



# Persistable: persistent and stable clustering

Luis Scoccola <sup>1</sup> and Alexander Rolle<sup>2</sup>

1 Northeastern University 2 Technical University of Munich

DOI: [10.21105/joss.05022](https://doi.org/10.21105/joss.05022)

## Software

- [Review](#) 
- [Repository](#) 
- [Archive](#) 

---

Editor: [Rachel Kurchin](#)  

## Reviewers:

- [@lmcinnes](#)
- [@AP6YC](#)

Submitted: 18 November 2022

Published: 08 March 2023

## License

Authors of papers retain copyright and release the work under a Creative Commons Attribution 4.0 International License ([CC BY 4.0](#)).

## Summary

Persistable is an implementation of density-based clustering algorithms intended for exploratory data analysis. What distinguishes Persistable from other clustering software is its visualization capabilities. Persistable's interactive mode lets the user visualize multi-scale and multi-density cluster structure present in the data. This is used to guide the choice of parameters that lead to the final clustering.

Persistable is based on multi-parameter persistence ([Botnan & Lesnick, 2022](#)), a method from topological data analysis; the theory behind Persistable is developed in ([Rolle & Scoccola, 2020](#)). Persistable is implemented in Python, with the most expensive computations—in particular the implementations borrowed from ([Virtanen et al., 2020](#)) and from the high-performance algorithms for density-based clustering developed in ([McInnes & Healy, 2017](#)) and implemented in ([McInnes et al., 2017](#))—done in Cython. Persistable's interactive mode is inspired by RIVET ([The RIVET Developers, 2020](#)) and is implemented in Plotly Dash ([Plotly Technologies Inc., 2015](#)). We test the core algorithms as well as the graphical user interface in Ubuntu, macOS, and Windows. We have designed Persistable with the goal of making both its computational components and its GUI easy to extend. We hope to keep adding high performance topological inference functionality.

The documentation for Persistable can be found at [persistable.readthedocs.io](https://persistable.readthedocs.io).

## Related work and statement of need

There exist many implementations of density-based clustering algorithms. Several of these are related to and serve some of the same purposes as Persistable. The main novel contribution of Persistable is that of providing visualization tools based on persistence which inform the selection of all parameters. We review existing algorithms and implementations that are related to Persistable, and briefly describe Persistable's main functionality.

**DBSCAN.** The algorithm was introduced in ([Ester et al., 1996](#)), and an implementation is available at ([Pedregosa et al., 2011](#)). DBSCAN takes two parameters, a scale parameter and a density threshold, which are used to construct a graph on the data. This graph models the connectivity properties of the data with respect to the chosen parameters, and the DBSCAN clustering is the set of components of this graph. A main advantage of DBSCAN is that the output clustering is very interpretable in terms of the chosen parameters; meanwhile, a major difficulty for practitioners is the choice of these parameters, especially without a priori knowledge of the scale of the data (see, e.g., ([Schubert et al., 2017](#))).

**HDBSCAN.** The algorithm was introduced in ([Campello et al., 2013](#)), and a high-performance implementation is in ([McInnes et al., 2017](#)). HDBSCAN can be seen as a hierarchical version of DBSCAN that eliminates the dependence on the scale parameter by considering the hierarchical clustering defined by fixing a density threshold and letting the distance scale vary. The algorithm extracts a single clustering from this hierarchy according to a certain notion of persistence, using an additional minimum cluster size parameter. Eliminating the scale parameter makes

HDBSCAN easier to use than DBSCAN in many cases, but choosing the density threshold can still be a challenge for practitioners. The implementation of (McInnes et al., 2017) has visualization functionality which aides in the choice of minimum cluster size.

**ToMATo.** The algorithm was introduced in (Chazal et al., 2013) and is implemented in the GUDHI library (The GUDHI Project, 2015). This was the first clustering algorithm to use persistence to do parameter selection, specifically by using a persistence diagram to choose the number of clusters. The basic input for ToMATo is a graph and a density estimate. How these are chosen is left to the user, and their choice is not informed by persistence.

**RIVET.** The theory behind RIVET is developed in (Lesnick & Wright, 2015) and (Lesnick & Wright, 2022), and the official implementation is in (The RIVET Developers, 2020). This is a very general tool for 2-parameter persistent homology, and can, in particular, produce visualizations that are very similar to the ones produced by Persistable. Indeed, we have taken RIVET as inspiration when designing Persistable. However, RIVET is not primarily intended to be clustering software, and it does not produce a clustering of the data.

**Persistable.** We contribute to this landscape of density-based clustering methods by providing a clustering pipeline in which every choice of parameter is guided by visualization tools. Persistable first builds a hierarchical clustering of data, in a way that is similar to HDBSCAN: we take a one-parameter family of DBSCAN\* graphs<sup>1</sup> by choosing a line through the DBSCAN\* parameter space. This choice is guided by a *component counting function* plot, which is a summary of the output of DBSCAN\* for a large set of parameters. We then extract a single clustering using persistence, similar to ToMATo. This choice is guided by a *prominence vineyard*.

## Example

We use a synthetic dataset from (McInnes & Healy, 2017) and (McInnes et al., 2017), which is challenging for most clustering algorithms but easy to visualize. As shown in Figure 1, Persistable's visualizations suggest that there are six clusters that persist across several distance scales as well as several density thresholds. We show in Figure 2 the flat clustering obtained by selecting those six high-prominence clusters.

