THE UNIVERSITY *of* EDINBURGH

Laplacians for Structure Recovery on Directed and Higher-order Graphs

Xue Gong

Doctor of Philosophy
University of Edinburgh
2023

Declaration

I declare that this thesis was composed by myself and that the work contained therein is my own, except where explicitly stated otherwise in the text.

(*Xue Gong*)

Lay Summary

In this thesis, we explore graph analysis using Laplacian matrices. These matrices provide valuable insights into how components are connected and grouped together. While there has been extensive research on undirected graphs, we identified a gap when it comes to analyzing more complex graphs when the connection is directional or when the relationship involves more than two components. To address this issue, we investigated mathematical frameworks for analyzing directed graphs, hypergraphs, and directed simplicial complexes using graph Laplacians. Specifically, we examined two existing Laplacian matrices for directed graphs which consider the direction of edges and developed associated generative models. We then extend this framework to hypergraphs where edges may connect more than one node. In addition, we defined and analyzed a new Magnetic Hodge Laplacian, which captures directional flows on triangles. Through case studies on triangles and tori, we demonstrated its ability to identify flow direction and discover direction-related patterns.

Abstract

The spectral properties of Laplacian matrices offer valuable insights into the structure of graphs. One significant application is the embedding of nodes into a low-dimensional Euclidean space using the eigenvectors of a suitable Laplacian matrix, enabling the recovery of the underlying connection pattern. Such embedding proves relevant for various subsequent learning tasks, including node reordering, clustering, and visualization. Despite extensive research on undirected graph Laplacians, there remains a gap in analyzing more complex graph structures. This includes situations where relationships are asymmetrical and represented by directed edges, as well as scenarios involving interactions among more than two agents, effectively captured using hypergraphs or simplicial complexes.

To address this research gap, we investigate mathematical frameworks utilizing graph Laplacians for directed networks, hypergraphs, and directed simplicial complexes. For directed graphs, we examine two existing Laplacian matrices associated with periodic and linear structures and developed corresponding generative models. Specifically, we establish and exploit connections between node reordering techniques, achieved through minimizing an objective function, and maximizing the likelihood of a random graph model. Leveraging this random graph framework, we compare the likelihood of each structure. Numerical experiments are conducted using synthetic graphs and graphs from social sciences, biology, and ecology. Furthermore, we extend this framework to hypergraphs and demonstrate its efficacy in quantifying structure strength, performing clustering, and predicting hyperedges on synthetic hypergraphs and social networks. Additionally, we define and analyze a new first-order Magnetic Hodge Laplacian for directed simplicial complexes, capturing directional flows on triangles. Through the analysis of triangle and torus examples, we showcase its ability to discover direction-related patterns.

Acknowledgements

First, I would like to express my sincere gratitude to my supervisors, Des Higham, Kostas Zygalakis, and my collaborator, Ginestra Bianconi, for their invaluable guidance and support. Their patience, inspiration, and thoughtful feedback have contributed significantly to shaping this work. I am also grateful to the MAC-MIGS CDT Scholarship under EPSRC grant no. EP/S023291/1 and the Alan Turing Institute for funding my research and providing a conducive research environment. I thank Colin Singleton from the CountingLab for suggesting the Dunnhumby data used in this thesis.

I greatly appreciate my family for their unwavering support. Their love and belief in me have been a constant source of motivation throughout this journey. I am especially grateful for the weekly chats with my mom, whose cheerful and relaxed personality always brings a sense of calm. Having a twin sister, Lynn, has been a blessing. Her encouragement has been a source of strength for me.

I want to extend my heartfelt gratitude to my friends, whose support has been invaluable. To Yajing, for being such a good friend and walking together on this research journey. To Xutong, Yue, Lin, Lily, Penghua, and Yano, for your friendship and kindness. To all my friends in Edinburgh, you have made this experience truly memorable. And to Mica, thank you for your advice and encouragement.

Finally, I would like to especially thank Jason for his love, support, and trust. He has been the greatest source of motivation throughout the completion of this thesis. Thank you for your constant encouragement and for standing by my side during this journey.

To Jason

Contents

Chapter 1

Introduction and Overview

1.1 Introduction

Graph-typed data sets capture interactions between pairs of nodes, where a node represents an agent, such as a person, molecule, or website, and an edge represents the relationship between two agents, such as their friendship, functional interaction, or hyperlinks. Typical graph-typed data, therefore, captures pairwise relationships. Hypergraphs and simplicial complexes generalize graphs to higher orders where interactions can involve more than two nodes. The term "graph" is commonly used in the fields of mathematics and computer science, while the term "network" is more widely employed in the domains of data science, biology, and social sciences. For this thesis, we will be using the terms "graph" and "network" interchangeably.

Graph-typed data are prevalent in social science and biology when the interactions between components can be directly observed. Examples include political blog networks [1], transportation networks [34], and gene-regulatory networks [98]. In such scenarios, important questions arise, such as: "what are the closely connected clusters?", "can we predict node labels based on connections?", and "can we infer missing links from observed data?" In essence, our goal is to deduce node-level information from their patterns of interaction.

In other applications, we may only have node-level information, and edges do not exist in the raw data. In this case, we can construct the edges based on affinity and similarity between data points. Such constructions allow for the application of graph-based algorithms to effectively learn from the data. For example, with point cloud data or image data, one can construct graphs from pairwise distances or similarities. Then spectral clustering using the graph Laplacian matrix can help classify and segment image data [99]. Compared with k-means, which returns clusters that are contained in convex sets, spectral clustering can detect non-convex boundaries between clusters. The latter case is called non-linear clustering [108]. In another example in geometric processing, graphs or simplicial complexes can arise from the discretization of continuous manifolds [8, 97]. By computing a heat flow involving a graph connection Laplacian, one can efficiently compute the parallel transport of tangent vectors on the curved surface.

One of the fundamental tasks on this type of data is to embed the nodes into

a low-dimensional Euclidean space [90, 96] to facilitate various downstream tasks, such as node clustering, node reordering, and graph visualization. Node clustering is an important unsupervised learning task on graphs, which partitions nodes into disjoint sets, where typically, nodes within the same cluster tend to have more interactions with each other than with nodes from different clusters. Node reordering is helpful in revealing a hierarchy between the nodes, such as ranking sports teams based on their pairwise match results. Graph visualization typically involves positioning nodes according to their learned embedding to reveal the underlying structural patterns in the data. However, in real-world datasets, these patterns may be obscure. Therefore, in addition to visualizing the reordered network, it is essential to quantify the relative strength of different structures.

Several classes of embedding and clustering algorithms are used for graphs. One type of algorithm involves analyzing Laplacian matrices that are related to the graph [17]. Among these algorithms, spectral clustering is one of the most extensively used [70]. This algorithm embeds vertices using eigenvectors of a graph Laplacian matrix and applies k-means clustering to the embedded vertices to partition the nodes into k clusters. Spectral clustering methods have been extensively used due to their many advantages. One of their benefits is that they can be implemented efficiently on vast sparse graphs, which are ubiquitous in real-world applications. Furthermore, they have a robust theoretical foundation, as they are supported by accompanying consistency theory [26, 107] and performance guarantees [71, 86]. Additionally, they require less parameter tuning and have demonstrated outstanding performance in various applications. In this thesis, we focus on this category of algorithms, often referred to as spectral algorithms, to derive node embeddings from eigenvectors of a Laplacian matrix.

Another type of algorithm involves solving maximum likelihood problems that are associated with random graph models. In this setting, one assumes that the graph is generated from a random process and then uses maximum likelihood estimation to infer the underlying parameters of the model that best fit the observed data. The Stochastic Block Model (SBM) [85] is one of the most popular models used in this context, which assumes that the graph contains several densely connected subgroups.

In more complex scenarios, machine learning frameworks are utilized to develop more sophisticated algorithms for graph analysis. For instance, the Graph Convolutional Network (GCN) [46] framework uses a neural network to learn node embeddings by aggregating information from the local neighborhood.

In this thesis, we take advantage of the concept of associating a spectral method with a generative random graph model. This approach allows us to compare the outputs of spectral methods based on the likelihood of the associated random graph. This connection was first proposed in [48], which demonstrated that the standard spectral method for undirected networks is equivalent to maximum likelihood optimization assuming a class of range-dependent random graphs (RDRGs) introduced in [44]. In [45, 86], this idea was further developed, and a likelihood ratio test was created to determine whether a network with a range-dependent structure is more linear or periodic.

The thesis is organized as follows. In this chapter, we present background material and examine the node embedding problem using the graph Laplacian. Ad-

ditionally, we explore how this relates to minimizing distances between neighbors in the embedded space. Chapter 2 will center on the directed graph Laplacians. Specifically, we will explore two spectral embedding algorithms for directed graphs and develop corresponding models for random graphs. These models will then be utilized to measure the strength of linear and periodic structures. In Chapter 3, we will extend the models to hypergraphs, which have edges that connect more than two nodes. Since considering direction in hypergraphs is challenging and strongly application dependent, we will focus on the undirected case. We will then examine random hypergraph models for linear and periodic structures and present empirical results for structure quantification and hyperedge prediction. Chapter 4 will explore the Hodge Laplacian for directed simplicial complexes. We will introduce a new 1-Magnetic Hodge Laplacian that incorporates the direction of triangles. The Laplacian will be analyzed on directed triangles and tori. We demonstrate that the proposed Magnetic Laplacian can assign phase angles to edges based on edge and triangle directions. Lastly, we conclude the thesis and discuss future research directions in Chapter 5.

1.2 Notation

Here we introduce some of the notation used throughout the thesis. Let $G = (V, E)$ be a graph consisting of a finite set of nodes $V := \{1, 2, ..., n\}$ and an edge set $E = \{\{i, j\} \mid i \neq j \in V\}$. Here we assume an edge is an ordered set containing two distinct vertices, meaning that self-loops are not allowed. An edge $E = \{i, j\}$ can be illustrated as $i \to j$, where i has a link to j. We can conveniently represent all the nodes and edges in a graph with an $n \times n$ adjacency matrix A, such that

$$A_{ij} = \begin{cases} 1 & \text{if } i, j \in E \\ 0 & \text{otherwise.} \end{cases} \tag{1.1}$$

The graph is *undirected* when A is symmetric, and *directed* otherwise. In subsequent chapters, we will also consider weighted graphs, where each edge $i \to j$ is assigned a non-negative weight w_{ij}. In this case, we let $A_{ij} = w_{ij}$. The *degree matrix* D is a diagonal matrix with $D_{ii} = d_i$, where $d_i = \sum_j A_{ij}$ is the degree of the node i. A path on a graph is a sequence of distinct vertices such that each pair of adjacent vertices are connected by an edge. The graph is *connected* if there exists a path between all pairs of vertices; and the graph is *disconnected* otherwise. Moreover, we denote $\mathbf{i}$ as the square root of -1. $\boldsymbol{x}^T$ represents the transpose of a vector $\boldsymbol{x} \in \mathbb{R}^n$, $\boldsymbol{x}^H$ represents the conjugate transpose of a complex vector $\boldsymbol{x} \in \mathbb{C}^n$, and $\|\boldsymbol{x}\|$ refers to the l^2 norm of the vector. Let $\boldsymbol{x} \perp \boldsymbol{\nu}$ denote that $\boldsymbol{x}$ is orthogonal to $\boldsymbol{\nu}$. The set of all permutation vectors, which consists of all vectors in $\mathbb{R}^n$ with distinct components given by the integers $1, 2, \ldots, n$, is denoted as $\mathcal{P}$. We use $\mathbf{1}$ to represent an all-ones vector and I_d stands for the $d \times d$ identity matrix.

1.3 Graph Laplacian

1.3.1 Undirected Graph Laplacian

In this section, we consider graph Laplacians for undirected graphs. The standard graph Laplacian matrix is widely used for node clustering, and it is defined as

$$L = D - A. \tag{1.2}$$

Equivalently, one can write

$$L_{ij} = \begin{cases} d_i & \text{if } i = j \\ -1 & \text{if } \{i, j\} \in E \\ 0 & \text{otherwise.} \end{cases}$$

Various versions of normalized Laplacian matrix have been widely used in clustering and reordering tasks. For example, the symmetric normalized Laplacian matrix is defined as

$$L_{norm} = I_n - D^{-1/2} A D^{-1/2}.$$

Here, we assume that every node has degree ≥ 1, so that D is invertible. Eigenvalues of this matrix lie between 0 and 2 [17], which makes it easier to compare graphs of different sizes. The eigenvalues and eigenvectors of the Laplacian carry rich information about the geometry and connectivity of the graph. The Laplacian L can be interpreted as an operator on the space of real-valued functions defined on nodes.

Proposition 1.3.1 (Laplacian Operator on Graph)**.** *For a real-valued function defined on nodes $g : V \to \mathbb{R}$, we have*

$$Lg(i) = \sum_{j \in V} A_{ij}(g(i) - g(j)).$$

Proof. We have

$$Lg(i) = d_i g(i) - \sum_{j \in V} A_{ij} g(j) = \sum_{j \in V} A_{ij} g(i) - \sum_{j \in V} A_{ij} g(j) = \sum_{j \in V} A_{ij}(g(i) - g(j)).$$

□

In other words, the Laplacian operator calculates the overall difference in the function values across the nodes that are connected to each other.

1.3.2 Spectral Properties of Graph Laplacian

The Laplacian matrix here is symmetric since both D and A are symmetric in the undirected case. Therefore it is Hermitian, and we can apply the spectral decomposition stated below. Note that while the standard graph Laplacian is

real, we will need to address the complex graph Laplacian later in the thesis, especially when considering directed graphs. Consequently, in the following theorem, we make the assumption that the matrix is Hermitian to ensure that it can be generalized to complex cases.

Theorem 1.3.1 (Spectral Decomposition of Hermitian Matrices [69])**.** *Let $\mathcal{A} \in \mathbb{C}^{m\times m}$ be a Hermitian matrix with eigenvalues $\lambda_1 \leq \lambda_2 \leq ... \leq \lambda_m$ and associated orthonormal eigenvectors $\boldsymbol{v}_1, \boldsymbol{v}_2, ..., \boldsymbol{v}_m$ such that $\mathcal{A}\boldsymbol{v}_i = \lambda_i \boldsymbol{v}_i$. Then there is a unitary matrix U, whose columns are $\boldsymbol{v}_1, \boldsymbol{v}_2, ..., \boldsymbol{v}_m$, and a real diagonal matrix $\Lambda = diag(\lambda_1, \lambda_2, ..., \lambda_m)$ such that*

$$\mathcal{A} = U\Lambda U^H = \sum_{i=1}^{m} \lambda_i \boldsymbol{v}_i \boldsymbol{v}_i^H.$$

Furthermore, many interesting properties of the graph Laplacian can be found by examining the following quadratic form.

