Computing Topological Descriptors for Periodic Data

Herbert Edelsbrunner[1] Teresa Heiss[2] Jiaqi Zheng[3]

[1]*Institute of Science and Technology Austria* [2]*Australian National University* [3]*National University of Singapore*

Introduction. Periodic data frequently arises in materials science, in particular the study of crystalline materials. Modeling the atoms of a crystalline material as points leads to a *periodic set* (which we define as the Minkowski sum of a point set and a lattice), see Figure 1 for an example. In this poster, we introduce *Periodica* – a C++ based Python library for analyzing the topological structures of a periodic set. It can compute how the points connect with each other (captured by the *Delaunay triangulation*) at different length scales (encoded in the *periodic merge tree* and the *topological descriptors*).

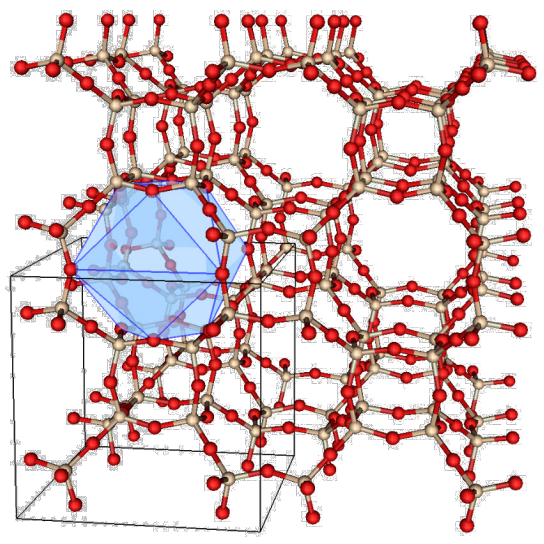

Figure 1. Example of periodic data*

*Taken from *https://doi.org/10.1021/acs.jpcc.0c01167*

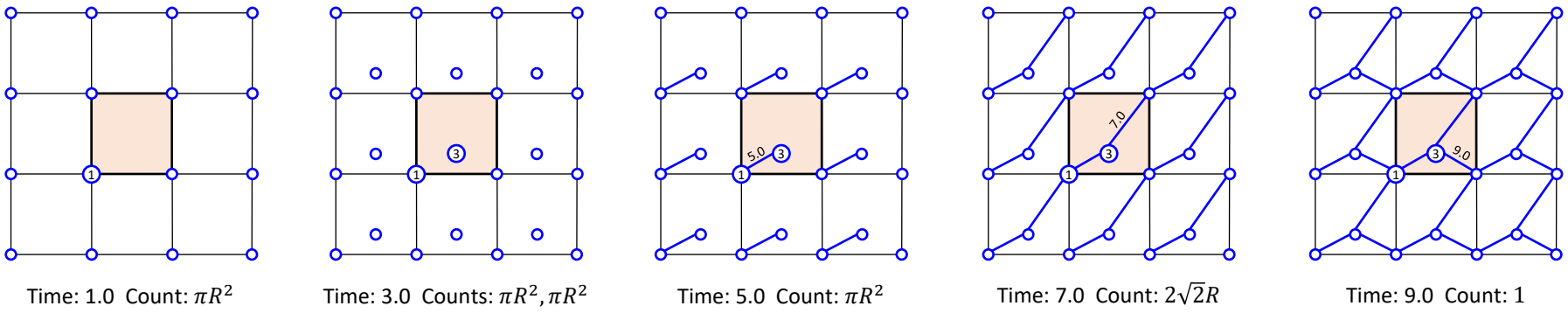

Figure 2. Example of a periodic filtration

Periodic merge tree. The periodic points and the gaps between them imply a *filtration* (a sequence of increasing complexes or spaces); see Figure 2 for an example. In the periodic filtration, the connected components merge as the time increases. The periodic merge tree keeps track how they merge and counts the periodic copies of each component within a spherical window of radius R, see Figure 3. The periodic filtration can be represented by a finite structure called *quotient complex*, which can serve as a direct input to our Periodica library.

