Pomona College

Senior Thesis in Mathematics

# Victor's Senior Thesis


*Author:*
Victor de Fontnouvelle

*Advisor:*
Dr. Vin De Silva


Submitted to Pomona College in Partial Fulfillment
of the Degree of Bachelor of Arts

May 8, 2019

**Abstract**

We will explore various methods of analyzing high-dimensional data. We'll investigate techniques that make inferences about the underlying structure of the data, cluster the data, or reduce the dimension of the data. We'll apply these methods to real-world datasets, compare the methods, and suggest improvements to the methods.

# Contents

# Chapter 1

# Introduction

## 1.1 Inference About the Structure of the Data

Cohomology analysis and Helmholtz decomposition provide insight on the structure of the data. Cohomology analysis provides a broader framework for detecting clusters, holes, and higher-order features within the data. Helmholtz decomposition is a specific application of cohomology analysis which seeks to find an ordered list which best accounts for a weighted graph.

## 1.2 Clustering

Mapper clusters the data, by mapping clusters of points onto intervals on the real line using a filter function, and connecting overlapping clusters.

## 1.3 Dimensionality Reduction

Calculating the eigenvalues of the laplacian matrix reduces the dimension of the data. The laplacian is a symmetric matrix encoding the pairwise distances between points, normalized by row. This matrix can be thought of as a linear operator encoding heat flows. Given an input of initial temperatures, it outputs the changes in temperatures after one time step. Eigenvectors corresponding to low eigenvalues thus correspond to stable temperature configurations. The eigenvectors are perpendicular, and thus eigenvectors corresponding to slightly higher eigenvalues often capture geometric structure existing in the data. Additionally, nearby points will have similar values in the eigenvectors, and thus the eigenvectors are also a useful tool for reducing the dimension of the data.

# Chapter 2

# Review of Literature

Papers by Gunnar Carlsson [1] and Vin de Silva [3] explain the intuition behind cohomolgy analysis, and provides several examples. Carlsson [2] and deSilva [3] also describe the algorithm used to compute cohomology, which will be useful if I choose to implement it.

Curto [4] and Ulmer [5] both provide various examples of the uses of homology analysis, both in detecting underlying structure, and in providing fingerprints that identify different phenomena. Curto analyzes a dataset representing connection strengths between neurons in rats responsible for spatial recognition. Curto first keeps a certain proportion of edges $\rho$ s.t. $0\langle\rho\langle 1$, keeping those edges which are the strongest. Curto then runs cohomology analysis on this graph to determine the Betti curve. Curto determined that the Betti curve obtained from spatially organized neurons is different than the Betti curve that would be obtained from neurons with random structure. Cohomolgy analysis thus provides a method for detecting spatial neurons. Ulmer uses cohomology analysis to evaluate two different models of social interaction for aphids roaming in a dish. Standard measures that compare the models to real data include angular momentum, and average distance to closest neighbor. Ulmer found that cohomology analysis provided an equally strong measurement for assessing model accuracy.

A paper by Jiang [6] explains the Helmholtz decomposition that converts ranked data into ordinal data. It provides both the algorithm for the decomposition, as well as three examples of its use. Jiang uses Helmholtz decomposition to rank movies. Most users rate several movies, so each time a user rated two movies, this introduced an edge from one movie to the other indicating the user's preference. Jiang used Helmholtz decomposition to create an absolute index of currency values based off trading rates. Jiang also used Helmholtz decomposition to rank websites based off of connection strengths. Jiang found that Helmholtz decomposition performed as well as some of the standard methods for website ranking, indicating that it is a useful tool.

Carlsson [1] also explains the Mapper tool, and provides examples of its use. I have already used the algorithm described in this paper to analyze several datasets.

Singh [7] describes the intuition behind the laplacian analysis, which I have also implemented.

