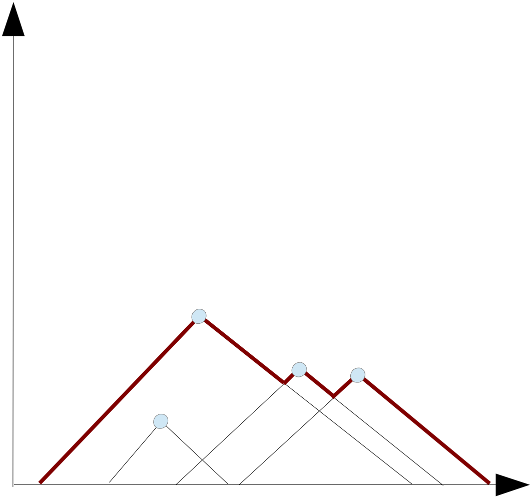
Persistence landscapes.



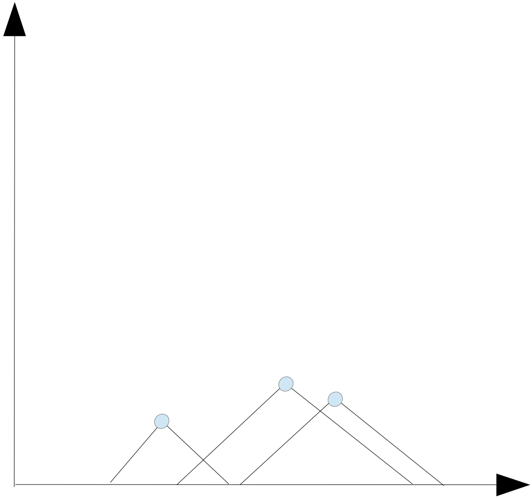
Persistence landscapes.



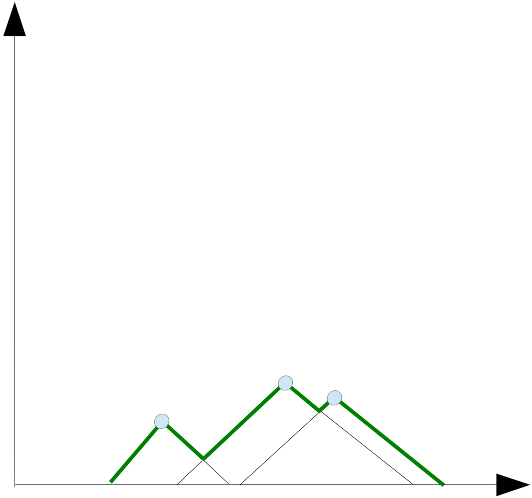
Persistence landscapes.



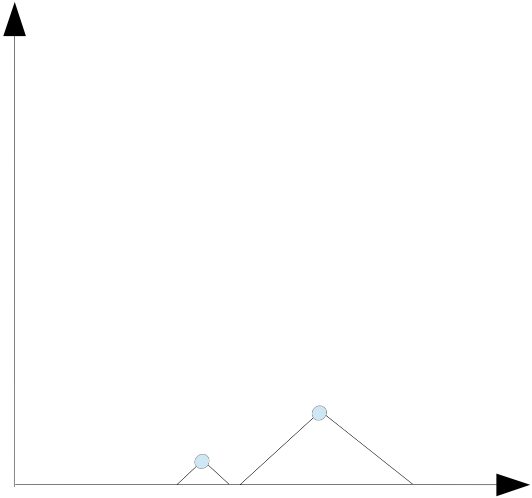
Persistence landscapes.



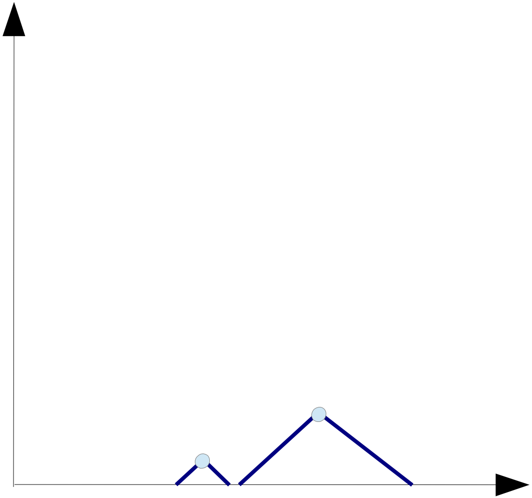
Persistence landscapes.