Proposition 1.3.2 (Quadratic Form of the Graph Laplacian)**.** *For any vector $\boldsymbol{x} = [x_1, x_2, ..., x_n]^T \in \mathbb{R}^n$, we can write*

$$\boldsymbol{x}^T L \boldsymbol{x} = \tfrac{1}{2} \sum_{i,j} A_{ij}(x_i - x_j)^2. \tag{1.3}$$

We reproduce the proof from [70] below:

Proof. We have

$$\begin{aligned}
\boldsymbol{x}^T L \boldsymbol{x} = \boldsymbol{x}^T (D - A)\boldsymbol{x} = \boldsymbol{x}^T D \boldsymbol{x} - \boldsymbol{x}^T A \boldsymbol{x} &= \sum_{i\in V} x_i \sum_{j\in V} A_{ij} x_i - \sum_{i,j\in V} x_i A_{ij} x_j \\
&= \tfrac{1}{2}\Big(\sum_{i,j\in V} x_i^2 A_{ij} + \sum_{j,i\in V} x_j^2 A_{ji} - 2\sum_{i,j\in V} x_i A_{ij} x_j\Big) \\
&= \tfrac{1}{2} \sum_{i,j\in V} A_{ij}(x_i^2 + x_j^2 - 2x_i x_j) = \tfrac{1}{2} \sum_{i,j\in V} A_{ij}(x_i - x_j)^2.
\end{aligned}$$

□

From the quadratic form (1.3), it is straightforward to see that the Laplacian matrix is positive-semidefinite. It then follows that all of its eigenvalues are non-negative. It can also be confirmed that 0 is an eigenvalue of L when $\boldsymbol{x}$ is a constant vector. Combined with Theorem 1.3.1, we can let $0 = \lambda_1 \leq \lambda_2 \leq ... \leq \lambda_n$ be the eigenvalues of the Laplacian matrix L with associated eigenvectors $\boldsymbol{v}_1, \boldsymbol{v}_2, ..., \boldsymbol{v}_n$. Another interesting property is that the multiplicity and eigenspace associated with the eigenvalue zero can reveal the connected components in the graph.

Proposition 1.3.3 (Number of Connected Components Graph Spectrum)**.** *The multiplicity of the eigenvalue 0 of L equals the number of connected components in the graph. Additionally, the indicator vectors of those components span the eigenspace of eigenvalue 0.*

We omit the proof here and refer the interested reader to [17, 70] for details. The proposition says that we can find the connected clusters by examining the zero eigenvalue and its corresponding eigenvector(s). When the graph is connected, so there is only one connected component in the graph, the eigenvector associated with eigenvalue 0 becomes a constant vector. Furthermore, if the second smallest eigenvalue is positive, one can conclude that the graph is connected. The second smallest eigenvalue is also called the algebraic connectivity or Fiedler value.

Corollary 1.3.1 (Algebraic Connectivity)**.** *Let* $0 = \lambda_1 \leq \lambda_2 \leq ... \leq \lambda_n$ *be the eigenvalues of the Laplacian matrix* L*, then the graph is connected if and only if* $\lambda_2 \neq 0$*.*

This naturally leads to the question of whether we can obtain insights from non-zero eigenvalues as well. To understand this question, we first give the definition of the Rayleigh quotient below.

Definition 1.3.1 (Rayleigh quotient)**.** *Let* $\mathcal{A}$ *be an* $n \times n$ *Hermitian matrix. Its Rayleigh quotient for a vector* $\boldsymbol{x} \in \mathbb{C}^n$ *is defined as*

$$R_{\mathcal{A}}(\boldsymbol{x}) \equiv \frac{\boldsymbol{x}^H \mathcal{A} \boldsymbol{x}}{\boldsymbol{x}^H \boldsymbol{x}}.$$

It is easy to show that the Rayleigh quotient for an eigenvector is exactly its eigenvalue, that is, if $\mathcal{A}\boldsymbol{v} = \lambda \boldsymbol{v}$, then

$$R_{\mathcal{A}}(\boldsymbol{v}) \equiv \frac{\boldsymbol{v}^H \mathcal{A} \boldsymbol{v}}{\boldsymbol{v}^H \boldsymbol{v}} = \frac{\boldsymbol{v}^H \lambda \boldsymbol{v}}{\boldsymbol{v}^H \boldsymbol{v}} = \lambda.$$

The Rayleigh-Ritz theorem [69] states that the eigenvector associated with the smallest eigenvalue of a Hermitian matrix minimizes the Rayleigh quotient, and similarly, the eigenvector of the largest eigenvalue maximizes the Rayleigh quotient.

Theorem 1.3.2 (Rayleigh-Ritz Theorem)**.** *Let* $\mathcal{A} \in \mathbb{C}^{n \times n}$ *be a Hermitian matrix, and* λ_{min}*,* λ_{max} *be its smallest and largest eigenvalue, then*

$$\lambda_{min} = \min_{\|\boldsymbol{v}\| \neq 0} R_{\mathcal{A}}(\boldsymbol{v}),$$

and

$$\lambda_{max} = \max_{\|\boldsymbol{v}\| \neq 0} R_{\mathcal{A}}(\boldsymbol{v}).$$

This implies that the unit vector $\hat{\boldsymbol{v}}$ associated with the smallest eigenvector minimizes the quadratic form of the Hermitian matrix $\hat{\boldsymbol{v}}^H \mathcal{A} \hat{\boldsymbol{v}}$ since

$$\hat{\boldsymbol{v}}^H \mathcal{A} \hat{\boldsymbol{v}} = R_{\mathcal{A}}(\hat{\boldsymbol{v}}) \cdot \hat{\boldsymbol{v}}^H \boldsymbol{v} = R_{\mathcal{A}}(\hat{\boldsymbol{v}}) \cdot 1 = R_{\mathcal{A}}(\hat{\boldsymbol{v}}).$$

Additionally, we can extend the interpretation to other eigenvalues with the Courant-Fischer Theorem [69] stated below.

Theorem 1.3.3 (Courant-Fischer Theorem)**.** *Let $\mathcal{A} \in \mathbb{C}^{m\times m}$ be a Hermitian matrix with eigenvalues $\lambda_1 \leq \lambda_2 \leq ... \leq \lambda_m$ and associated eigenvectors $\boldsymbol{v}^{[1]}, \boldsymbol{v}^{[2]}, ..., \boldsymbol{v}^{[m]}$ such that $\mathcal{A}\boldsymbol{v}^{[i]} = \lambda_i \boldsymbol{v}^{[i]}$. Then,*

$$\lambda_1 = \min_{\|\boldsymbol{x}\|\neq 0} R_{\mathcal{A}}(\boldsymbol{x}),$$

and for $1 < i \leq n$

$$\lambda_i = \min_{\|\boldsymbol{x}\|\neq 0, \boldsymbol{x}\perp \boldsymbol{v}^{[j]} \text{ for } j<i} R_{\mathcal{A}}(\boldsymbol{x}),$$

and

$$\lambda_i = \max_{\|\boldsymbol{x}\|\neq 0, \boldsymbol{x}\perp \boldsymbol{v}^{[j]} \text{ for } j>i} R_{\mathcal{A}}(\boldsymbol{x}).$$

1.3.3 Spectral Embedding

Spectral methods study the structural properties of a graph through the eigenvalues and eigenvectors of its Laplacian matrices. Such methods have found applications in many fields, including computer science [99], biology [50], and social sciences [110]. They offer a powerful tool for understanding the structure and properties of graphs and are widely used in various applications such as clustering, ranking, and visualization. In this section, we explore spectral embedding, which maps nodes into Euclidean space using eigenvectors of a graph Laplacian.

One-dimensional Embedding

Consider an undirected connected graph $G = (V, E)$ with n nodes. Suppose we would like to plot all nodes in the graph on a real-axis. Therefore the node position can be denoted as $\boldsymbol{x} \in \mathbb{R}^n$. Let us consider the problem of finding $\boldsymbol{x}$ that minimizes the quadratic form of the graph Laplacian L

$$\boldsymbol{x}^T L \boldsymbol{x} = \tfrac{1}{2} \sum_{i,j\in V} A_{ij}(x_i - x_j)^2.$$

The quadratic form measures the total distance between linked nodes in the embedded space. Therefore the minimization problem can be interpreted as plotting each node on the real axis such that the neighboring nodes are near each other in the embedded space.

Fiedler Vector The problem above is susceptible to the trivial solution that all nodes are assigned to zero, or $\boldsymbol{x}$ is a constant vector. Therefore we impose the constraint that

$$\|\boldsymbol{x}\| = 1. \tag{1.4}$$

Appealing to the Rayleigh-Ritz theorem, the optimal solution is given by the eigenvector corresponding to the smallest eigenvalue of the Laplacian. When the graph is connected, according to Proposition 1.3.3, this is just a constant vector $\mathbf{1}/\sqrt{n}$, which is not useful for visualizing the graph. Therefore, we further impose

that

$$\boldsymbol{x} \perp \mathbf{1}. \tag{1.5}$$

From Theorem 1.3.2 and 1.3.3 it is clear that the minimum of the quadratic form subject to constraints (1.4) and (1.5) is achieved by the eigenvector associated with the second smallest eigenvalue of the graph Laplacian, also called the *Fiedler vector*. We therefore have the following result.

Proposition 1.3.4. *Let L be the graph Laplacian of an undirected connected graph $G = (V, E)$ with n vertices. We also let $0 = \lambda_1 < \lambda_2 \leq ... \leq \lambda_n$ be the eigenvalues of L associated with eigenvectors $\boldsymbol{v}^{[1]}, \boldsymbol{v}^{[2]}, ..., \boldsymbol{v}^{[m]}$. Then*

$$\lambda_2 = \min_{\|\boldsymbol{x}\| \neq 0, \boldsymbol{x} \perp \mathbf{1}} \sum_{i,j \in V} A_{ij}(x_i - x_j)^2,$$

and the minimizing vector is $\boldsymbol{v}^{[2]}$.

Other Eigenvectors One can also generate a one-dimensional embedding with $\boldsymbol{v}^{[3]}$. In fact, $\boldsymbol{v}^{[3]}$ can be interpreted as the Fiedler vector of a new matrix where the new adjacency matrix $\tilde{A}$ is given by

$$\tilde{A}_{ij} = A_{ij} - \mu v_i^{[2]} v_j^{[2]},$$

for any constant $\mu > \lambda_3 - \lambda_2$ [50]. Such modification reduces weight if two neighbors have the same sign in $\boldsymbol{v}^{[2]}$ and increases weight if they are assigned different signs in $\boldsymbol{v}^{[2]}$. Therefore, $\boldsymbol{v}^{[3]}$ can be interpreted as the minimizing vector to a new problem that corrects the tendency of $\boldsymbol{v}^{[2]}$ to partition nodes into two clusters according to the sign. Similar arguments can be applied to other eigenvectors [50]. For example, by subtracting $\boldsymbol{v}^{[2]}\boldsymbol{v}^{[2]^T}$, $\boldsymbol{v}^{[3]}\boldsymbol{v}^{[3]^T}$, ..., $\boldsymbol{v}^{[k-1]}\boldsymbol{v}^{[k-1]^T}$ with sufficiently large multipliers, $\boldsymbol{v}^{[k]}$ will become the Fiedler vector of the modified Laplacian.

Higher-dimensional Embedding

Now, consider the case where nodes are embedded into a d-dimensional Euclidean space for some $d \geq 2$, while minimizing the squared Euclidean distance between neighboring nodes. Let $\boldsymbol{x}^{[k]} = (x_1^{[k]}, x_2^{[k]}, ..., x_n^{[k]})^T \in \mathbb{R}^n$ represent the location in the k-th dimension, and let $X = [\boldsymbol{x}^{[1]}, \boldsymbol{x}^{[2]}, ..., \boldsymbol{x}^{[d]}] \in \mathbb{R}^{n \times d}$ denote d-dimensional positions. This implies that each node i is mapped to $\boldsymbol{x}_i = (x_i^{[1]}, x_i^{[2]}, ..., x_i^{[d]}) \in \mathbb{R}^d$, which corresponds to the i-th row in X. Then the total squared Euclidean distance between neighbors can be written as

$$\sum_{i,j \in V} A_{ij} \|\boldsymbol{x}_i - \boldsymbol{x}_j\|^2. \tag{1.6}$$

The distance above is related to the trace of the matrix $X^T L X$ and can be minimized with the eigenvectors of the the graph Laplacian L.

Theorem 1.3.4 (Optimization of Trace [69])**.** *Let $\mathcal{A} \in \mathbb{C}^{m \times m}$ be a Hermitian matrix with eigenvalues $\lambda_1 \leq \lambda_2 \leq ... \leq \lambda_m$ and associated eigenvectors*

$\boldsymbol{v}^{[1]}, \boldsymbol{v}^{[2]}, ..., \boldsymbol{v}^{[m]}$ *such that* $\mathcal{A}\boldsymbol{v}^{[i]} = \lambda_i \boldsymbol{v}^{[i]}$. *Then,*

$$\lambda_1 + \lambda_2 + ... + \lambda_d = \min_{B \in \mathbb{C}^{m \times d},\ B^H B = I_d} tr(B^H \mathcal{A} B),$$

and a minimizing matrix is $B = [\boldsymbol{v}^{[1]}, \boldsymbol{v}^{[2]}, ..., \boldsymbol{v}^{[d]}]$*;*

$$\lambda_m + \lambda_{m-1} + ... + \lambda_{m-d+1} = \max_{B \in \mathbb{C}^{m \times d},\ B^H B = I_d} tr(B^H \mathcal{A} B),$$

and a maximizing matrix is $B = [\boldsymbol{v}^{[m]}, \boldsymbol{v}^{[m-1]}, ..., \boldsymbol{v}^{[m-d+1]}]$.

According to the theorem above, one can use the top smallest eigenvalues to minimize the trace. From this result we derive the following theorem:

Proposition 1.3.5. *Consider an undirected connected graph* $G = (V, E)$ *with* n *vertices. Let* $0 = \lambda_1 < \lambda_2 \leq ... \leq \lambda_n$ *be the eigenvalues of its graph Laplacian* L *with associated eigenvectors* $\boldsymbol{v}^{[1]}, \boldsymbol{v}^{[2]}, ..., \boldsymbol{v}^{[m]}$. *Then*

$$\lambda_1 + \lambda_2 + ... + \lambda_d = \min_{X \in \mathbb{R}^{n \times d},\ X^T X = I_d} \sum_{i,j \in V} A_{ij} \|\boldsymbol{x}_i - \boldsymbol{x}_j\|^2,$$

where $X = [\boldsymbol{x}^{[1]}, \boldsymbol{x}^{[2]}, ..., \boldsymbol{x}^{[d]}]$ *and a minimizing solution is* $X = [\boldsymbol{v}^{[1]}, \boldsymbol{v}^{[2]}, ..., \boldsymbol{v}^{[d]}]$.

Proof. The trace of $X^T L X$ is exactly half of the squared Euclidean distance we hope to minimize:

$$\begin{aligned} tr(X^T L X) &= \sum_{k=1}^{d} (\boldsymbol{x}^{[k]})^T L \boldsymbol{x}^{[k]} \\ &= \tfrac{1}{2} \sum_{k=1}^{d} \sum_{i,j \in V} A_{ij} (x_i^{[k]} - x_j^{[k]})^2 \\ &= \tfrac{1}{2} \sum_{i,j \in V} A_{ij} \|\boldsymbol{x}_i - \boldsymbol{x}_j\|^2. \end{aligned}$$

From here we can complete the proof with Theorem 1.3.4. □

Therefore embedding nodes into d-dimensional space with d orthogonal vectors while minimizing the total squared distances between neighbors can be achieved by mapping each node i into the i-th row of $X = [\boldsymbol{v}^{[1]}, \boldsymbol{v}^{[2]}, ..., \boldsymbol{v}^{[d]}]$, that is, letting $\boldsymbol{x}_i = (v_i^{[1]}, v_i^{[2]}, ..., v_i^{[d]}) \in \mathbb{R}^d$.

1.3.4 Directed Graph Laplacian

While spectral methods have been widely used in analyzing undirected graphs, Laplacians for directed graphs are less well-studied. Nonetheless, there have been several extensions of the standard Laplacian matrix specifically for the directed case.