# Chapter 3

# Helmholtz Decomposition

## 3.1 Explanation of Helmholtz Decomposition

### 3.1.1 Motivation

Suppose we have three tennis players: Anne, Bob, and Carl. Let us represent the matches they play by a matrix $\bar{Y}$. Then

$$\bar{Y} = \begin{bmatrix} 0 & -2 & -3 \\ 2 & 0 & -4 \\ 3 & 4 & 0 \end{bmatrix}$$

means that Anne beat Bob by 2, Anne beat Carl by 3, and Bob beat Carl by 4. Suppose further that we have a second matrix $W$ which represents the number of matches played. Then

$$W = \begin{bmatrix} 0 & 1 & 2 \\ 1 & 0 & 3 \\ 2 & 3 & 0 \end{bmatrix}$$

Means that Anne and Bob played 1 match, Anne and Carl played 2 matches, and Bob and Carl played 3 matches. We would like to assign a numerical rating to all three players that best accounts for their respective match scores. For example, if Anne, Bob, and Carl receive respective ratings of 5, 3, and 2, then this would perfectly account for the matches (Anne, Bob) and (Anne, Carl), but would not account for the match (Bob, Carl).

More generally, we consider any skew-symmetric matrix $\bar{Y} \in M_n$ (recall that $A \in M_n$ is skew-symmetric when $A^T = -A$) along with a symmetric matrix $W$. We would like to find a vector $s \in M_{nx1}$ that minimizes $||\text{grad}s - \bar{Y}||_{2,W}$. Note that the norm is computed with respect to the inner product space defined by $W$. More precisely, $\langle X, Y \rangle_W = \Sigma_{\{i,j\} \in E} w_{ij} X_{ij} Y_{ij}$.

### 3.1.2 Intuition

The Hodge Decomposition Theorem provides that $\bar{Y}$ can be expressed as the sum of three orthogonal components:

1. The gradient flow $G$
2. The curl flow $C$
3. The harmonic flow $H$

The gradient flow $G$ represents a comparison matrix that corresponds directly to a vector $v$ where each item is assigned a real value. $G$ is the gradient of $v$, thus each entry is computed as $G[ij] = v[j] - v[i]$.

The harmonic flow $H$ represents a ranking that is curl-free and divergence-free. This means that any three items in $H$ will have pairwise rankings that are logically consistent. Specifically, $H[ij] + H[jk] + H[ki] = 0, \forall i, j, k$. Because $\text{div}(H) = 0$, $H$ contains plausible values in the sense that it could have been produced by real-world data. A harmonic flow thus indicates that there are cycles in the graph with more than three edges, where the edge weights don't sum to zero.

The curl flow $C$ is the image of curl$^*$, the adjoint of the curl. Nonzero values of C thus indicate cycles of length three whose edge weights don't sum to zero.

The gradient flow provides the ranking that minimizes the least-squared residual, while the harmonic and curl flows characterize the residual. Specifically, $\bar{Y} - H = G + C$. A large curl flow indicates that the ordering of items ranked closely together is unreliable, while a large harmonic flow indicates that the ordering of items ranked further apart is unreliable. For example, if the curl flow is large but the harmonic flow is small, this indicates that the ranking is valid at a larger scare, but that the specific ordering of closely-ranked alternatives isn't very precise.

### 3.1.3 Representation of Matrices

It is necessary to have matrix representations of $\delta_0$ (grad), $\delta_1$ (curl), $\delta_0^*$, and $delta_1^*$ in order to compute the gradient, curl, and harmonic flows. Throughout this section, $f$, $\alpha$, and $A$ refer to maps from edges, vertices, and triples into $\mathbb{R}$, respectively.

The gradient operator assigns values to edges based on the difference of the values of the endpoints. Specifically, $\alpha((a,b)) = -f(a) + f(b)$. For example, when the graph has four vertices and all edge weights are nonzero, the matrix representation of the gradient is

$$\begin{bmatrix} -1 & 1 & 0 & 0 \\ -1 & 0 & 1 & 0 \\ -1 & 0 & 0 & 1 \\ 0 & -1 & 1 & 0 \\ 0 & -1 & 0 & 1 \\ 0 & 0 & -1 & 1 \end{bmatrix} \begin{bmatrix} a \\ b \\ c \\ d \end{bmatrix} = \begin{bmatrix} ab \\ ac \\ ac \\ bc \\ bd \\ cd \end{bmatrix}$$

We can verify the correctness of this matrix by examination. For example, the first row corresponds to the edge $(a,b)$ and thus we want $\alpha((a,b)) = -f(a) + f(b)$. Indeed, the value $-1$ in the first column corresponds to $a$, and the value $+1$ in the second column corresponds to $b$. If any of the edge weights are zero, we remove the corresponding row from the gradient matrix. For example, if $W((a,b)) = 0$, we would remove the first row from the matrix above.