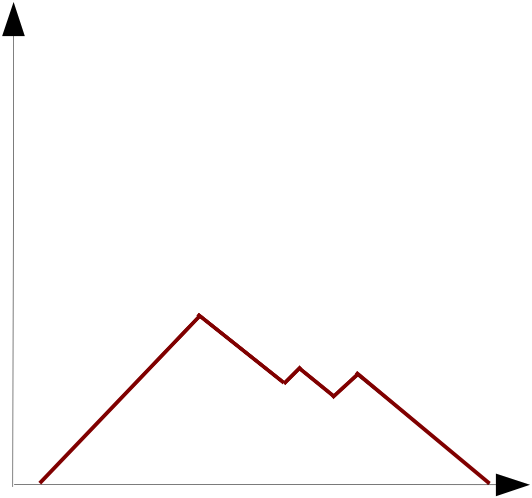
Persistence landscapes.



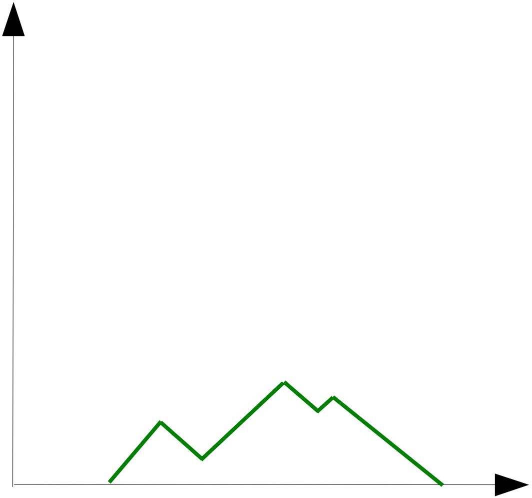
Persistence landscapes.



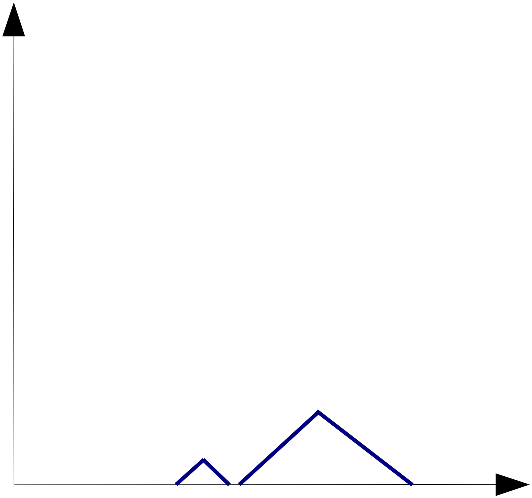
Persistence landscapes.



Persistence landscapes.



Persistence landscapes.



Persistence landscapes.

- Bottleneck stability.
- Averages.
- L^p distances.
- Scalar products.
- Various ways to vectorize.
- My "adventure" with persistence landscapes begin with persistence landscape toolbox, more than 3 years ago.

Persistence landscape toolbox.

- Computations of distance matrix.
- Computation of averages landscapes.
- Standard deviation.
- Computations of integrals.
- Moments computations.
- Permutation test.
- T-test, anova.
- Classifiers.
- Warning: this software is barely maintained at the moment.

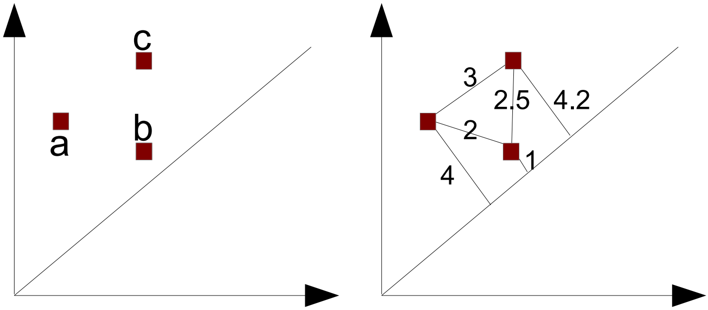
What do we need to do statistics?