One of the spectral methods for directed networks is the Magnetic Laplacian algorithm, which was introduced in [30, 31]. The Magnetic Laplacian is a

complex Hermitian matrix that can be derived from the adjacency matrix of a directed network and is designed to capture periodic flow structures. Another spectral method for directed networks is the Trophic Laplacian algorithm which was proposed in [73]. The Trophic Laplacian is a matrix that captures the hierarchical structure of the network, which is the flow of resources from lower to higher trophic levels. We will discuss these two directed Laplacian in more detail in Chapter 2. Other frameworks for directed graphs include the Hermitian matrix method, which was introduced in [23]. This method groups nodes into clusters based on a strong imbalance of flow between clusters. It constructs a skew-symmetric matrix that emphasizes net flow between pairs of nodes but ignores reciprocal edges. Another related technique is the spectral clustering algorithm motivated by random walks, which was derived in [84]. This approach leads to a graph Laplacian for directed networks that was proposed earlier in [18]. This Laplacian matrix is similar to the standard graph Laplacian used for undirected networks but with modifications to account for the directed edges in the network.

1.4 Higher-order Networks

In recent years, there has been a growing interest in understanding higher-order interactions in complex networks. These higher-order interactions involve multiple nodes and cannot be captured by traditional pairwise interactions. Researchers have proposed different paradigms to capture such group-level interactions. Two of the most widely studied frameworks are hypergraphs and simplicial complexes.

In the hypergraph formulation, each hyperedge involves two or more nodes. Hypergraphs have been extensively studied [3, 4, 103], and have found application in various real-world problems such as epidemic spread modeling [51], image classification [115], and the study of biological networks [88]. We will introduce the definition of hypergraph and related notations in more detail in Section 3.2.

A simplicial complex is a special case of a hypergraph that consists of simplices. Each simplex represents a set of nodes that are mutually connected. Simplicial complexes have been used to study various network properties such as clustering, community structure, motif analysis, and topology [2, 7, 83]. Section 4.2 will delve into the theoretical background of simplicial complexes.

1.5 Node Embedding

A fundamental learning task on graph-based data is to embed nodes into a low-dimensional Euclidean space [90, 96]. The learned embedding can be used in follow-on tasks such as clustering, classification, and structure recovery. There are various types of learning algorithms for graphs; some design and analyze Laplacian matrices related to the graph [70], some solve maximum likelihood problems associated with random graph models [44, 85], and others involve more complex machine learning frameworks [46, 47]. In this thesis, we build on the

spectral approach outlined in Section 1.3.3, which derives node embeddings from eigenvectors of a Laplacian matrix. Such spectral algorithms are popular since they can be implemented efficiently on large sparse graphs, and they are backed up by accompanying consistency theory [26].

1.6 Generative Graph Models

Another aspect of our work is the connection between spectral methods and random models. Many graph embedding [54], reordering [44], clustering [16], and structure recovery [21, 105] techniques solve maximum-likelihood problems on graphs assuming specific generative models. Besides their application in these inverse problems, random graph models are useful inference tools for quantifying structure, predicting new or missing links, and improving the interpretability of learning algorithms by relating node embeddings to edge probabilities [45]. The random graph models relevant to this thesis are introduced in Sections 2.3 and 3.5.

Many spectral algorithms are naturally related to optimization problems. This is the case when the Laplacian matrix is Hermitian so that its eigenvectors are critical points of a quadratic form [69]. For example, spectral embedding for undirected graphs using the standard combinatorial Laplacian is related to minimizing the unnormalized cut [52, 70]. Sometimes the objective to be optimized is called the incoherence or frustration of the graph; for example, the Magnetic Laplacian for a directed graph minimizes the frustration defined in [31]. In addition, such optimization formulations may lead to interesting random graph interpretations of spectral algorithms. When the quadratic form can be expressed as the log-likelihood of the graph under a suitable model, the optimization problem may be restated as a node reordering. Such connections have been investigated for undirected graphs [48], and in Chapters 2 and 3, we extend these ideas to directed graphs and hypergraphs.

Chapter 2

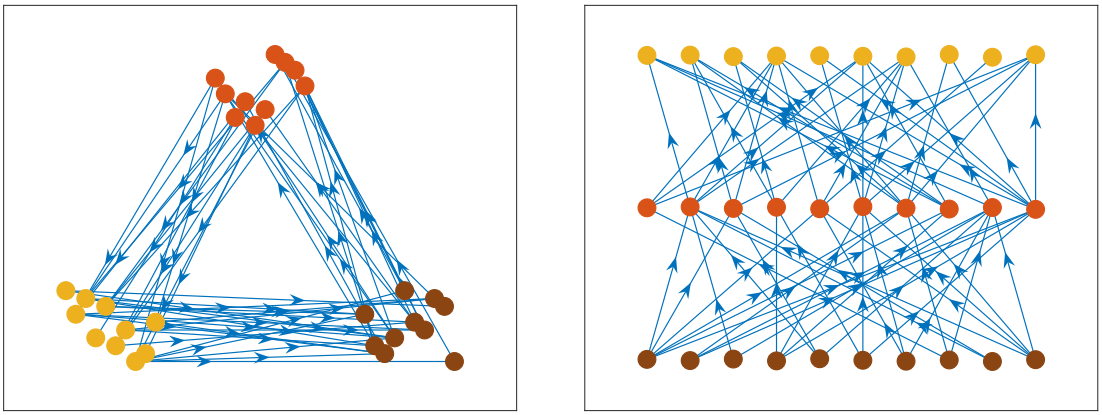
Directed Graph Laplacians

2.1 Introduction

Uncovering structure by clustering or reordering nodes is an important and widely studied topic in network science [70, 102]. The issue is especially challenging if we move from undirected to directed networks, because there is a greater variety of possible structures. For example, even a simple motif of three connected nodes has 13 distinct forms [5, Figure 1(a)]. Moreover, when spectral methods are employed, directed edges lead to asymmetric eigenproblems [23, 31, 73, 74]. Our objective in this chapter[1] is to study spectral (Laplacian-based) methods for directed networks that aim to reveal *clustered, directed, hierarchical structure*; that is, groups of nodes that are related because, when visualized appropriately, one group is seen to have links that are directed towards the next group. This hierarchy may be periodic or linear, depending on whether there are well-defined start and end groups. Figures 2.1a and 2.1b illustrate the two cases. Mapping a network to a linear structure may help us understand the upstreamness and downstreamness of nodes, which is useful, for example, in the study of cascading effects such as social or financial contagion [91]. Similarly, periodic hierarchies have been associated with sustainability and risk management issues in commerce [15], and also with the existence of echo chambers in online social media [58].

Of course, on real data these structures may not be so pronounced; hence in addition to visualizing the reordered network, we are interested in quantifying the relative strength of each type of signal. Laplacian-based methods are often motivated from the viewpoint of optimizing an objective function. This chapter focuses on two such methods. Minimizing *frustration* leads to the *Magnetic Laplacian* which may be used to reveal periodic hierarchy [30, 31]. Minimizing *trophic incoherence* leads to what we call the *Trophic Laplacian*, which may be used to reveal linear hierarchy [73]. We will exploit the idea of associating a spectral method with a generative random graph model. This in turn allows us to compare the outputs from spectral methods based on the likelihood of the associated random graph. This connection was proposed in [48] to show that the standard spectral method for undirected networks is equivalent to maximum likelihood optimization assuming a class of range-dependent random graphs (RDRGs) in-

[1] The content of this chapter is adapted from [42].



(a) (b)

Figure 2.1: Directed networks with (a) periodic hierarchy (edges point from nodes in one cluster to nodes in the next cluster, counterclockwise) and (b) linear hierarchy (edges point from nodes in one level to nodes in the next highest level). Node colours indicate the three clusters.

troduced in [44]. The idea was further pursued in [45], where a likelihood ratio test was developed to determine whether a network with RDRG structure is more linear or periodic.

The main contributions of this chapter are as follows.

- We propose two new directed random graph models. One model has the unusual property that the probability of an $i \to j$ connection is not independent of the probability of the reciprocated $j \to i$ connection.

- We establish connections between these random graph models and algorithms from [30] and [73] that use the Magnetic Laplacian and Trophic Laplacian, respectively, by showing that reordering nodes or mapping them onto a specific lattice structure using these algorithms is equivalent to maximizing the likelihood that the network is generated by the models proposed.

- We show that by calibrating a given network to both models, it is possible to quantify the relative presence of periodic and linear hierarchical structures using a likelihood ratio.

- We illustrate the approach on synthetic and real networks.

The rest of the chapter is organised as follows. In the next section, we introduce the Magnetic and Trophic Laplacian algorithms. Section 2.3 defines the new classes of random directed graphs and establishes their connection to these spectral methods. Illustrative numerical results on synthetic networks are given in Section 2.4, and in Section 2.5 we show results on real networks from a range of applications areas. We finish with a brief discussion in Section 2.6.

2.2 Magnetic and Trophic Laplacians

2.2.1 Notation

We consider an unweighted directed graph $G = (V, E)$ with node set V and edge set E, with no self-loops. The adjacency matrix A is defined in Section 1.1. Since A is asymmetric in general, it is useful to define the symmetrized adjacency matrix

$$W^{(s)} = \frac{A + A^T}{2}. \tag{2.1}$$

It is straightforward to see that $W^{(s)} = A$ when the graph is undirected. The symmetrized degree matrix D is diagonal with $D_{ii} = d_i$, where

$$d_i = \sum_j W^{(s)}_{ij}$$

is the average of the in-degree and out-degree of node i.

2.2.2 Spectral Methods for Directed Networks

Spectral methods explore properties of graphs through the eigenvalues and eigenvectors of associated matrices [17, 50, 70, 102]. In the undirected case, the standard graph Laplacian defined in Equation (1.2) is widely-used for clustering and reordering, along with normalized variants. For instance, in Section 1.3.3, we demonstrated the utilization of the Laplacian matrix for node embedding.

The directed case has received less attention; however, several extensions of the standard Laplacian have been proposed [74]. We focus on two spectral methods for directed networks, which are discussed in the next two subsections: the Magnetic Laplacian algorithm, which reveals periodic flow structures [30, 31], and the Trophic Laplacian algorithm, which reveals linear hierarchical structures [73]. We choose to study these two algorithms because they have an optimization formulation and, as we show in Section 2.3, may be interpreted in terms of random graph models. We mentioned some other Laplacian matrices for analyzing directed graphs in Section 1.3.4.

2.2.3 The Magnetic Laplacian

Given a network and a vector of angles $\boldsymbol{\theta} = (\theta_1, \theta_2, ..., \theta_n)^T$ in $[0, 2\pi)$, we may define the corresponding *frustration*

$$\eta(\boldsymbol{\theta}) = \sum_{i,j} W^{(s)}_{ij} |e^{\mathbf{i}\theta_i} - e^{\mathbf{i}\delta_{ij}} e^{\mathbf{i}\theta_j}|^2, \tag{2.2}$$

where $\delta_{ij} = -2\pi g \alpha_{ij}$ with $g \in [0, \frac{1}{2}]$. Here $\alpha_{ij} = 0$ if the edge between i and j is reciprocated; $\alpha_{ij} = 1$ if the edge $i \to j$ is unreciprocated; and $\alpha_{ij} = -1$ if the edge $j \to i$ is unreciprocated. For convenience we also set $\alpha_{ij} = 0$ if i and j are

not connected. This can also be expressed as

$$\alpha_{ij} = \begin{cases} 1 & \text{if } A_{ij} = 1, A_{ji} = 0, \\ -1 & \text{if } A_{ij} = 0, A_{ji} = 1, \\ 0 & \text{otherwise.} \end{cases}$$

To understand the Definition (2.2), suppose that for a given graph we wish to choose angles that produce low frustration. Each term $W_{ij}^{(s)}|e^{\mathbf{i}\theta_i} - e^{\mathbf{i}\delta_{ij}}e^{\mathbf{i}\theta_j}|^2$ in (2.2) can make a positive contribution to the frustration if $W_{ij}^{(s)} \neq 0$; that is, if i and j are involved in at least one edge. In this case, if there is an edge from i to j that is not reciprocated, then we can force this term to be zero by choosing $\theta_j = \theta_i + 2\pi g$. If the edge is reciprocated, then we can force the term to be zero by choosing $\theta_j = \theta_i$. Hence, intuitively, choosing angles to minimize the frustration can be viewed as mapping the nodes into directed clusters on the unit circle in such a way that (a) nodes in the same cluster tend to have reciprocated connections, and (b) unreciprocated edges tend to point from source nodes in one cluster to target nodes in the next cluster, periodically. Setting the parameter $g = 1/k$ for some positive integer k indicates that we are looking for k directed clusters.

We note that, since $\delta_{ji} = -\delta_{ij}$, $W_{ij}^{(s)} = W_{ji}^{(s)}$ for $i \neq j$, and $W_{ii}^{(s)} = 0$, we may express $\eta(\boldsymbol{\theta})$ as a sum over ordered pairs with a factor of 2 appearing:

$$\eta(\boldsymbol{\theta}) = 2\sum_{i<j} W_{ij}^{(s)} \left|e^{\mathbf{i}\theta_i} - e^{\mathbf{i}\delta_{ij}}e^{\mathbf{i}\theta_j}\right|^2. \tag{2.3}$$

Indeed, in [30] the definition of frustration differs from (2.2) by having a factor of 2 in the denominator, along with a further normalization by $\sum_{ij} W_{ij}^{(s)}$. This rescaling does not affect any of the arguments that we present, and hence we omit it for simplicity.

On a real network it is unlikely that the frustration (2.2) can be reduced to zero, but it is of interest to find a set of angles that give a minimum value. This minimization problem is closely related to the *angular synchronization* problem [24, 100], which estimates angles from noisy measurements of their phase differences $\theta_i - \theta_j \mod 2\pi$. Moreover, we note that for visualization purposes it makes sense to reorder the rows and columns of the adjacency matrix based on the set of angles that minimizes the frustration. We also note that in [30] the expression in (2.2) for the frustration is normalized through a division by $2\sum_i d_i$. This is immaterial for our purposes, since that denominator is independent of the choice of $\boldsymbol{\theta}$.

The frustration (2.2) is connected to the Magnetic Laplacian, which is defined as follows, where $A \circ B$ denotes the elementwise, or Hadamard, product between matrices of the same dimension; that is,

$$(A \circ B)_{ij} = A_{ij}B_{ij}.$$

Definition 2.2.1. *Given $g \in [0, \frac{1}{2}]$, the* ***Magnetic Laplacian*** $L^{(g)}$ *[30, 31] is*

defined as

$$L^{(g)} = D - T^{(g)} \circ W^{(s)},$$

where $T^{(g)}_{ij} = e^{\mathbf{i}\delta_{ij}}$. *Here, the* transporter matrix $T^{(g)}$ *assigns a rotation to each edge according to its direction.*

From Definition 2.2.1, we can find each element in $L^{(g)}$:

$$L^{(g)}_{ij} = \begin{cases} d_i, & \text{if } i = j \\ -\frac{1}{2}e^{-\mathbf{i}2\pi g}, & \text{if } i \to j \\ -\frac{1}{2}e^{\mathbf{i}2\pi g}, & \text{if } j \to i \\ -1, & \text{if } i \leftrightarrow j \\ 0, & \text{otherwise.} \end{cases} \tag{2.4}$$

It is straightforward to see from the equation above that $L^{(g)}$ is a Hermitian matrix. When $g = 0$ and the graph is undirected, the Magnetic Laplacian reduces to the standard graph Laplacian. The following result, which is implicit in [30, 31], shows that the frustration (2.2) may be written as a quadratic form involving the Magnetic Laplacian.