The curl operator assigns values to triples based off the values of edges. Specifically, $A((a,b,c)) = \alpha(b,c) - \alpha(a,c) + \alpha(a,b)$. For example, when the graph has four vertices and all edge weights are nonzero, the matrix representation of the curl is:

$$\begin{bmatrix} 1 & -1 & 0 & 1 & 0 & 0 \\ 1 & 0 & -1 & 0 & 1 & 0 \\ 0 & 1 & -1 & 0 & 0 & 1 \\ 0 & 0 & 0 & 1 & -1 & 1 \end{bmatrix} \begin{bmatrix} ab \\ ac \\ ad \\ bc \\ bd \\ cd \end{bmatrix} = \begin{bmatrix} abc \\ abd \\ acd \\ bcd \end{bmatrix}$$

To check correctness, we examine the triple $(a,b,c)$. We would want to assign this triple the value $\alpha((b,c)) - \alpha((a,c)) + \alpha((b,c))$. We can verify that indeed the row corresponding to edge $(a, b)$ has values $+1$, $-1$, and $+1$ at the columns corresponding to the edges $(b, c)$, $(a, c)$, and $(b, c)$. If any of the edge weights are zero, we remove the corresponding column from the matrix. If any of the triples contain an edge with zero weight, we remove the corresponding row from the matrix.

We now compute $\delta_0^*$ and $\delta_1^*$. Let $M_{\delta_0^*}$ and $M_{\delta_0}$ denote the matrix representations of $\delta_0^*$ and $\delta_0$, respectively. By definition of the adjoint operator, we must have that:

$$\langle \delta_0 f, \alpha \rangle = \langle f, \delta_0^* \alpha \rangle \tag{3.1}$$

$\delta_0 f$ and $\alpha$ are in $C^1$, thus their inner product includes multiplication by edge weights, and is computed as $\sum_{i,j} (\delta_0 f)_{ij} \alpha_{ij} W_{ij}$. $f$ and $\delta_0^*$ are in $C^0$, thus their inner product does not include edge weights. So in order for 3.1 to be valid, we need $\delta_0^*$ to include multiplication by edge weights. Then multiplying the rows of $M_{\delta_0}$ by the corresponding edge weights and transposing the result will yield $M_{\delta_0^*}$. Specifically, $M_{\delta_0^*} = (DM_{\delta_0})^T$ where $D = \text{diag}(W_{ij})$ for all nonzero edges $(i, j)$.

Similarly, in the case of the curl operator we must have:

$$\langle \delta_1 \alpha, A \rangle = \langle \alpha, \delta_1^* A \rangle \tag{3.2}$$

This time $\alpha$ and $\delta_1^* A$ are in $C^1$ while $\delta_1 \alpha$ and $A$ are in $C^2$. Thus only the inner product $\langle \alpha, \delta_1^* A \rangle$ includes multiplication by edge weights. Then dividing the rows of $M_{\delta_1}$ by the corresponding edge weights and transposing the result will yield $M_{\delta_1^*}$. Specifically, $M_{\delta_1^*} = (M_{\delta_1} D')^T$ where $D' = \text{diag}(\frac{1}{W_{ij}})$ for all nonzero edges $(i, j)$.

### 3.1.4 Computation of the Gradient, Curl, and Harmonic Flows

Having defined the necessary matrix representations, we are ready to compute the three flows, which we denots $G$, $C$, and $H$. The gradient flow is the orthogonal projection of $Y$ onto $\text{im}(\delta_0)$. So $G = \delta_0(\delta_0^* \delta_0)^+ \delta_0^* Y$, where $^+$ denotes the Moore-Penrose pseudo-inverse.

The curl flow is the orthogonal projection on to $\text{im}(\delta_1^*)$. We thus hope to find $A \in C^2$ which is the least squares solution to $Y = \delta_1^* A$. Note that this equation is in the inner product space $C_1$ which includes multiplication by edge weights. To solve this equation in the standard inner product space, it is equivalent to multiply both sides by $\sqrt{D}$, and solve $\sqrt{D}Y = \sqrt{D}\delta_1^* A$, where $D = \text{diag}(W_{ij})$. Then $C = \delta_1^* A$.