Topological descriptors. Our library provides several options for output. These include: 1. The *periodic merge tree* as introduced in the last paragraph. 2. *Persistence barcodes (resp. diagram)*, which represent the topological features as horizontal bars (resp. points in the plane) indicating their birth and death times, where each bar (resp. point) has a multiplicity that reflects the density of the feature in the space. 3. *Persistence image*, a vectorized version of the persistence diagram that is compatible with machine learning applications. See Figure 4 for Periodica's outputs for the small toy example from Figures 2 and 3.

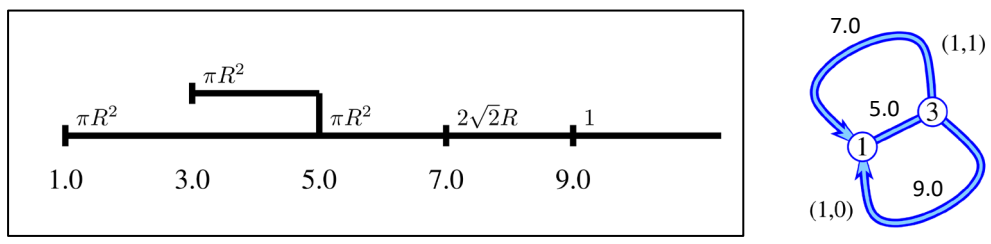

Figure 3. Periodic merge tree and quotient complex

We will continue to improve the library and would appreciate any suggestions and feedback.

Which feature would YOU want us to add?

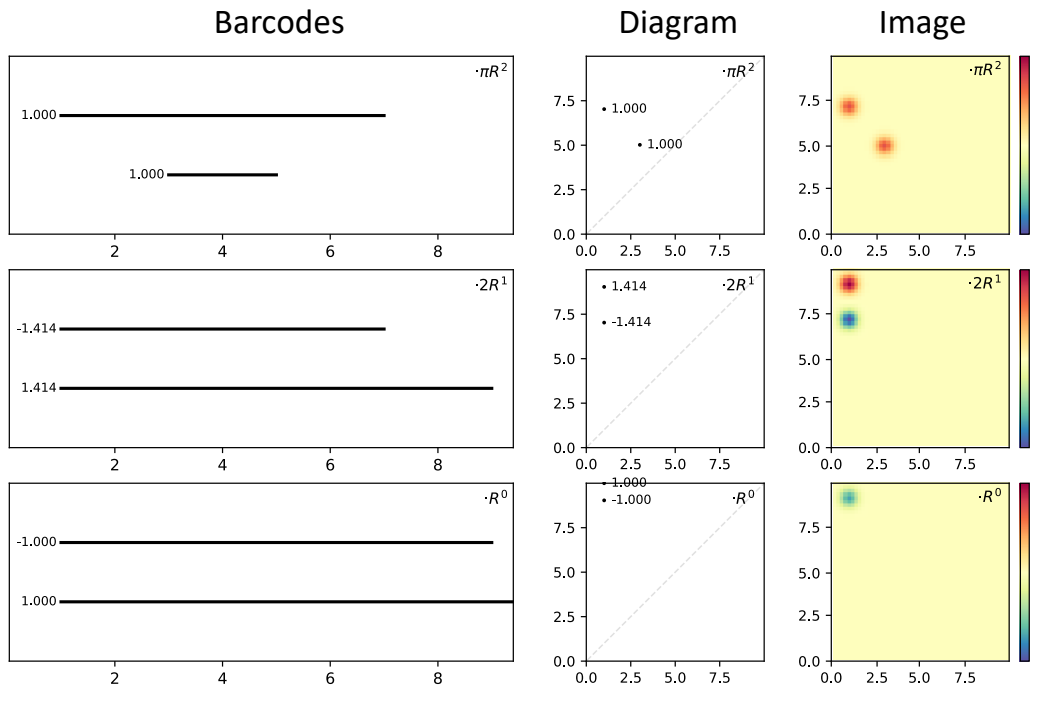

Figure 4. Topological descriptors

Library available on GitHub: *https://github.com/orzzzjq/periodica*

You can contact the developer via email: *jiaqi@u.nus.edu*