- In almost all the cases, we used only a few property of the landscapes.
- And it was not important at all that we use landscapes.
- What I needed:
  - Distances.
  - Averages.
  - Scalar product.
  - Vectorization.
  - Confidence bounds (?)

Other representations of persistence.

- Persistence landscapes on a grid (simplified representation used in TDA R-package).
- Persistence vectors (by M. Cariere, S. Oudot and M. Ovsjanikov).
- Various types based on summing distributions centered at diagram points:
  - Persistence Weighted Gaussian Kernel by G. Kusano, K. Fukumizu, Y. Hiraoka.
  - Persistence Stable Space Kernel, by J. Reininghaus, U. Bauer, R. Kwitt.
  - Persistence Images by Chepushtanova, Emerson, Hanson, Kirby, Motta, Neville, Peterson, Shipman, Ziegelmeier.

Persistence vectors.

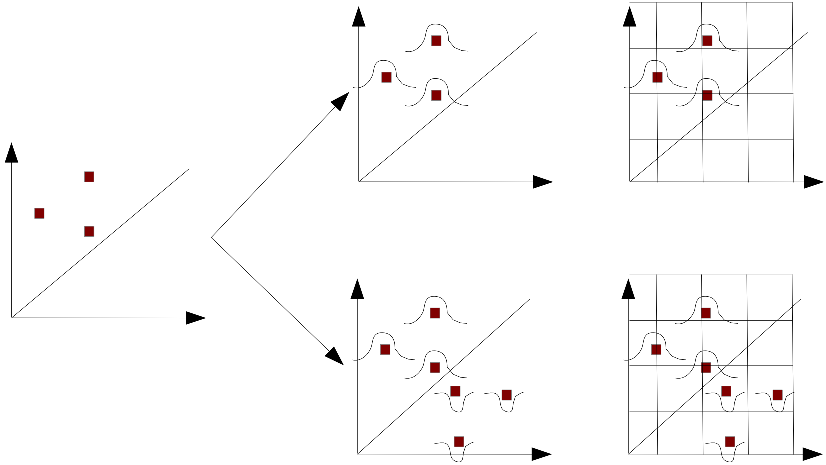


(3,1,1)

Persistence vectors, statistical operations.

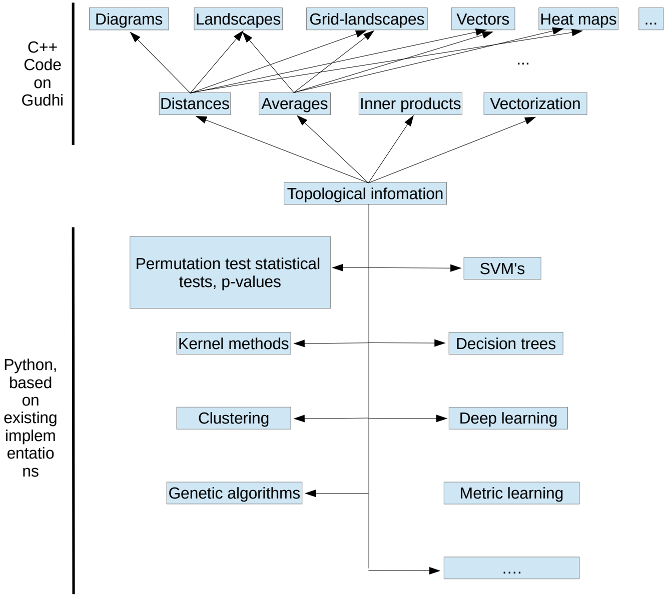
1. Point-wise averages.
2. Max, l^p distances.
3. Various projections to $\mathbb{R}$ are possible.
4. Scalar products of vectors well defined.
5. Vectorization is for free.
6. Confidence bounds (?).

Distributions on diagrams.



Distributions on diagrams.

1. Distances and averages.
2. W-1 stable.
3. Vectorization possible.
4. Real-valued function possible to define.
5. Confidence bounds (?).


C++ Code on Gudhi
Diagrams
Landscapes
Grid-landscapes
Vectors
Heat maps
...
...
Distances
Averages
Inner products
Vectorization
Topological infomation
Python, based on existing implementations
Permutation test statistical tests, p-values
SVM's
Kernel methods
Decision trees
Clustering
Deep learning
Genetic algorithms
Metric learning
....

