Advanced Data Analytics Project Proposal: Persistent Homology

Matthew LeBar

October 15, 2025

Introduction

My project will be on persistent homology, a technique in Topological Data Analaysis (TDA). As the name would suggest, TDA seeks to use concepts and methods from topology for the analysis of real-world datasets by detecting underlying structure and shape in datasets. TDA has found a tremendous range of applications, including biology [1], manufacturing [8], molecular science [9], and finance [5], among many others. Persistent homology is considered to be one of the most important approaches in TDA [7], and its development can be seen as the origin of TDA [2]. Thus it is a natural place to start in building my mastery of TDA.

Project Plan

My project will take place in two phases. The first, which I aim to complete by week 7 of the quarter, is familiarizing myself with the mathematical theory behind persistent homology. I will use Edelsbrunner and Harer's *Computational Topology: An Introduction* [3], and plan to read the majority of chapters 1, 3, 4, and 7 in order to understand persistent homology. I have a significant amount of familiarity with background topics (i.e. graph theory, simplical complexes, group theory, topology), so even though this is a large amount of material to cover, I anticipate that I can get through it at an appropriate level of detail in the timeline suggested. My midterm presentation will be a high-level overview of this material.

In the second phase, I aim to write an algorithm using NumPy that can compute persistent homology on a given dataset. To do so, I will be coding the algorithm presented in "Computing Persistent Homology" [10], the paper which originally introduced the technique. My understanding is that this algorithm takes place in three phases, so I will devote one week each of weeks 7, 8, and 9 to each phase. The first phase is construction of simplical complexes from the dataset, the second the construction of boundary matrices from the simplical complexes, and the third the reduction of the boundary matrices. My final presentation would then explain the algorithm at a high level and demonstrate that my code correctly executes the algorithm.

Project Evaluation

I plan to develop and test the code using artificial datasets I develop (e.g., data generated noisily in the shape of a torus), but if I have time it would be exciting to test my code on some open-source databases that TDA is known to work on, including MNIST [4] or RCBS, a database for protein structures [6]. Producing my own artificial dataset will allow me to control the shape of the underlying dataset, so I can verify that the persistent homology algorithm is working correctly. To make this even easier, I can compare against some open-source packages for computing persistent homology, like Ripser or Gudhi. Giving more specifics on the validation process will require a deeper understanding of the underlying mathematical theory than I currently have (since the outputs of the algorithm are mathematical objects that I am not familiar with), but I will include a detailed explanation in my final report.

Project Flexibility

This is a highly ambitious project, and while I believe it is doable in the time I have, I've also tried to construct a timeline that makes it easy to adjust the final goal should I find myself pressed for time. Firstly, I can truncate my research into the underlying mathematics - although it is interesting and significant for me, I imagine that there will be details that are not necessary for me to understand in order to code the overall algorithm. Secondly, if I find the underlying mathematical background sufficiently dense and interesting in its own regard, I can write a report explaining that background in a *very* high amount of detail, rather than trying to code the algorithm. Finally, I can stop partway through the persistent homology algorithm - if I were, say, only to complete the construction of boundary matrices, that combined with a strong report on persistent homology would be a fairly substantial project. All of these offer quite a bit of flexibility in project approach.

References

[1] Erik Amézquita, Michelle Quigley, Tim A. E. Ophelders, Erik Munch, and D. J. Chitwood. The shape of things to come: Topological data analysis and biology, from molecules to organisms. *Developmental Dynamics*, 249:816–833, 2020.

[2] Frédéric Chazal and Bertrand Michel. An introduction to topological data analysis: fundamental and practical aspects for data scientists, 2021.

[3] Herbert Edelsbrunner and John Harer. *Computational Topology - an Introduction.* American Mathematical Society, 2010.

[4] Adélie Garin and Guillaume Tauzin. A topological ”reading” lesson: Classification of mnist using tda, 2019.

[5] Marian Gidea and Yuri Katz. Topological data analysis of financial time series: Landscapes of crashes. *Physica A: Statistical Mechanics and its Applications*, 491:820–834, February 2018.

[6] Christian D Madsen, Agnese Barbensi, Stephen Y Zhang, Lucy Ham, Alessia David, Douglas E V Pires, and Michael P H Stumpf. The topological properties of the protein universe. *Nature Communications*, 16(7503), 2025.

[7] Zhe Su, Xiang Liu, Layal Bou Hamdan, Vasileios Maroulas, Jie Wu, Gunnar Carlsson, and Guo-Wei Wei. Topological data analysis and topological deep learning beyond persistent homology – a review, 2025.

[8] Martin Uray, Barbara Giunti, Michael Kerber, and Stefan Huber. Topological data analysis in smart manufacturing: State of the art and future directions. *Journal of Manufacturing Systems*, 76:75–91, October 2024.

[9] JunJie Wee and Jian Jiang. A review of topological data analysis and topological deep learning in molecular sciences, 2025.

[10] Afra Zomorodian and Gunnar Carlsson. Computing persistent homology. In *Proceedings of the Twentieth Annual Symposium on Computational Geometry*, SCG ’04, page 347–356, New York, NY, USA, 2004. Association for Computing Machinery.