Theorem 2.2.1. *Let* $\boldsymbol{\psi} \in \mathbb{C}^n$ *be such that* $\psi_j = e^{\mathbf{i}\theta_j}$, *then*

$$\boldsymbol{\psi}^H L^{(g)} \boldsymbol{\psi} = \tfrac{1}{2} \sum_{i,j} W^{(s)}_{ij} |e^{\mathbf{i}\theta_i} - e^{\mathbf{i}\delta_{ij}} e^{\mathbf{i}\theta_j}|^2. \tag{2.5}$$

Proof. Since $W^{(s)}$ is symmetric, $W^{(s)}_{ij} = W^{(s)}_{ji}$. By definition $\delta_{ij} = -\delta_{ji}$. We will use the property of complex modulus $\mid a + b\mathbf{i} \mid^2 = a^2 + b^2 = (a + b\mathbf{i})(a - b\mathbf{i})$, or

$|\ z\ |^2 = \overline{z} \cdot z$, for a complex number $z = a + b\mathbf{i}$. We can write

$$\begin{aligned}
\psi^H L^{(g)} \psi &= \psi^H (D - T^{(g)} \circ W^{(s)}) \psi \\
&= \sum_{i,j} e^{-\mathbf{i}\theta_i} (D_{ij} - e^{\mathbf{i}\delta_{ij}} W_{ij}^{(s)}) e^{\mathbf{i}\theta_j} \\
&= \sum_{i} e^{-\mathbf{i}\theta_i} d_i e^{\mathbf{i}\theta_i} - \sum_{i,j} e^{-\mathbf{i}\theta_i} e^{\mathbf{i}\delta_{ij}} W_{ij}^{(s)} e^{\mathbf{i}\theta_j} \\
&= \sum_{i,j} W_{ij}^{(s)} - \sum_{i,j} e^{-\mathbf{i}\theta_i} e^{\mathbf{i}\delta_{ij}} W_{ij}^{(s)} e^{\mathbf{i}\theta_j} \\
&= \sum_{i,j} W_{ij}^{(s)} - \frac{1}{2} (\sum_{i,j} e^{-\mathbf{i}\theta_i} e^{\mathbf{i}\delta_{ij}} W_{ij}^{(s)} e^{\mathbf{i}\theta_j} + \sum_{j,i} e^{-\mathbf{i}\theta_j} e^{\mathbf{i}\delta_{ji}} W_{ji}^{(s)} e^{\mathbf{i}\theta_i}) \\
&= \sum_{i,j} W_{ij}^{(s)} - \frac{1}{2} (\sum_{i,j} e^{-\mathbf{i}\theta_i} e^{\mathbf{i}\delta_{ij}} W_{ij}^{(s)} e^{\mathbf{i}\theta_j} + \sum_{i,j} e^{-\mathbf{i}\theta_j} e^{-\mathbf{i}\delta_{ij}} W_{ij}^{(s)} e^{\mathbf{i}\theta_i}) \\
&= \frac{1}{2} \sum_{i,j} W_{ij}^{(s)} (2 - e^{-\mathbf{i}\theta_i} e^{\mathbf{i}\delta_{ij}} e^{\mathbf{i}\theta_j} - e^{-\mathbf{i}\theta_j} e^{-\mathbf{i}\delta_{ij}} e^{\mathbf{i}\theta_i}) \\
&= \frac{1}{2} \sum_{i,j} W_{ij}^{(s)} (e^{-\mathbf{i}\theta_i} - e^{-\mathbf{i}\delta_{ij}} e^{-\mathbf{i}\theta_j})(e^{\mathbf{i}\theta_i} - e^{\mathbf{i}\delta_{ij}} e^{\mathbf{i}\theta_j}) \\
&= \frac{1}{2} \sum_{i,j} W_{ij}^{(s)} |e^{\mathbf{i}\theta_i} - e^{\mathbf{i}\delta_{ij}} e^{\mathbf{i}\theta_j}|^2
\end{aligned}$$

□

Appealing to the Rayleigh-Ritz theorem [69] the quadratic form on the left hand side of (2.5) is minimized over all $\boldsymbol{\psi} \in \mathbb{C}^n$ with $\|\boldsymbol{\psi}\| = 1$ by taking $\boldsymbol{\psi}$ to be an eigenvector corresponding to the smallest eigenvalue of the Magnetic Laplacian. Now, such an eigenvector will not generally be proportional to a vector with components of the form $\{e^{\mathbf{i}\theta_j}\}_{j=1}^n$. However, a useful heuristic is to force this relationship in a componentwise sense; that is, to assign to each θ_j the phase angle of ψ_j, effectively solving a relaxed version of the desired minimization problem. This leads to Algorithm 1 below, as used in [30].

Algorithm 1: Magnetic Laplacian algorithm

Result: Phase angles of nodes $\boldsymbol{\theta}$
Input adjacency matrix A;
Symmetrize adjacency matrix $W^{(s)} = (A + A^T)/2$;
Calculate degree matrix $D_{ii} = d_i = \sum_j W_{ij}^{(s)}$;
Construct transporter $T_{ij}^{(g)} = e^{\mathbf{i}\delta_{ij}}$;
Calculate Magnetic Laplacian $L^{(g)} = D - T^{(g)} \circ W^{(s)}$;
Compute eigenvectors $\{\psi_m^{(g)}\}_{m=1}^n = \mathrm{Eigs}(L^{(g)})$ and associated eigenvalues;
Calculate phase angles $\boldsymbol{\theta} = \mathbf{phase}(\psi_1^{(g)})$ **using eigenvector** $\psi_1^{(g)}$ **associated with the smallest eigenvalue**;
Reorder nodes with θ_i or **visualise** with $(\cos(\theta_i), \sin(\theta_i))$

2.2.4 Magnetic Laplacian and Connection Laplacian

A *connection graph* [19] is a special type of graph $G = (V, E)$ in which each edge $(i, j) \in E$ possesses an assigned weight w_{ij} and a $d \times d$ rotational matrix O_{ij} that represents the orthogonal transformation $SO(d)$ between nodes satisfying

$$O_{ij}O_{ji} = I_d.$$

Here $SO(d)$ represents the special orthogonal group of dimension d, that is, the group of all $d \times d$ matrices M that are characterized by the properties

$$M^{-1} = M^T \text{ and } \det(M) = 1.$$

The adjacency matrix A_c for a connection graph is a $nd \times nd$ matrix and is defined as follows:

$$A_c = \begin{cases} w_{ij}O_{ij} \text{ if } (i,j) \in E, \\ \mathbf{0}_d \text{ if } (i,j) \notin E. \end{cases}$$

Here, $\mathbf{0}_d$ is the zero matrix of dimension $d \times d$. The Connection Laplacian L_c is defined as the difference between a diagonal matrix D_c and the adjacency matrix A_c, expressed as:

$$L_c = D_c - A_c,$$

where D_c is a $nd \times nd$ diagonal matrix, the entries of which are given by

$$[D_c]_{ii} = d_i I_d$$

for $i \in V$, and $d_i = \sum_{(i,j)\in E} w_{ij}$. If we assign a vector $\nu_i \in \mathbb{R}^d$ to node i, and let $\boldsymbol{\nu} = (\nu_1, ..., \nu_n)$, then the top eigenvector of L_c minimizes the frustration $\eta(\boldsymbol{\nu})$ defined as follows:

$$\eta(\boldsymbol{\nu}) = \frac{\boldsymbol{\nu}^T L_c \boldsymbol{\nu}}{\boldsymbol{\nu}^T D_c \boldsymbol{\nu}} = \frac{1}{2} \frac{\sum_{i,j\in V} W_{ij} \|\nu_i - O_{ij}\nu_j\|^2}{\sum_i d_i \|\nu_i\|^2}.$$

In this case, $\eta(\nu)$ quantifies the level of inconsistency within the connection graph. A connection Laplacian is called *consistent* if, for every cycle in the graph, the total rotation is equivalent to the identity matrix. This property ensures that one can always rotate back to the same phase after completing any directed cycle [20]. Moreover, when a connection Laplacian is consistent, there exist exactly d eigenvalues with a value of 0.

The Magnetic Laplacian is a special case of connection Laplacian for directed graphs with a $U(1)$ or $SO(2)$ transformation. It utilizes the direction of each edge to construct the corresponding rotation matrix as follows:

$$O_{ij} = \begin{cases} e^{-\mathbf{i}2\pi g} \text{ if } (i,j) \in E \text{ and } (j,i) \notin E \\ e^{\mathbf{i}2\pi g} \text{ if } (j,i) \in E \text{ and } (i,j) \notin E \\ 0 \text{ otherwise .} \end{cases} \tag{2.6}$$

2.2.5 The Trophic Laplacian

The idea of discovering a linear directed hierarchy arises in many contexts where edges represent dominance or approval, including the ranking of sports teams [72] and web pages [40]. A particularly well-defined case is the quantification of trophic levels in food webs, where each directed edge represents a consumer-resource relationship [60, 66, 79]. We focus here on the approach in [73], where the aim is to assign a trophic level h_i to each node i such that along any directed edge the trophic level increases by one. This motivates the minimization of the *trophic incoherence*

$$F(\boldsymbol{h}) = \frac{\sum_{i,j} A_{ij}(h_j - h_i - 1)^2}{\sum_{i,j} A_{ij}}. \tag{2.7}$$

Denoting the total weight of node i as $\omega_i = \sum_{j \in V}(A_{ji} + A_{ij})$ and the imbalance as $\chi_i = \sum_{j \in V}(A_{ji} - A_{ij})$, the trophic level vector $\boldsymbol{h} \in \mathbb{R}^n$ that minimizes the trophic incoherence solves the linear system of equations

$$\Lambda \boldsymbol{h} = \boldsymbol{\chi}, \tag{2.8}$$

where $\Lambda = \mathrm{diag}(\boldsymbol{\omega}) - A - A^T$, and the solution to (2.8) is unique up to a constant shift [73]. Since it employs a Laplacian-style matrix, Λ, we refer to it as the *Trophic Laplacian* algorithm; see Algorithm 2.

Algorithm 2: Trophic Laplacian algorithm

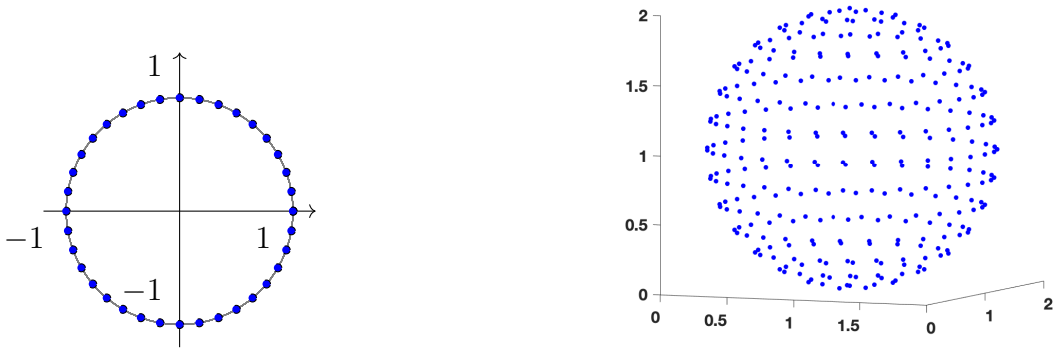
Result: The trophic levels $\boldsymbol{h}$
Input adjacency matrix A;
Calculate the node weights $\omega_i = \sum_j A_{ji} + \sum_j A_{ij}$;
Calculate the node imbalances $\chi_i = \sum_j A_{ji} - \sum_j A_{ij}$;
Calculate the Trophic Laplacian $\Lambda = \mathrm{diag}(\boldsymbol{\omega}) - A - A^T$;
Solve the linear system (2.8);
Reorder or **visualize** nodes using $\boldsymbol{h}$

2.3 Random Graph Interpretation

In this section, we associate two new random graph models with the Magnetic and Trophic Laplacian algorithms, using a similar approach to the work in [48]. After establishing these connections, we proceed as in [45] and propose a maximum likelihood test to compare the two models on a given network.

2.3.1 The Directed pRDRG Model

Given a set of phase angles $\{\theta_i\}_{i=1}^n$, we will define a model for unweighted, directed random graphs. The model generates connections between each pair of distinct nodes i and j with four possible outcomes—a pair of reciprocated edges, an unreciprocated edge from i to j, an unreciprocated edge from j to i, or no edges—



(a) (b)

Figure 2.2: (a) Points uniformly distributed on the unit circle and (b) a sphere.

as follows

$$\mathbf{P}(A_{ij} = 1, A_{ji} = 1) = f(\theta_i, \theta_j), \tag{2.9}$$
$$\mathbf{P}(A_{ij} = 1, A_{ji} = 0) = q(\theta_i, \theta_j), \tag{2.10}$$
$$\mathbf{P}(A_{ij} = 0, A_{ji} = 1) = l(\theta_i, \theta_j), \tag{2.11}$$
$$\mathbf{P}(A_{ij} = 0, A_{ji} = 0) = 1 - f(\theta_i, \theta_j) - q(\theta_i, \theta_j) - l(\theta_i, \theta_j), \tag{2.12}$$

where f, q and l are functions that define the model, and, of course, they must be chosen such that all probabilities lie between zero and one. We emphasize that this model has a feature that distinguishes it from typical random graph models, including directed Erdős–Rényi and small-world style versions [62]: the probability of the edge $i \to j$ is not independent of the probability the edge $j \to i$, in general.

We are interested here in the inverse problem where we are given a graph and a model (2.9)–(2.12), and we wish to infer the phase angles. This task arises naturally when the nodes are supplied in some arbitrary order. We will assume that the phase angles are to be assigned values from a discrete set $\{\nu_i\}_{i=1}^n$; that is, we must set $\theta_i = \nu_{p_i}$, where p is a permutation vector. This setting includes the cases of (directed) clustering and reordering. For example, with $n = 12$, we can specify $\nu_1 = \nu_2 = \nu_3 = 0$, $\nu_4 = \nu_5 = \nu_6 = \pi/2$, $\nu_7 = \nu_8 = \nu_9 = \pi$, and $\nu_{10} = \nu_{11} = \nu_{12} = 3\pi/2$, in order to assign the nodes to four directed clusters of equal size. Alternatively, $\nu_i = (i-1)2\pi/12$ would assign the nodes to equally-spaced phase angles, as shown in Figure 2.2a, as a means to reorder the graph. The following theorem shows that solving this type of inverse problem for suitable f, q and l is equivalent to minimizing the frustration.

Theorem 2.3.1. *Suppose $\boldsymbol{\theta} \in \mathbb{R}^n$ is constrained to take values such that $\theta_i = \nu_{p_i}$, where p is a permutation vector. Then minimizing the frustration $\eta(\boldsymbol{\theta})$ in (2.2) over all such $\boldsymbol{\theta}$ is equivalent to maximizing the likelihood that the graph came from*

a model of the form (2.9)–(2.12) in the case where

$$f(\theta_i,\theta_j) = \frac{1}{Z_{ij}},$$
$$q(\theta_i,\theta_j) = \frac{1}{Z_{ij}}\exp[\gamma(1-2\cos\beta_{ij}+\cos(\beta_{ij}+2\pi g))],$$
$$l(\theta_i,\theta_j) = \frac{1}{Z_{ij}}\exp[\gamma(1-2\cos\beta_{ij}+\cos(\beta_{ij}-2\pi g))],$$

with $\beta_{ij} = \theta_i - \theta_j$ *and normalization constant*

$$Z_{ij} = 1 + e^{\gamma(1-2\cos\beta_{ij}+\cos(\beta_{ij}+2\pi g))} + e^{\gamma(1-2\cos\beta_{ij}+\cos(\beta_{ij}-2\pi g))} + e^{\gamma(2-2\cos\beta_{ij})},$$

for any positive constant γ.