Coming soon.

1. Bootstraps.
2. Subsampling.
3. Confidence bounds.
4. Extrapolation.
5. etc...

Some more goodies!

The code.

1. Python bindings (not all functionalities yet available).
2. Source code (boost, gmp, eigen, cgal, tbb and you are good to go).
3. Statically compiled sources (windows and osx), links on the wordpress workshop webpage.
4. Some exercises for you are also there!

Looking ahead.

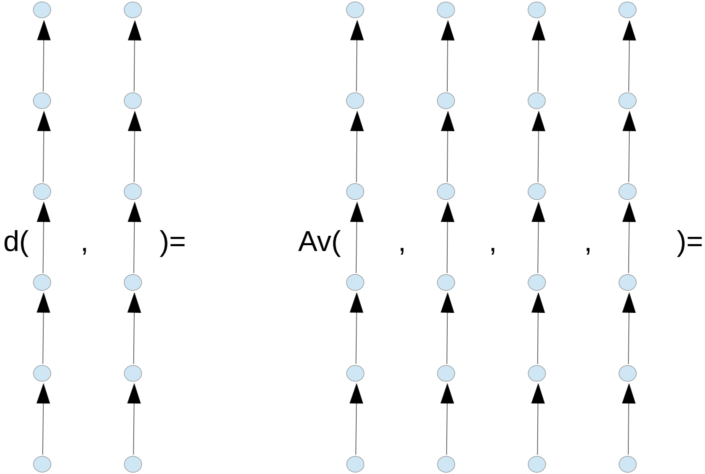
Time varying data.

- Quite often our data are time–varying.
- In each time step we are given a scalar value function.
- But filtration is changing (in a smooth way).
- Multi dimensional persistence not feasible at the moment.
- Methods for time varying data.
- Note that we cannot go back in time.

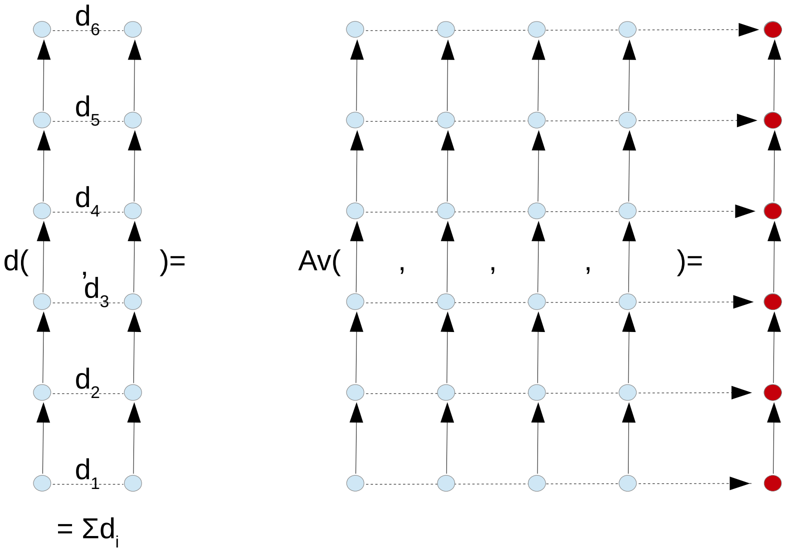
Time varying data.

- Suppose we know only the data from the constitutive time steps.
- We do not know how they were transformed to each other.

Distances and averages.



Distances and averages.



Spinodal decomposition in alloys.

50/50

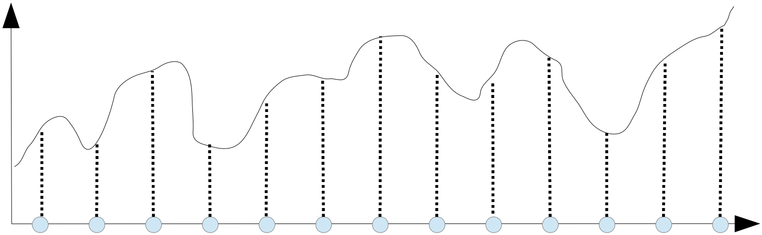
60/40

75/25

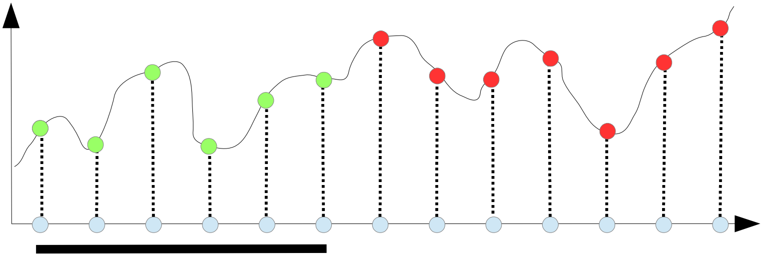
Topological signal processing.