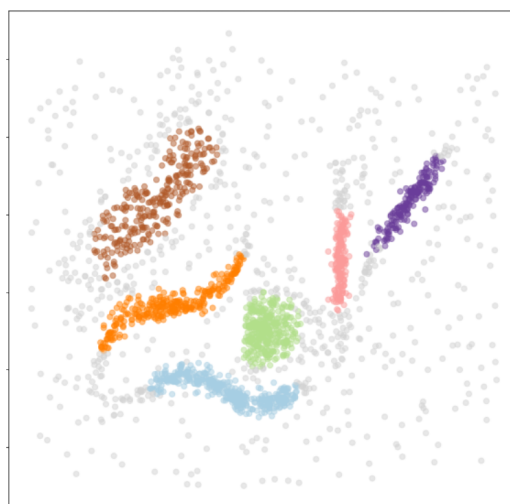
It is worth pointing out that the component counting function of Figure 1 summarizes 10,000 runs of DBSCAN\* on a dataset with a bit more than 2000 points, and takes about 2 seconds to run on a MacBook Pro with M1 chip and a 10-Core CPU.

---

<sup>1</sup>DBSCAN\* is a slight modification of DBSCAN introduced in (Campello et al., 2013), which removes the notion of border point, simplifying the theory and implementation without compromising efficacy.



**Figure 1:** Using Persistable's GUI to select parameters. We look at a family of 100 one-parameter hierarchical clustering that are restrictions of the two-parameter hierarchical clustering obtained by running DBSCAN\* with all possible parameters.



**Figure 2:** Output clustering obtained with the parameters of Figure 1. Grey points do not belong to any cluster.

## Acknowledgements

We thank Leandro Lovisolo and Manuel Ferreria for several fruitful conversations. L.S. was partially supported by the National Science Foundation through grants CCF-2006661 and CAREER award DMS-1943758.

## References

- Botnan, M. B., & Lesnick, M. (2022). An introduction to multiparameter persistence. *Proceedings of the 2020 International Conference on Representations of Algebras, (in Press)*.

- Campello, R. J., Moulavi, D., & Sander, J. (2013). Density-based clustering based on hierarchical density estimates. *Pacific-Asia Conference on Knowledge Discovery and Data Mining*, 160–172. [https://doi.org/10.1007/978-3-642-37456-2\\_14](https://doi.org/10.1007/978-3-642-37456-2_14)
- Chazal, F., Guibas, L. J., Oudot, S. Y., & Skraba, P. (2013). Persistence-based clustering in Riemannian manifolds. *J. ACM*, 60(6), Art. 41, 38. <https://doi.org/10.1145/2535927>
- Ester, M., Kriegel, H.-P., Sander, J., & Xu, X. (1996). A density-based algorithm for discovering clusters in large spatial databases with noise. *Proceedings of the Second International Conference on Knowledge Discovery and Data Mining*, 226–231.
- Lesnick, M., & Wright, M. (2015). *Interactive visualization of 2-d persistence modules*. (under review). <https://doi.org/10.48550/ARXIV.1512.00180>
- Lesnick, M., & Wright, M. (2022). Computing minimal presentations and bigraded Betti numbers of 2-parameter persistent homology. *SIAM J. Appl. Algebra Geom.*, 6(2), 267–298. <https://doi.org/10.1137/20M1388425>
- McInnes, L., & Healy, J. (2017). Accelerated hierarchical density based clustering. *2017 IEEE International Conference on Data Mining Workshops (ICDMW)*, 33–42. <https://doi.org/10.1109/ICDMW.2017.12>
- McInnes, L., Healy, J., & Astels, S. (2017). Hdbscan: Hierarchical density based clustering. *Journal of Open Source Software*, 2(11), 205. <https://doi.org/10.21105/joss.00205>
- Pedregosa, F., Varoquaux, G., Gramfort, A., Michel, V., Thirion, B., Grisel, O., Blondel, M., Prettenhofer, P., Weiss, R., Dubourg, V., Vanderplas, J., Passos, A., Cournapeau, D., Brucher, M., Perrot, M., & Duchesnay, E. (2011). Scikit-learn: Machine learning in Python. *Journal of Machine Learning Research*, 12, 2825–2830.
- Plotly Technologies Inc. (2015). *Collaborative data science*. Plotly Technologies Inc. <https://plot.ly>
- Rolle, A., & Scoccola, L. (2020). Stable and consistent density-based clustering. (*Under Review*).
- Schubert, E., Sander, J., Ester, M., Kriegel, H. P., & Xu, X. (2017). DBSCAN revisited, revisited: Why and how you should (still) use DBSCAN. *ACM Trans. Database Syst.*, 42(3). <https://doi.org/10.1145/3068335>
- The GUDHI Project. (2015). *GUDHI user and reference manual*. GUDHI Editorial Board. <http://gudhi.gforge.inria.fr/doc/latest/>
- The RIVET Developers. (2020). *RIVET* (Version 1.1.0). <https://github.com/rivetTDA/rivet/>
- Virtanen, P., Gommers, R., Oliphant, T. E., Haberland, M., Reddy, T., Cournapeau, D., Burovski, E., Peterson, P., Weckesser, W., Bright, J., van der Walt, S. J., Brett, M., Wilson, J., Millman, K. J., Mayorov, N., Nelson, A. R. J., Jones, E., Kern, R., Larson, E., ... SciPy 1.0 Contributors. (2020). SciPy 1.0: Fundamental Algorithms for Scientific Computing in Python. *Nature Methods*, 17, 261–272. <https://doi.org/10.1038/s41592-019-0686-2>