Proof. We first note that, since $\delta_{ji} = -\delta_{ij}$, $W^{(s)}_{ij} = W^{(s)}_{ji}$ for $i \neq j$, and $W^{(s)}_{ii} = 0$, we may express $\eta(\boldsymbol{\theta})$ (2.2) in terms of a sum over ordered pairs:

$$\tfrac{1}{2}\,\eta(\boldsymbol{\theta}) = \sum_{i<j} W^{(s)}_{ij}\left|e^{\mathbf{i}\theta_i} - e^{\mathbf{i}\delta_{ij}}e^{\mathbf{i}\theta_j}\right|^2. \tag{2.13}$$

Then, distinguishing between the three different ways in which each i and j may be connected, we have

$$\tfrac{1}{2}\,\eta(\boldsymbol{\theta}) = \sum_{i<j:A_{ij}=1,A_{ji}=1} |e^{\mathbf{i}\theta_i} - e^{\mathbf{i}\theta_j}|^2 + \sum_{i<j:A_{ij}=1,A_{ji}=0} \tfrac{1}{2}|e^{\mathbf{i}\theta_i} - e^{-\mathbf{i}2\pi g}e^{\mathbf{i}\theta_j}|^2 \tag{2.14}$$
$$+ \sum_{i<j:A_{ij}=0,A_{ji}=1} \tfrac{1}{2}|e^{\mathbf{i}\theta_i} - e^{\mathbf{i}2\pi g}e^{\mathbf{i}\theta_j}|^2. \tag{2.15}$$

The likelihood L of the graph G from a model of the form (2.9)–(2.12) is given by

$$L(G) = \prod_{i<j:A_{ij}=1,A_{ji}=1} f(\theta_i,\theta_j) \prod_{i<j:A_{ij}=1,A_{ji}=0} q(\theta_i,\theta_j) \prod_{i<j:A_{ij}=0,A_{ji}=1} l(\theta_i,\theta_j)$$
$$\times \prod_{i<j:A_{ij}=0,A_{ji}=0} \left(1 - f(\theta_i,\theta_j) - q(\theta_i,\theta_j) - l(\theta_i,\theta_j)\right),$$

which we may rewrite as

$$\begin{aligned} L(G) = &\prod_{i<j:A_{ij}=1,A_{ji}=1} \frac{f(\theta_i,\theta_j)}{1-f(\theta_i,\theta_j)-q(\theta_i,\theta_j)-l(\theta_i,\theta_j)} \\ &\times \prod_{i<j:A_{ij}=1,A_{ji}=0} \frac{q(\theta_i,\theta_j)}{1-f(\theta_i,\theta_j)-q(\theta_i,\theta_j)-l(\theta_i,\theta_j)} \\ &\times \prod_{i<j:A_{ij}=0,A_{ji}=1} \frac{l(\theta_j,\theta_i)}{1-f(\theta_i,\theta_j)-q(\theta_i,\theta_j)-l(\theta_i,\theta_j)} \\ &\times \prod_{i<j} \left(1-f(\theta_i,\theta_j)-q(\theta_i,\theta_j)-l(\theta_i,\theta_j)\right). \end{aligned}$$

The final factor on the right hand side, which is the probability of the null graph, takes the same value for any $\boldsymbol{\theta} \in \mathbb{R}^n$ such that $\theta_i = \nu_{p_i}$, since each ordered pair of arguments appears exactly once. We may therefore ignore this factor when maximizing the likelihood. Then, taking the logarithm and negating, we see that maximizing the likelihood is equivalent to minimizing the expression

$$\sum_{i<j:A_{ij}=1,A_{ji}=1} \ln\left[\frac{1-f(\theta_i,\theta_j)-q(\theta_i,\theta_j)-l(\theta_i,\theta_j)}{f(\theta_i,\theta_j)}\right] \tag{2.16}$$

$$+ \sum_{i<j:A_{ij}=1,A_{ji}=0} \ln\left[\frac{1-f(\theta_i,\theta_j)-q(\theta_i,\theta_j)-l(\theta_i,\theta_j)}{q(\theta_i,\theta_j)}\right] \tag{2.17}$$

$$+ \sum_{i<j:A_{ij}=0,A_{ji}=1} \ln\left[\frac{1-f(\theta_i,\theta_j)-q(\theta_i,\theta_j)-l(\theta_i,\theta_j)}{l(\theta_i,\theta_j)}\right]. \tag{2.18}$$

Comparing terms in (2.16)–(2.18) and (2.14)–(2.15) we see that the two minimization problems are equivalent if

$$\begin{aligned} \ln\left[\frac{1-f(\theta_i,\theta_j)-q(\theta_i,\theta_j)-l(\theta_i,\theta_j)}{f(\theta_i,\theta_j)}\right] &= \gamma\left|e^{\mathbf{i}\theta_i}-e^{\mathbf{i}\theta_j}\right|^2 \\ &= \gamma(2-2\cos(\theta_i-\theta_j)), \\ \ln\left[\frac{1-f(\theta_i,\theta_j)-q(\theta_i,\theta_j)-l(\theta_i,\theta_j)}{q(\theta_i,\theta_j)}\right] &= \frac{\gamma}{2}\left|e^{\mathbf{i}\theta_i}-e^{-\mathbf{i}2\pi g}e^{\mathbf{i}\theta_j}\right|^2 \\ &= \gamma(1-\cos(\theta_i-\theta_j+2\pi g)), \\ \ln\left[\frac{1-f(\theta_i,\theta_j)-q(\theta_i,\theta_j)-l(\theta_i,\theta_j)}{l(\theta_i,\theta_j)}\right] &= \frac{\gamma}{2}\left|e^{\mathbf{i}\theta_i}-e^{\mathbf{i}2\pi g}e^{\mathbf{i}\theta_j}\right|^2 \\ &= \gamma(1-\cos(\theta_i-\theta_j-2\pi g)), \end{aligned}$$

where we may choose any positive constant γ since the minimization problems are scale invariant. Solving for f, q and l as functions of θ_i and θ_j we arrive at the model in the statement of the theorem. □

For the model in Theorem 2.3.1, the probability of an edge from node i to node j depends on the phase difference $\beta_{ij} = \theta_i - \theta_j$, the decay rate γ, and the

parameter g. We see that γ determines how rapidly the edge probability varies with the phase difference. In the extreme case when $\gamma = 0$, we obtain $f(\theta_i, \theta_j) = q(\theta_i, \theta_j) = l(\theta_i, \theta_j) = 1/4$, and thus the model reduces to a conditional Erdős–Rényi form. In addition, as γ increases the graph generally becomes more sparse. This is because the likelihood of disconnection, $\exp[2\gamma(1 - \cos(\theta_i - \theta_j))]/Z_{ij}$, is greater than or equal to that of the other cases.

We note that having applied the Magnetic Laplacian algorithm to estimate $\boldsymbol{\theta}$, there are two straightforward approaches to estimating γ. One way is to maximize the graph likelihood over $\gamma > 0$. Another is to choose γ so that the expected edge density from the random graph model matches the edge density of the given network. We illustrate these approaches in Section 2.4.

Remark 2.3.1. *Since the edge probabilities are functions of the phase differences and have a periodicity of* 2π*, this model resembles the* periodic Range-Dependent Random Graph *(pRDRG) model in [45], which generates an undirected edge between* i *and* j *with probability* $f(\min\{|j-i|, n-|j-i|\})$ *for a given decay function* f*. We will therefore use the term* directed periodic Range-Dependent Random Graph *model (directed pRDRG) to describe the model in Theorem 2.3.1.*

2.3.2 The Trophic Range-dependent Model

Now, given a set of trophic levels $\{h_i\}_{i=1}^n$, we define an unweighted, directed random graph model where

$$\mathbf{P}(A_{ij} = 1) = f(h_i, h_j), \tag{2.19}$$
$$\mathbf{P}(A_{ij} = 0) = 1 - f(h_i, h_j), \tag{2.20}$$

for some function f. Here, the probability of an edge $i \to j$ is independent of the probability of the edge $j \to i$.

Following our treatment of the directed pRDRG case, we are now interested in the inverse problem where we are given a graph and the model (2.19)–(2.20), and we wish to infer the trophic levels. We will assume that the trophic levels are to be assigned values from a discrete set $\{\nu_i\}_{i=1}^n$; that is, we must set $h_i = \nu_{p_i}$, where p is a permutation vector. This setting includes the cases of assignment of nodes to trophic levels of specified size; for example, with $n = 12$, we can set $\nu_1 = \nu_2 = \nu_3 = 1$, $\nu_4 = \nu_5 = \nu_6 = 2$, $\nu_7 = \nu_8 = \nu_9 = 3$, and $\nu_{10} = \nu_{11} = \nu_{12} = 4$, in order to assign the nodes to four equal levels. Alternatively, $\nu_i = i$ would assign each node to its own level, which is equivalent to reordering the nodes. The following theorem shows that solving this type of inverse problem for suitable f is equivalent to minimizing the trophic incoherence.

Theorem 2.3.2. *Suppose* $\boldsymbol{h} \in \mathbb{R}^n$ *is constrained to take values such that* $h_i = \nu_{p_i}$*, where* p *is a permutation vector. Then minimizing the trophic incoherence* $F(\boldsymbol{h})$ *in (2.7) over all such* $\boldsymbol{h}$ *is equivalent to maximizing the likelihood that the graph came from a model of the form (2.19)–(2.20) in the case where*

$$f(h_i, h_j) = \frac{1}{1 + e^{\gamma(h_j - h_i - 1)^2}}$$

for any positive γ.

Proof. Noting that the denominator in (2.7) is independent of the choice of $\boldsymbol{h}$, this result is a special case of Theorem 2.3.4 below, with $I(h_i, h_j) = (h_j - h_i - 1)^2$. □

For the model in Theorem 2.3.2, the probability of an edge $i \to j$ is a function of the shifted, directed, squared difference in levels, $(h_j - h_i - 1)^2$. The larger this value, the lower the probability. Within the same level, where $h_i = h_j$, the probability is $1/(1 + e^{\gamma})$. The edge probability takes its maximum value of $1/2$ when $h_j - h_i = 1$, that is, when the edge starts at one level and finishes at the next highest level. We also see that the overall expected edge density is always smaller than $1/2$. Across different levels, where $h_i \neq h_j$, the edge $i \to j$ and the edge $j \to i$ are not generated with the same probability. If $|h_j - h_i - 1| < |h_i - h_j - 1|$, the edge $i \to j$ is more likely than $j \to i$. The two edge probabilities are equal if and only if $h_i = h_j$. Therefore, this model can be interpreted as a combination of an Erdős–Rényi model within the same level and a periodic range-dependent model across different levels.

The parameter γ controls the decay rate of the likelihood as the shifted, directed, squared difference in levels increases. When $h_j - h_i = 1$, γ plays no role. If $\gamma = 0$, the model reduces to Erdős–Rényi with an edge probability of $1/2$. As $\gamma \to \infty$, the edge probability tends to zero if $h_j - h_i \neq 1$. In this case, the model will generate a multipartite graph where edges are only possible in one direction between adjacent levels, and this happens with probability $1/2$. As mentioned previously in Section 2.3.1 and illustrated in Section 2.4, γ can be fitted from a maximum likelihood estimate or by matching the edge density.

We note that the definition of trophic incoherence in (2.7) and the resulting Trophic Laplacian algorithm make sense for a non-negatively weighted graph, in which case we have the following result. Here, to be concrete we assume that weights lie strictly between zero and one. Similar results can be obtained for weights from a discrete distribution.

Theorem 2.3.3. *Suppose $\boldsymbol{h} \in \mathbb{R}^n$ is constrained to take values such that $h_i = \nu_{p_i}$, where p is a permutation vector. Then minimizing the trophic incoherence $F(\boldsymbol{h})$ in (2.7) over all such $\boldsymbol{h}$ for a weighted graph with weights in $(0, 1)$ is equivalent to maximizing the likelihood that the graph came from a model where each edge weight A_{ij} is independent with density function*

$$f_{ij}(x) := \frac{1}{Z_{ij} e^{\gamma x (h_j - h_i - 1)^2}} \text{ for } x \in (0, 1), \quad \text{and } f(x) = 0 \text{ otherwise}, \tag{2.21}$$

for any positive γ, where $Z_{ij} = \frac{1 - e^{-\gamma (h_j - h_i - 1)^2}}{\gamma (h_j - h_i - 1)^2}$ is a normalization factor.

Proof. This is a special case of Theorem 2.3.5 below, where $I(h_i, h_j) = (h_j - h_i - 1)^2$. □

2.3.3 Generalised Random Graph Model

The results in Sections 2.3.1 and 2.3.2 exploit the form of the objective function: the sum over all edges of a kernel function can be viewed as the sum of

log-likelihoods. This shows that the minimization problem is equivalent to maximizing the likelihood of an associated random graph model, in the setting where we assign nodes to a discrete set of scalar values. The restriction to discrete values is used in the proofs to make the probability of the null graph constant. However, we emphasize that in practice the relaxed versions of the optimization problems, which are solved by the two algorithms, do not have this restriction. The Magnetic Laplacian algorithm produces real-valued phase angles and the Trophic Laplacian algorithm produces real-valued trophic levels.

We may extend the connection in Theorem 2.3.2 to the case of higher dimensional node attributes, that is, where we wish to associate each node with a discrete vector from a set $\{\boldsymbol{\nu}^{[k]}\}_{k=1}^{n}$, where each $\boldsymbol{\nu}^{[k]} \in \mathbb{R}^d$ for some $d \geq 1$. This setting arises, for example, if we wish to visualize the network in higher dimension; a natural extension of the ring structure would be to place nodes at regularly spaced points on the surface of the unit sphere, see Figure 2.2b, which we produced with the algorithm in [29]. The next result generalizes Theorem 2.3.2 to this case.

Theorem 2.3.4. *Suppose we have an unweighted directed graph with adjacency matrix A and a kernel function $I : \mathbb{R}^d \times \mathbb{R}^d \to \mathbb{R}_+$, and suppose that we are free to assign elements $\{\boldsymbol{h}^{[k]}\}_{k=1}^{n}$ to values from the set $\{\boldsymbol{\nu}^{[k]}\}_{k=1}^{n}$; that is, we allow $\boldsymbol{h}^{[k]} = \boldsymbol{\nu}^{[p_k]}$ where p is a permutation vector. Then minimizing*

$$\sum_{i,j} A_{ij} I(\boldsymbol{h}^{[i]}, \boldsymbol{h}^{[j]}) \tag{2.22}$$

over all such $\{\boldsymbol{h}^{[k]}\}_{k=1}^{n}$ is equivalent to maximizing the likelihood that the graph came from a model where the (independent) probability of the edge $i \to j$ is

$$f(\boldsymbol{h}^{[i]}, \boldsymbol{h}^{[j]}) = \frac{1}{1 + e^{\gamma I(\boldsymbol{h}^{[i]}, \boldsymbol{h}^{[j]})}}, \tag{2.23}$$

for any positive γ.