- $f : \mathbb{R} \to \mathbb{R}$.
- Time series $(x_1, t_1), (x_2, t_2), \ldots, (x_n, t_n)$.
- Aim: Turn time series/ function into point cloud.

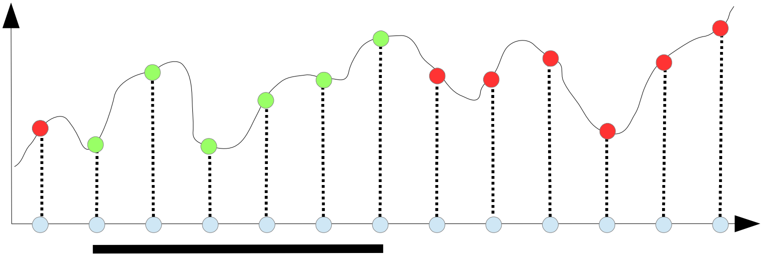
Topological signal processing.



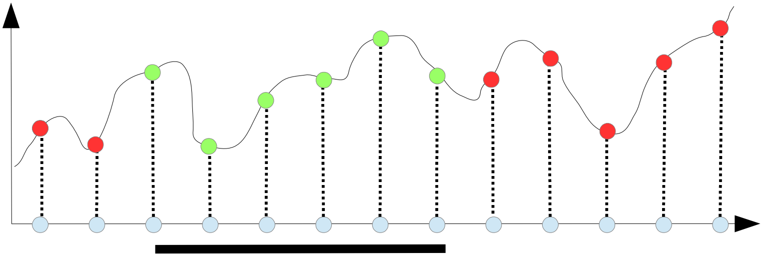
Topological signal processing.



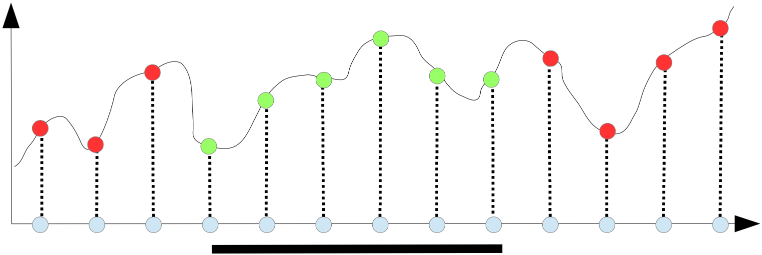
Topological signal processing.



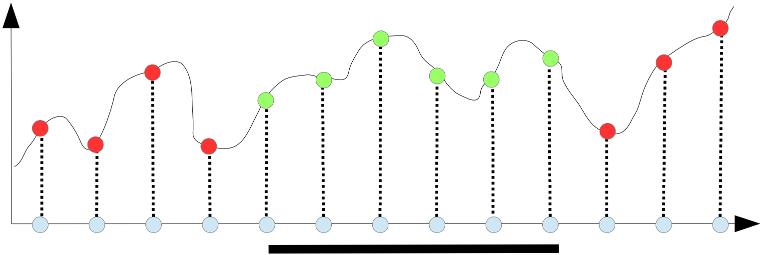
Topological signal processing.



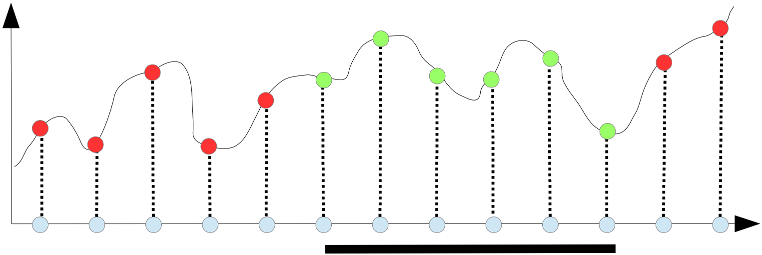
Topological signal processing.