The harmonic flow is $\ker\delta_0^* \cap \ker\delta_1 = \ker\Delta_1$ where $\Delta_1 = \delta_1^*\delta_1 + \delta_0\delta_0^*$. Then $D\Delta_1 = \Delta_1^T D$. So $D^{1/2}\Delta_1 D^{-1/2} = D^{-1/2}\Delta_1^T D^{1/2}$. So $S = D^{1/2}\Delta_1 D^{-1/2}$ is symmetric, and we can compute its pseudo-inverse. We thus have that the matrix representation of the projection in to $\ker\Delta_1$ is $P = D^{1/2}(S^+S)D^{1/2}$. So $H = (I - P)Y$.

## 3.2 Application of Helmholtz Decomposition

We apply Helmholtz decomposition to a dataset of matches from all major professional tennis tournaments in 2017. There are 528 players and 3831 matches. Each match functions as a comparison between two players. The entries in the $\bar{Y}$ matrix are computed as the fraction of games the winner won minus 0.5. For example, if player B won 0.7 of the games in a match against player A, the entry at index (A, B) in $\bar{Y}$ would be $0.7 - 0.5 = 0.2$. We experiment with four different weighting methods:

1. "Uniform" : All matches are weighted equally
2. "Sets" : A match is weighted by the number of sets played
3. "Games" : A match is weighted by the number of games played
4. "Tourney" : A match is weighted more if it is played in a later round in the tournament. Final-round matches have a weight of 1, semifinal matches a weight of 0.5, quarterfinal matches a weight of 0.25, and so on.

We take a subset of the data consisting of only those matches involving the top 50 players as chosen using "uniform" weighting. We apply Helmholtz decomposition to this smaller dataset using each of the four different weighting methods. We then compare the three flows for each weighting method, where the flows are represented as their fraction of the whole matrix Y, using the Frobenius norm with respect to the $C^1$ inner product $\langle A, B\rangle = \sum_{i,j} A_{ij}B_{ij}W_{ij}$. The results are as follows:

| Weighting Method | Gradient | Curl | Harmonic |
|---|---|---|---|
| Uniform | 0.368911 | 0.630049 | 0.001040 |
| Sets | 0.361733 | 0.637242 | 0.001025 |
| Games | 0.351814 | 0.646887 | 0.001299 |
| Tourney | 0.424746 | 0.574440 | 0.000814 |

Note that all weighting methods produce similar results except for "tourney," in which case the gradient flow is higher and the curl flow lower. This suggests that "tourney" weighting allows for the most consistent ranking of players. This is not intuitive: we would not expect a match to be more reflective of the payers' true skill just because it occurs in a later round.

# Chapter 4

# Cohomology Analysis

## 4.1 Explanation of Cohomology Analysis

## 4.2 Application of Cohomology Analysis

# Chapter 5

# Mapper

## 5.1 Explanation of Mapper

## 5.2 Explanation of Mapper

# Chapter 6

# Laplacian Eigenvector Analysis

## 6.1 Explanation of Laplacian Eigenvector Analysis

## 6.2 Application of Laplacian Eigenvector Analysis

# Chapter 7

# Discussion

# Chapter 8

# Conclusion

# Bibliography

[1] Carlsson, Gunnar. "Topology and data." *Bulletin of the American Mathematical Society* 46.2 (2009): 255-308.

[2] Zomorodian, Afra, and Gunnar Carlsson. "Computing persistent homology." *Discrete & Computational Geometry* 33.2 (2005): 249-274.

[3] De Silva, Vin, Dmitriy Morozov, and Mikael Vejdemo-Johansson. "Persistent cohomology and circular coordinates." *Discrete & Computational Geometry* 45.4 (2011): 737-759.

[4] Giusti, Chad, et al. "Clique topology reveals intrinsic geometric structure in neural correlations." *Proceedings of the National Academy of Sciences* 112.44 (2015): 13455-13460.

[5] Ulmer, M., Lori Ziegelmeier, and Chad M. Topaz. "Assessing biological models using topological data analysis." *arXiv preprint arXiv:1811.04827* (2018).

[6] Jiang, Xiaoye, et al. "Statistical ranking and combinatorial Hodge theory." *Mathematical Programming* 127.1 (2011): 203-244.

[7] Singh, Gurjeet, Facundo Mmoli, and Gunnar E. Carlsson. "Topological methods for the analysis of high dimensional data sets and 3d object recognition." *SPBG.* 2007.