Proof. Given $\{\boldsymbol{h}^{[k]}\}_{k=1}^{n}$, the probability of generating a graph G from the model stated in the theorem is

$$\begin{aligned} L(G) &= \prod_{i,j:A_{ij}=1} f(\boldsymbol{h}^{[i]}, \boldsymbol{h}^{[j]}) \prod_{i,j:A_{ij}=0} \left(1 - f(\boldsymbol{h}^{[i]}, \boldsymbol{h}^{[j]})\right) \\ &= \prod_{i,j:A_{ij}=1} \frac{f(\boldsymbol{h}^{[i]}, \boldsymbol{h}^{[j]})}{1 - f(\boldsymbol{h}^{[i]}, \boldsymbol{h}^{[j]})} \prod_{i,j} \left(1 - f(\boldsymbol{h}^{[i]}, \boldsymbol{h}^{[j]})\right). \end{aligned}$$

The second factor on the right hand side, the probability of the null graph, does not depend on the choice of $\{\boldsymbol{h}^{[k]}\}_{k=1}^{n}$. So we may ignore this factor, and after taking logs and negating we arrive at the equivalent problem of minimizing

$$\sum_{i,j:A_{ij}=1} \ln \left[\frac{1 - f(\boldsymbol{h}^{[i]}, \boldsymbol{h}^{[j]})}{f(\boldsymbol{h}^{[i]}, \boldsymbol{h}^{[j]})} \right]. \tag{2.24}$$

Comparing (2.24) and (2.22), we see that two minimization problems have the same solution when

$$\ln\left[\frac{1-f(\boldsymbol{h}^{[i]},\boldsymbol{h}^{[j]})}{f(\boldsymbol{h}^{[i]},\boldsymbol{h}^{[j]})}\right]=\gamma I(\boldsymbol{h}^{[i]},\boldsymbol{h}^{[j]}),$$

for any positive γ, and the result follows. □

For the model in Theorem 2.3.4, given $\{\boldsymbol{h}^{[k]}\}_{k=1}^n$ the edge $i\to j$ appears according to a Bernoulli distribution with probability $f(\boldsymbol{h}^{[i]},\boldsymbol{h}^{[j]})$, and hence with variance

$$f(\boldsymbol{h}^{[i]},\boldsymbol{h}^{[j]})[1-f(\boldsymbol{h}^{[i]},\boldsymbol{h}^{[j]})]=\frac{e^{\gamma I(\boldsymbol{h}^{[i]},\boldsymbol{h}^{[j]})}}{[1+e^{\gamma I(\boldsymbol{h}^{[i]},\boldsymbol{h}^{[j]})}]^2}.$$

When $I(\boldsymbol{h}^{[i]},\boldsymbol{h}^{[j]})=0$ the probability is $1/2$ and the variance takes its largest value, $1/4$. The edge probability is symmetric about i and j if and only if the function I is symmetric about its arguments. In the case of squared Euclidean distance, $I(\boldsymbol{h}^{[i]},\boldsymbol{h}^{[j]})=\|\boldsymbol{h}^{[i]}-\boldsymbol{h}^{[j]}\|^2$, and an undirected graph, the relaxed version of the minimization problem is solved by taking d eigenvectors corresponding to the smallest eigenvalues of the standard graph Laplacian.

For completeness, we now state and prove a weighted analogue of Theorem 2.3.4 assuming that weights lie strictly between zero and one. Discrete-valued weights may be dealt with similarly.

Theorem 2.3.5. *Suppose $\{\boldsymbol{h}^{[k]}\}_{k=1}^n$ may take values from the given set $\{\boldsymbol{\nu}^{[k]}\}_{k=1}^n$; that is, $\boldsymbol{h}^{[k]}=\boldsymbol{\nu}^{[p_k]}\in\mathbb{R}^d$, where p is a permutation vector. Then, given a weighted graph with weights in $(0,1)$, minimizing the expression (2.22) over all such $\{\boldsymbol{h}^{[k]}\}_{k=1}^n$ is equivalent to maximizing the likelihood that the graph came from a model where A_{ij} has (independent) density*

$$f_{ij}(x)=\frac{1}{Z_{ij}e^{\gamma x I(\boldsymbol{h}^{[i]},\boldsymbol{h}^{[j]})}},\ \text{for } x\in(0,1),\quad \text{and } f(x)=0 \text{ otherwise},\tag{2.25}$$

for any positive γ, where

$$Z_{ij}=\frac{1-e^{-\gamma I(\boldsymbol{h}^{[i]},\boldsymbol{h}^{[j]})}}{\gamma I(\boldsymbol{h}^{[i]},\boldsymbol{h}^{[j]})}$$

is a normalization factor.

Proof. It is straightforward to check that the normalization factor Z_{ij} ensures

$$\int_{y=0}^{1} f_{ij}(y)\,dy=1.$$

Now the product over all pairs $\prod_{i,j}Z_{ij}$ is independent of the choice of permutation vector p. Hence, under the model defined in the theorem, maximizing the likelihood of the graph G is equivalent to maximizing $\prod_{i,j}f_{ij}(A_{ij})$. After tak-

ing logarithms and negating, we see that the choice (2.25) allows us to match (2.22). □

Remark 2.3.2. *It is natural to ask whether the frustration (2.2) fits into the form (2.22), and hence has an associated random graph model of the form (2.23). We see from (2.13) that the frustration may be written*

$$\eta(\boldsymbol{\theta}) = \sum_{i,j} A_{ij} |e^{\mathbf{i}\theta_i} - e^{\mathbf{i}\delta_{ij}} e^{\mathbf{i}\theta_j}|^2.$$

However, the factor $|e^{\mathbf{i}\theta_i} - e^{\mathbf{i}\delta_{ij}} e^{\mathbf{i}\theta_j}|^2$ depends (through δ_{ij}) on A_{ij}, and hence we do not have expression of the form (2.22). This explains why a new type of model, with conditional dependence between the $i \to j$ and $j \to i$ connections, was needed for Theorem 2.3.1.

2.3.4 Model Comparison

The random graph models appearing in Section 2.3 capture the characteristics of linear and periodic directed hierarchies. Hence it may be of interest (a) to analyze properties of these models and (b) to use these models to evaluate the performance of computational algorithms. However, in the remainder of this chapter we focus on a follow-on topic of more direct practical significance. The Magnetic Laplacian and Trophic Laplacian algorithms allow us to compute node attributes $\boldsymbol{\theta}$ and $\boldsymbol{h}$ in $\mathbb{R}^n$ for a given graph, leading to unsupervised node ordering. The main computation required in this step is finding dominant eigenvector-eigenvalue pairs. Assuming that the network is sparse (each node has an $O(1)$ degree) and that the power method gives the required accuracy in a finite number of iterations, this is an $\mathcal{O}(n)$ computation. Motivated by Theorems 2.3.1 and 2.3.2, we may then compute the likelihood of the graph for this choice of attributes, which has a complexity of $\mathcal{O}(n^2)$. By comparing likelihoods we may quantify which underlying structure is best supported by the data. An extra consideration is that both random graph models involve a free parameter, $\gamma > 0$, which is needed to evaluate the likelihood. As discussed earlier, one option is to fit γ to the data, for example by matching the expected edge density from the model with the edge density of the given graph. However, based on our computational tests, we found that a more reliable approach was to choose the γ that maximizes the likelihood, once the node attributes were available; see Sections 2.4 and 2.5 for examples. Our overall proposed workflow for model comparison is summarized in Algorithm 3.

Algorithm 3: Model Comparison

Result: Comparison of possible graph structures
Input adjacency matrix A;
for *Candidate spectral methods* **do**
  Compute node attributes (in our case with Magnetic and Trophic Laplacian algorithms);
  Derive the associated random graph model ;
  Calculate maximum likelihood over $\gamma > 0$;
end
Report or compare maximum likelihoods

2.4 Results on Synthetic Networks

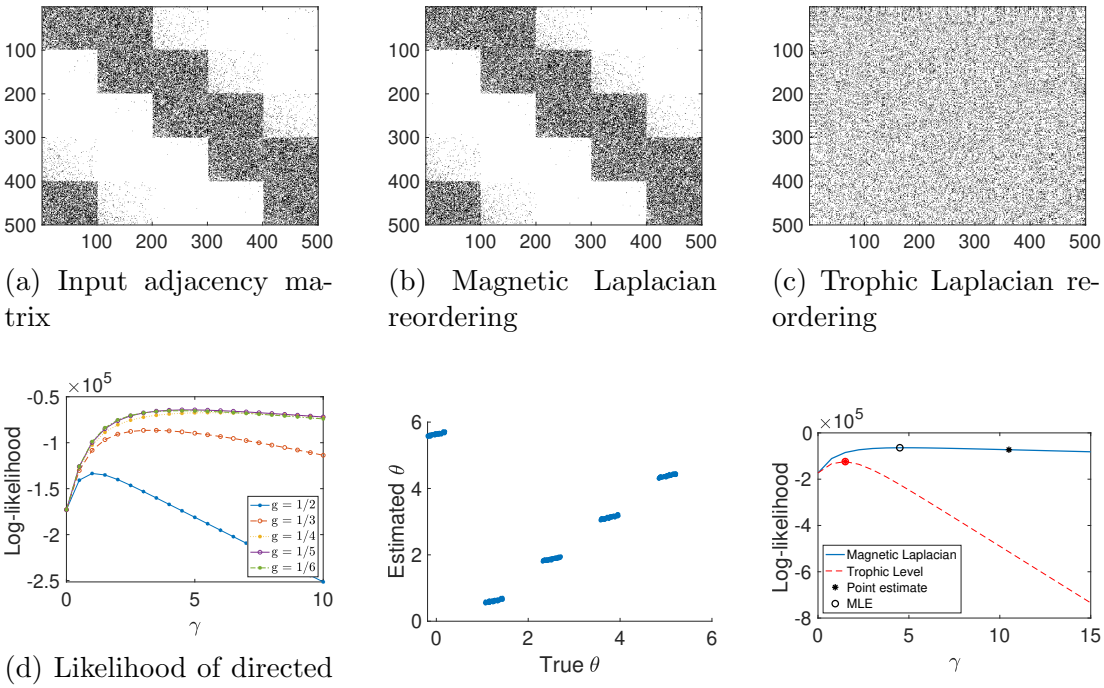
In this section, we demonstrate the model comparison workflow on synthetic networks. These networks are generated using the directed pRDRG model and the trophic RDRG model. Hence, we have a "ground truth" concerning whether a network is more linear or periodic. Note that the Magnetic Laplacian algorithm and associated random graph model have a parameter g that controls the spacing between clusters. Therefore, when using the Magnetic Laplacian algorithm our first step is to select the parameter g based on the maximum likelihood of the graph.

2.4.1 Directed pRDRG Model

We generate a synthetic network using the directed pRDRG model with K clusters of size m, and hence $n = mK$ nodes. An array of angles $\boldsymbol{\theta} \in \mathbb{R}^n$ is created, forming evenly spaced clusters $C_1, C_2, ..., C_K$. This is achieved by letting $\theta_i = \frac{2\pi(l-1)}{K} + \sigma$ if $i \in C_l$, where $\sigma \sim \text{unif}(-a, a)$ is added noise. We then construct the adjacency matrix according to the probabilities in Theorem 2.3.1 with $g = 1/K$. We choose $m = 100$, $K = 5$, $\gamma = 5$ and $a = 0.2$ and the corresponding adjacency matrix is shown in Figure 2.3a.

The Magnetic Laplacian algorithm is then applied to the adjacency matrix to estimate phase angles and reorder the nodes. The reordered adjacency matrix (Figure 2.3b) recovers the original structure. The Trophic Laplacian algorithm is also applied to estimate the trophic level of each node. Figure 2.3c shows the adjacency matrix reordered by the estimated trophic levels, which hides the original pattern. Intuitively, the Trophic Laplacian algorithm is unable to distinguish between these nodes since there is no clear "lowest" or "highest" level among the directed clusters.

Figure 2.3d illustrates how the optimal parameter g is selected. The plots show the likelihood that the network is generated by a directed pRDRG model for $g = \frac{1}{2}, \frac{1}{3}, \frac{1}{4}, \frac{1}{5}, \frac{1}{6}$, assuming we are interested in structures with at most 6 directed clusters. We see that $g = \frac{1}{5}$ has the highest maximum likelihood, as expected. Consequently, we choose $g = 1/5$ for the Magnetic Laplacian algorithm. In addition for this value of g we plot in Figure 2.3e the phase angles estimated with the Magnetic Laplacian algorithm against the true phase angles. The linear



(a) Input adjacency matrix

(b) Magnetic Laplacian reordering

(c) Trophic Laplacian reordering

(d) Likelihood of directed pRDRG

(e) Estimated θ

(f) Model comparison

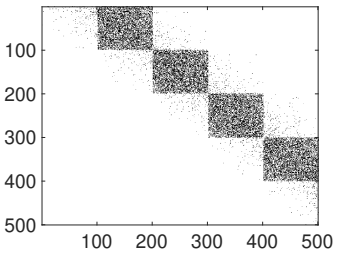
Figure 2.3: Magnetic Laplacian and Trophic Laplacian algorithms applied to a synthetic directed pRDRG

relationship confirms that the algorithm recovers the 5 clusters in the presence of noise.

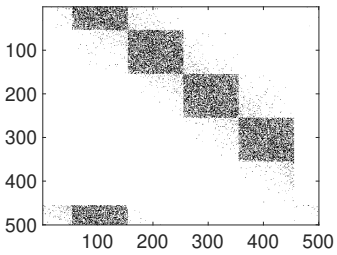
We finally in Figure 2.3f compare the likelihood of a directed pRDRG against the likelihood of a trophic RDRG. Both likelihoods are calculated using several test points for γ. The highest points are highlighted with circles and they correspond to the maximum likelihood estimators (MLE) for γ. Not surprisingly, in this case the Magnetic Laplacian algorithm achieves a higher maximum. Asterisks highlight the point estimates arising when the expected number of edges is matched to the actual number of edges. We see here, and also observed in similar experiments, that the maximum likelihood estimate for γ produces a more accurate result. We also found (numerical experiments not presented here) that the accuracy of both types of γ estimates improves as n increases when using the Magnetic Laplacian algorithm.

2.4.2 The Trophic RDRG model

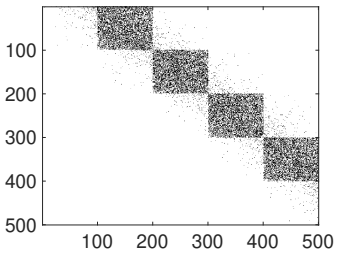
Following on from the previous subsection, we now generate synthetic data by simulating the trophic RDRG model with levels $C_1, C_2, \ldots, C_K$, where each level has m nodes. In particular, we generate an array of trophic indices $\boldsymbol{h} \in \mathbb{R}^n$, where the total number of nodes is $n = mK$. We let $h_i = l + \sigma$ if $i \in C_l$ for $1 \leq l \leq K$, where $\sigma \sim \text{unif}(-a, a)$ is added noise. The edges are then generated according to the probabilities in Theorem 2.3.2. In the following example we use $K = 5$, $m = 100$, $a = 0.2$ and $\gamma = 5$. This generates a network with 5 clusters forming a linear directed flow, as shown in Figure 2.4a.