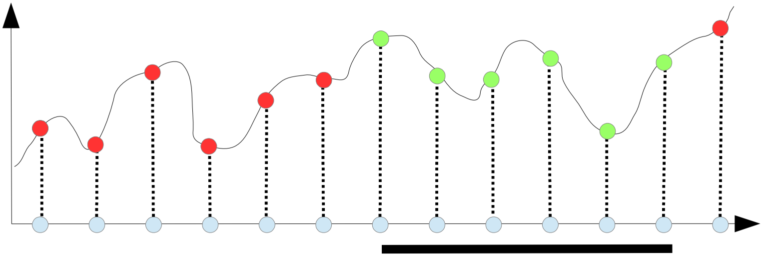
Topological signal processing.



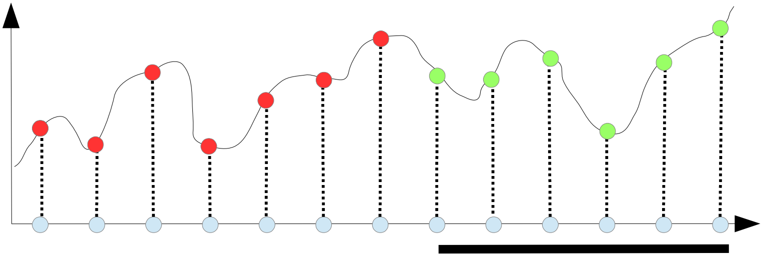
Topological signal processing.



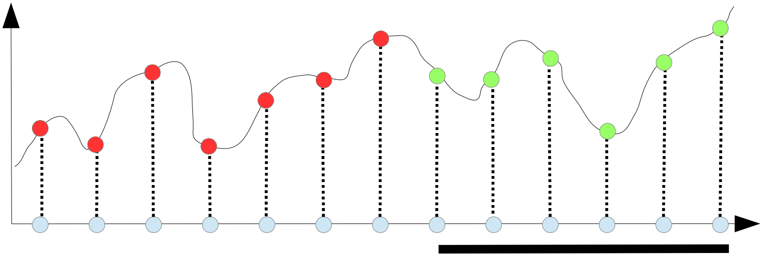
Topological signal processing.



Topological signal processing.



Topological signal processing.



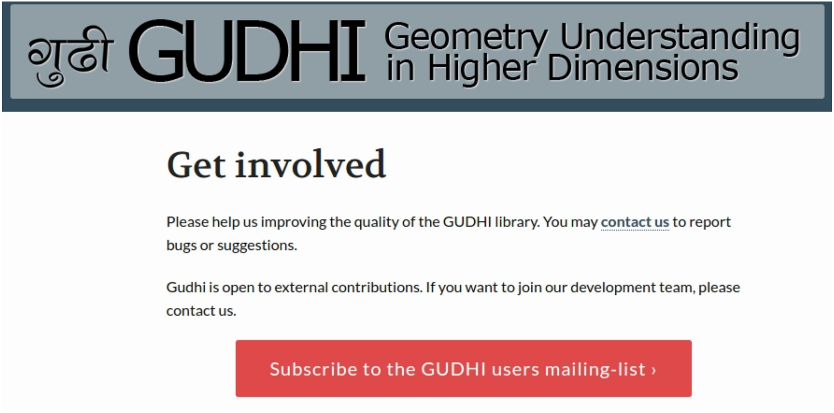
Sliding window embedding.

1. This toolbox will provide a collection of tests for periodicity/circularity of function or time series.
2. It will not provide guarantee unless restricted to specific types of functions.
3. It will allow comparison of time series by comparing their sliding window embeddings.

Current applications of Gudhi library.

- TDA package.
- Surface reconstruction with Tangential Complex.
- Mesh simplification by edge contractions.
- Zeolites.
- ...

Subscribe!



`http://gudhi.gforge.inria.fr/getinvolved/`

Thank you for your time!

Swansea University and DataShape Team, INRIA,
contact: pawel.dlotko, vincent.rouvreau @ inria.fr, p.t.dlotko @ swansea.ac.uk