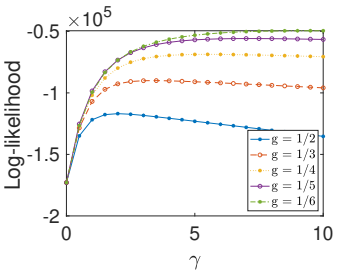
(a) Input adjacency matrix



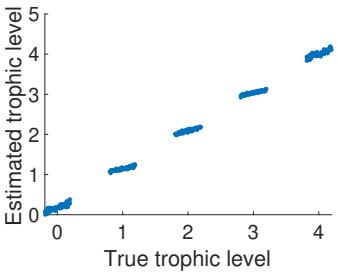
(b) Magnetic Laplacian reordering



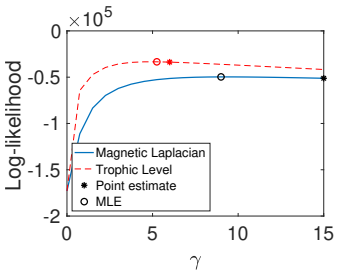
(c) Trophic Laplacian reordering



(d) Likelihood of directed pRDRG



(e) Estimated trophic level



(f) Model comparison

Figure 2.4: Magnetic Laplacian and Trophic Laplacian algorithms applied to a synthetic trophic RDRG

We see in Figure 2.4c that the Trophic Laplacian algorithm recovers the underlying pattern. Figure 2.4b shows that the Magnetic Laplacian algorithm also gives adjacent locations to nodes in the same cluster, and places the clusters in order, modulo a "wrap-around" effect that arises due to its periodic nature. Figure 2.4d suggests that the optimal Magnetic Laplacian parameter is $g = 1/6$. For this case, it is reasonable that $g = 1/K$ is not identified, since the disconnection between the first and the last cluster contradicts the structure of the directed pRDRG model.

The trophic levels estimated using the Trophic Laplacian are consistent with the true trophic levels, as shown by the linear pattern in Figure 2.4e. As expected, the Trophic Laplacian produces a higher maximum likelihood for this network (Figure 2.4f) and a more accurate MLE and point estimate for γ. We observe (in similar experiments not presented here) that when using the Trophic Laplacian, the accuracy of both estimates increase using the Trophic Laplacian.

2.5 Results on Real Networks

We now discuss practical use cases for the model comparison tool on a range of real networks. We emphasize that the tool is not designed to discover whether a given directed network has linear or directed hierarchical structure; rather it aims to quantify which of the two structures is best supported by the data in a relative sense. Since both models under investigation assume no self-loops, we discard these if they are present in the data. Following common practice, we also prepro-

cess by retaining the largest strongly connected component to emphasize directed cycles. This ensures that any pair of nodes can be connected through a sequence of directed edges. However, when the strongly connected component contains too few nodes, we analyze the largest weakly connected component instead.

We give details on four networks, covering examples of the two cases where linear and periodic structure dominate. For the first two networks, we show network visualizations to illustrate the results further. In subsection 2.5.5 we present summary results over 15 networks.

2.5.1 Food Web

In the Florida Bay food web[2][65], nodes are components of the system, and unweighted directed edges represent carbon transfer from the source nodes to the target nodes [106], which usually means that the latter feed on the former. Besides organisms, the nodes also contain non-living components, such as carbon dissolved in the water column. Since we are more interested in the relationship between organisms, we remove those non-living components from the network. We analyze the largest strongly connected component of the network, which comprises 12 nodes and 28 edges.

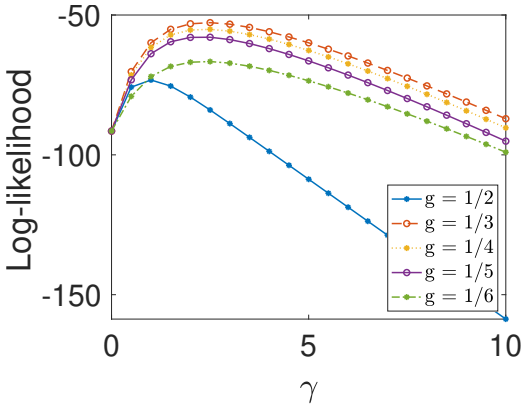
We estimate the phase angles of each node using the Magnetic Laplacian algorithm based on the optimal choice $g = 1/3$ (Figure 2.5a). Figure 2.5b compares the likelihood of the food web being generated by the directed pRDRG model with the likelihood of it being generated by the trophic RDRG model, as γ varies. The directed pRDRG model achieves a higher maximum likelihood, suggesting that the structure is more periodic than linear. In Figure 2.5c, the heights of the nodes correspond to their estimated trophic levels on a vertical axis. We see that 22 edges point upwards, these are shown in blue. There are 6 downward edges, highlighted in red, which violate the trophic structure. The Magnetic Laplacian mapping in Figure 2.5d arranges 26 edges in a counterclockwise direction, shown in blue, with 2 edges, shown in red, violating the structure and pointing in the reverse orientation.

With $g = 1/3$, the Magnetic Laplacian mapping is encouraging cycles in the food chain, and these are visible in Figure 2.5d, notably between members of three categories: (i) flatfish and other demersal fishes; (ii) lizardfish and eels; and (iii) toadfish and brotalus. Another noticeable distinction is that the Magnetic Laplacian mapping positions eels close to lizardfish, and flatfish near other demersal fishes by accounting for the reciprocal edges, while the Trophic Laplacian mapping places them further apart. In Figures 2.5e and 2.5f we show the reordered adjacency matrix arising from the two algorithms.

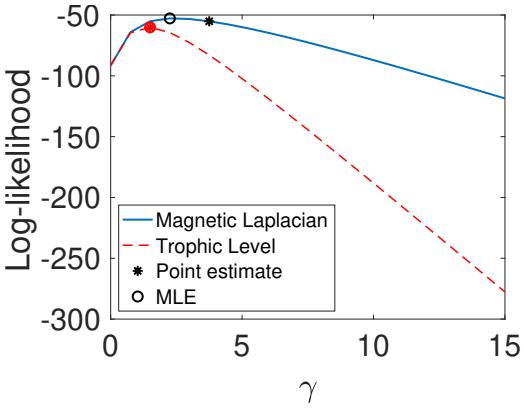
2.5.2 Influence Matrix

The influence matrix we study quantifies the influence of selected system factors in the Motueka Catchment of New Zealand [22]. The original influence matrix consists of integer scores between 0 and 5, measuring to what extent the row

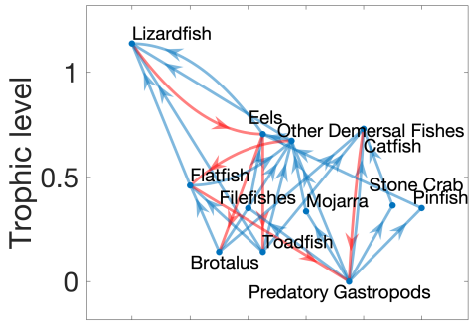
[2] https://snap.stanford.edu/data/Florida-bay.html



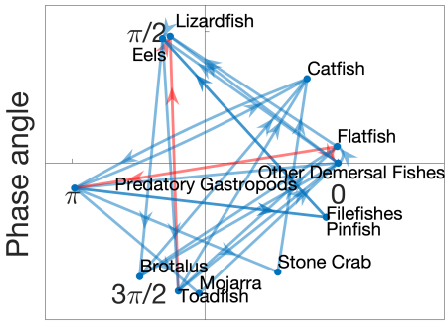
(a) Likelihood of directed pRDRG



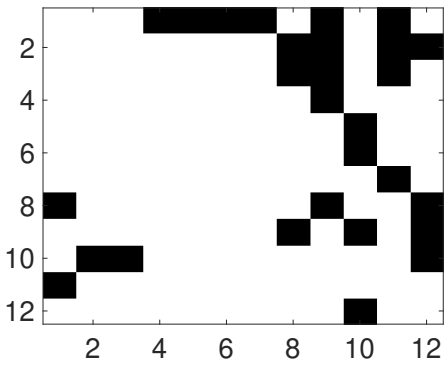
(b) Model comparison



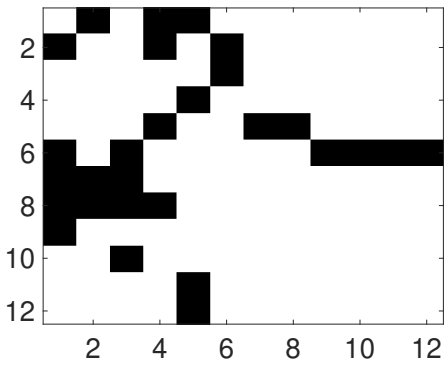
(c) Estimated trophic level



(d) Magnetic Eigenmap



(e) Trophic Laplacian reordering



(f) Magnetic Laplacian reordering

Figure 2.5: Results for the Florida Bay food web

factors influence the column factors, where a bigger value represents a stronger impact. The system factors and influence scores were developed by pooling the views of local residents. To convert to an unweighted network, we binarise the weights by keeping only the edges between each factor and the factor(s) it influences most strongly. We then select the largest strongly connected component, which comprises 14 nodes and 35 edges.

The optimal parameter for the Magnetic Laplacian is $g = 1/4$ (Figure 2.6a). The mapping from the Magnetic Laplacian has a higher maximum likelihood than the Trophic Laplacian mapping, indicating a more periodic structure (Figure 2.6b). The Trophic Laplacian mapping in Figure 2.6c aims to reveal a hierarchical influence structure. Here, scientific research and economic inputs are assigned lower trophic levels, suggesting that they are the fundamental influencers. The labour market is placed at the top, indicating that it tends to be influenced by other factors. However, there are 8 edges, highlighted in red, that point downwards, violating the directed linear structure.

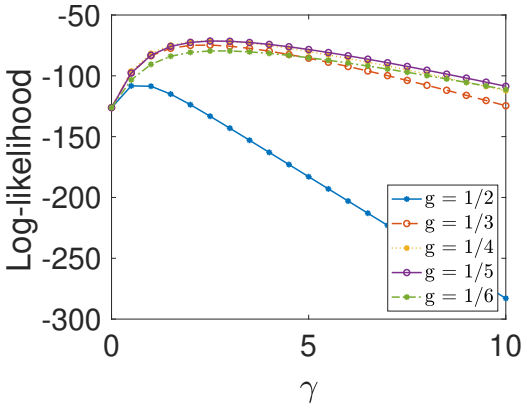
On the other hand, the Magnetic Laplacian mapping in Figure 2.6d aims to reveal four directed clusters with phase angles of approximately $0, \pi/2, \pi, 3\pi/2$. We highlight the nodes corresponding to ecological factors in red and social-economic factors in blue. The cluster near $\pi/2$ with 6 nodes contains a combination of ecological and social-economic factors, and includes 6 reciprocal edges between ecological factors and social-economic factors. Adjacency matrix reorderings are shown in Figures 2.6e and 2.6f. Overall, the pattern agrees with the conceptual schematic model proposed in [22, Figure 5], which we have reproduced in Figure 2.7. This model posits that ecological factors exert influence on social-economic factors, which in turn influence on ecological factors, while the ecological system also influences itself.

2.5.3 Yeast Transcriptional Regulation Network

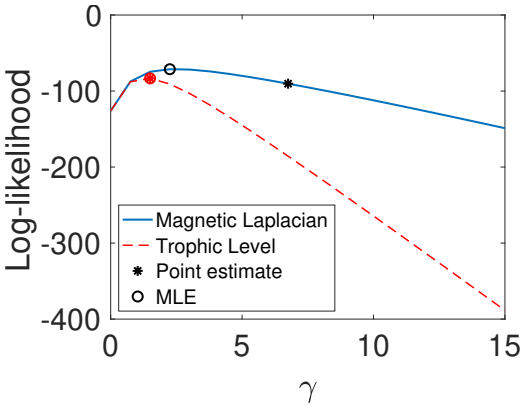
We now analyze a gene transcriptional regulation network[3][65] for a type of yeast called *S. cerevisiae* [77], where a node represents an operon made up of a group of genes in mRNA. An edge from operon i to j indicates that the transcriptional factor encoded by j regulates i. The original network is directed and signed, with signs indicating activation and deactivation. Here we ignore the signs and only consider the connectivity pattern. Since the largest strongly connected component has very few nodes, we take the largest weakly connected component, which comprises 664 nodes and 1078 edges.

This is a very sparse network and consequently the log-likelihood of the directed pRDRG (Figure 2.8a) keeps increasing as a function of the decay rate parameter γ in the range we tested. We select $g = 1/3$ as the optimal parameter for the Magnetic Laplacian, and compare the log-likelihood of two models in Figure 2.8b. This time the trophic version achieves a higher maximum likelihood, favouring a linear structure.

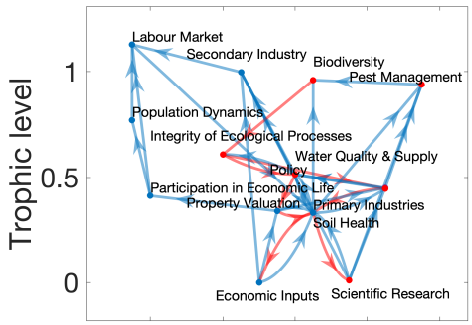
[3] http://snap.stanford.edu/data/S-cerevisiae.html



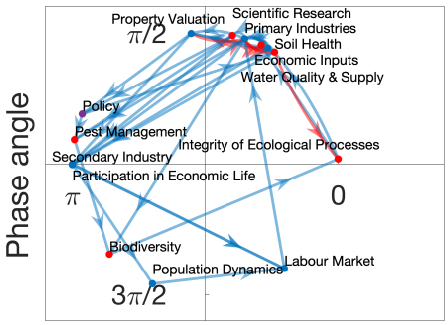
(a) Likelihood of directed pRDRG



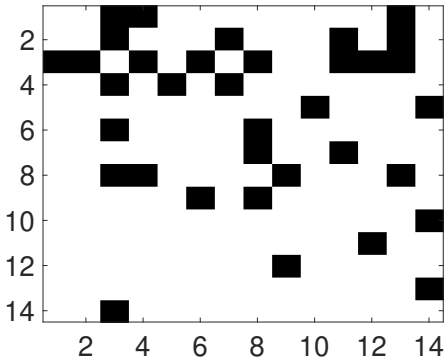
(b) Model comparison



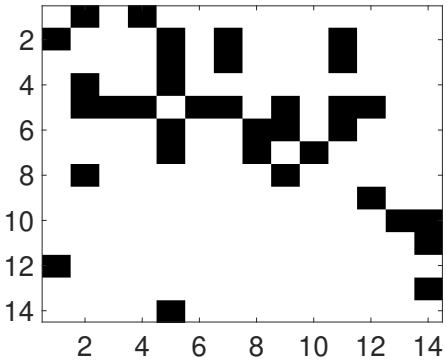
(c) Estimated trophic level



(d) Magnetic Eigenmap



(e) Trophic Laplacian reordering



(f) Magnetic Laplacian reordering

Figure 2.6: Results for the Motueka Catchment influence matrix

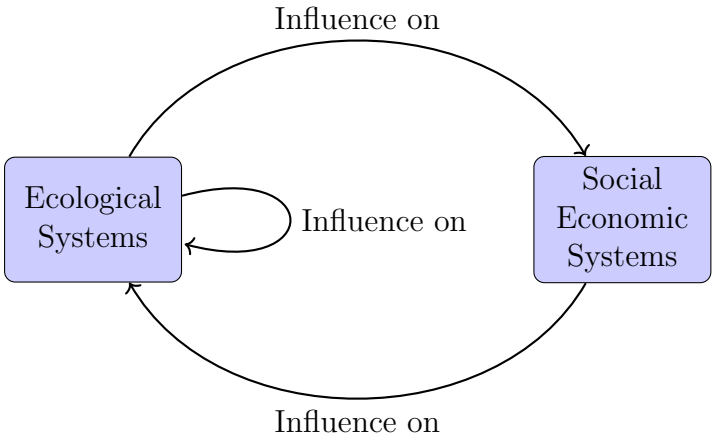
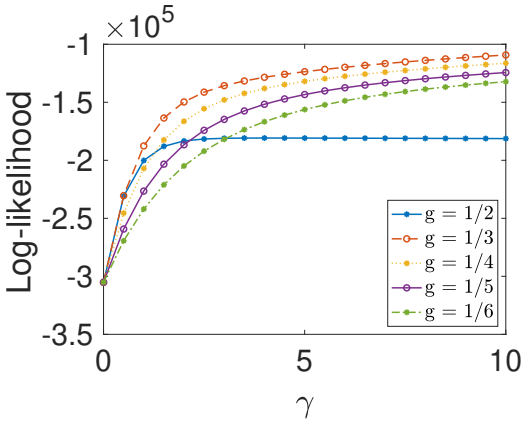
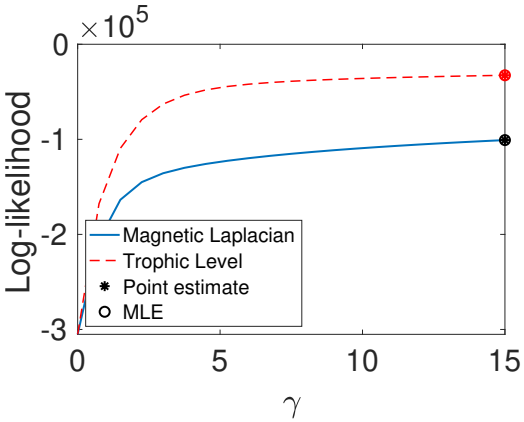


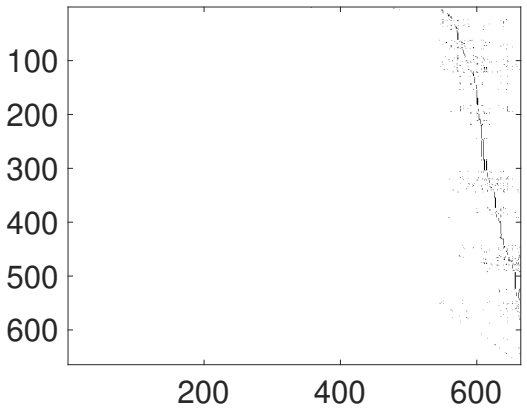
Figure 2.7: Influence matrix schematic graph, based on [22, Figure 5]



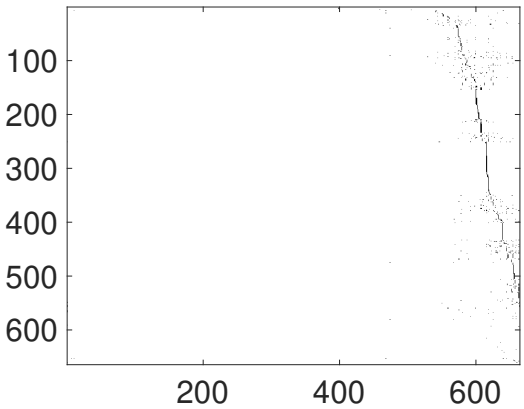
(a) Likelihood of directed pRDRG



(b) Model comparison

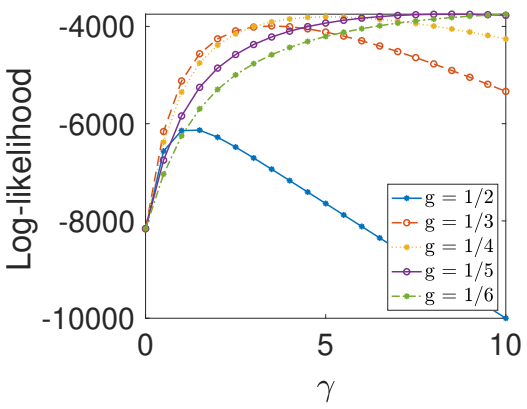


(c) Trophic Laplacian reordering

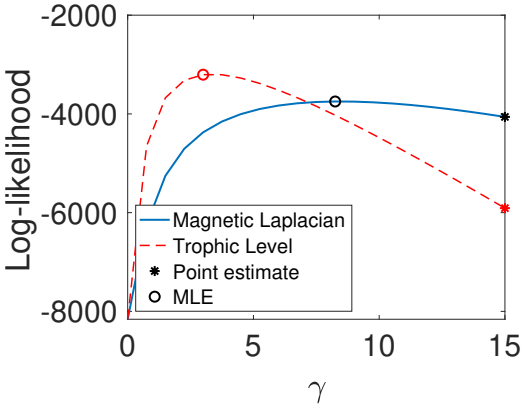


(d) Magnetic eigenmap

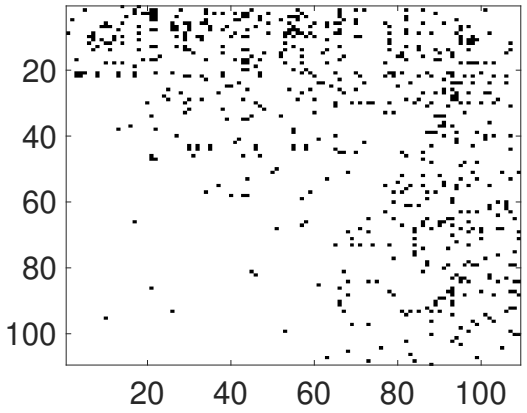
Figure 2.8: Yeast transcriptional regulation network



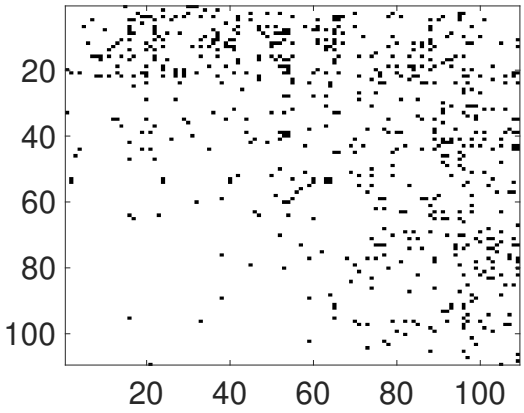
(a) Likelihood of directed pRDRG



(b) Model comparison



(c) Reordered matrix using the trophic level



(d) Reordered matrix using the Magnetic eigenmap

Figure 2.9: C. elegans frontal neural network

2.5.4 C. elegans Frontal Neural Network

C. elegans is the only organism whose neural network has been fully mapped. The neural network of C. elegans[4][65] is unweighted and directed, representing connections between neurons and synapses [61]. We investigate its largest strongly connected component with 109 nodes and 637 edges. The optimal value for the parameter g among the test points is $g = 1/5$ (Figure 2.9a). The Trophic Laplacian algorithm achieves a higher maximum likelihood than the Magnetic Laplacian algorithm using (Figure 2.9b). This preference for a linear directed structure is consistent with the tube-like shape of the organism [111].

2.5.5 Other Real Networks

A summary of further real-world network comparisons is given in Table 2.1. In the Data set column, we use (s) and (w) to indicate whether the largest *strongly* or *weakly* connected component is analyzed, respectively. The fourth column specifies the optimal parameter g for the Magnetic Laplacian determined through grid

[4] http://snap.stanford.edu/data/C-elegans-frontal.html

Data set	Nodes	Edges	g	$\ln(P_{pRDRG}/P_{Trophic})$
Directed pRDRG (s)	500	49277	1/5	5.99e+04
Food web (s) [106]	12	28	1/3	1.17e+01
Influence matrix (s) [22]	14	35	1/4	1.72e+01
US migration (s)[5]	51	729	1/6	5.03e+02
US IO (s)[6]	31	299	1/6	5.67e+01
Trade (s)[7]	17	85	1/6	2.02e+01
Transportation (s)[8] [65, 34]	456	71959	1/6	4.66e+04
Flight (s)[9]	227	23113	1/6	7.22e+03
Trophic level graph (w)	500	19956	1/6	-1.63e+04
C. elegans (s) [61]	109	637	1/6	-4.74e+02
Yeast (w) [77]	664	1078	1/3	-6.46e+04
Political blog (s)[10] [1]	793	15781	1/5	-3.42e+04
Shopping basket (w)[11]	27	84	1/6	-1.35e+02
Venue reopen (w)[6]	13	19	1/6	-1.82e+01
Word adjacency (w)[10][82]	112	425	1/6	-8.21e+02

Table 2.1: Comparison summary statistics. Periodic (linear) directed structure is found to be preferred for networks in the first 8 (last 7) rows.

search among the test points $g = 1/2, 1/3, 1/4, 1/5, 1/6$. The decay parameter γ used for the grid search ranges from 0 to 20 with a step size of 0.5. The last column shows the logarithm of the ratio between the maximum likelihoods of the directed pRDRG and trophic models. Hence, periodic/linear structure is seen to be favoured for the networks in the first 8 rows/last 7 rows.

2.6 Conclusion

Spectral methods can be used to extract structures from directed networks, allowing us to detect clusters, rank nodes, and visualize patterns. This chapter exploited a natural connection between spectral methods for directed networks and generative random graph models. We showed that the Magnetic Laplacian and Tropic Laplacian can each be associated with a range-dependent random graph. In the Magnetic Laplacian case, the new random graph model has the interesting property that the probabilities of $i \to j$ and $j \to i$ connections are not independent. Our theoretical analysis provided a workflow for quantifying the relative strength of periodic versus linear directed hierarchy, using a likelihood ratio, adding value to the standard approach of visualizing a new graph layout or

[5] https://www.census.gov/content/census/en/library/publications/2003/dec/censr-8.html

[6] https://stats.oecd.org/Index.aspx?DataSetCode=IOTSI4_2018

[7] http://www.economicswebinstitute.org/worldtrade.htm

[8] http://snap.stanford.edu/data/reachability.html

[9] https://www.visualizing.org/global-flights-network/

[10] http://www-personal.umich.edu/~mejn/netdata/

[11] https://www.dunnhumby.com/source-files/

reordering the adjacency matrix. We demonstrated the model comparison workflow on synthetic networks, and also showed examples where real networks were categorized as more linear or periodic. The results illustrate the potential for the approach to reveal interesting patterns in networks from ecology, biology, social sciences and other related fields.

Chapter 3

Hypergraph Laplacian

3.1 Introduction

A typical graph-based data set captures pairwise interactions between nodes. There is growing interest in understanding higher-order, group-level, interactions, with different paradigms being proposed [7, 63]. In this chapter, we represent such interactions with a hypergraph formulation; here each hyperedge involves two or more nodes. This framework is discussed in [4, 3, 103] and has found application in real-world problems such as epidemic spread modeling [51], image classification [115], and the study of biological networks [88].

A fundamental learning task on graph-based data is to embed nodes into a low-dimensional Euclidean space [96, 90]. The learned embedding can be used in follow-on tasks such as clustering, classification, and structure recovery. There are various types of learning algorithms for graphs; some design and analyze Laplacian matrices related to the graph [70], some solve maximum likelihood problems associated with random graph models [44, 85], and others involve more complex machine learning frameworks [46, 47].

In this chapter[1], we explore spectral methods that obtain node embeddings from the eigenvectors of a Laplacian matrix, as introduced in Section 1.3.3. Two main approaches have also recently been proposed for spectral clustering on hypergraphs. One approach is to employ higher-order Laplacian tensors [36]. Tensors in general contain richer information, however, their use can require considerably more computational expense than matrix algorithms, and the results can be difficult to visualize and interpret. A second approach is to "flatten" the higher order information into a representative node-level matrix. Some matrix-based approaches analyze the vertex-edge incidence matrix associated with a random walk interpretation [96], other frameworks utilise motif-based Laplacian matrices that can be generalized to various motifs and time steps [90, 68]. The methodology that we develop here fits into this second category by building a node-based matrix, using an intermediate step that looks over all hyperedge dimensions in order to incorporate higher order information.

A second aspect of this chapter is the connection between spectral methods and random models. Many graph embedding [54], re-ordering [44], clustering [16],

[1]Material in this chapter is adapted from [43].

and structure recovery [21, 105] techniques solve maximum-likelihood problems on graphs assuming specific generative models. Besides their application in these inverse problems, random graph models are useful inference tools for quantifying structure, predicting new or missing links, and improving the interpretability of learning algorithms by relating node embeddings to edge probabilities [45, 42]. Such connections have been investigated for undirected graphs [48] and directed graphs [42], and here we extend these ideas to the hypergraph setting. In particular, we associate customized spectral embedding algorithms with generative models that belong to a new class of range-dependent random hypergraphs that encourages short-range connections between nodes, generalizing existing graph models [44, 49]. These range-dependent random hypergraphs offer flexibility that is not available in stochastic block models [16] which require block sizes to be pre-specified or inferred.

We consider a spectral graph embedding algorithms, which is a special case of [90] when the motif considered are hyperedges. A periodic version, which extends [45] to hypergraph, is proposed where nodes are embedded into the unit circle. We allow weights for hyperedges to vary with their cardinality since the importance may vary in applications. For example, a conversation between two people may suggest stronger social ties between individuals than a meeting between a group of people; however, when it comes to model disease spread on social networks, the large hyperedges may have a higher impact. The rest of the chapter is structured as follows. Our notation is introduced in Section 3.2. In Sections 3.3 and 3.4 we define the linear and periodic hypergraph embedding algorithms and derive associated optimization problems. We propose random models associated with the hypergraph embedding algorithms in Section 3.5, which leads to a model comparison workflow that quantifies the relative strength of linear versus periodic structures. Numerical studies on synthetic and real-world hypergraphs using the proposed models are presented in Section 3.6.

The main contributions of this chapter are as follows.

- We propose new range-dependent generative models for hypergraphs that generate linear and periodic cluster patterns.
- We establish their connection with linear and periodic spectral embedding algorithms.
- We demonstrate on synthetic and real data that, after tuning model parameters to the data, these models can quantify the relative strength of linear and periodic structures.
- We perform prediction of triadic hyperedges (triangles) using the proposed linear model and show that it outperforms the existing average-score based method [2] on synthetic hypergraphs, and also on high school and primary school contact data when the amount of training data is limited.

3.2 Notation

We consider *undirected, unweighted* hypergraphs $G = (V, E)$ on the vertex set V containing n nodes and the hyperedge set E. We let $R \in \mathcal{R}$ be an unordered set of nodes, where $\mathcal{R}$ denotes the collection of all such sets. We use $|R|$ to denote the number of nodes in tuple R, that is, its cardinality, and we assume $2 \leq |R| \leq T$ for all $R \in E$.

Let A_R indicate the presence of a hyperedge, so that $A_R = 1$ if $R \in E$ and $A_R = 0$ otherwise. We define the t-th order n by n adjacency matrix $W^{[t]}$ such that $W^{[t]}_{ij}$ counts the number of hyperedges with cardinality t that contain distinctive nodes i and j; hence,

$$W^{[t]}_{ij} = \begin{cases} \sum_{R \in \mathcal{R}:|R|=t} A_R \, \mathbb{1}\left(i \in R\right) \mathbb{1}\left(j \in R\right) & \text{if } i \neq j, \\ 0 & \text{otherwise} , \end{cases}$$

where $\mathbb{1}$ is the indicator function. Similarly, we define the corresponding t-th order diagonal degree matrix $D^{[t]}$ such that

$$D^{[t]}_{ii} = \sum_{j \in V} W^{[t]}_{ij},$$

and the t-th order Laplacian matrix

$$L^{[t]} = D^{[t]} - W^{[t]}.$$

We will focus on one-dimensional embedding. We let $x_i \in \mathbb{R}$ be the location to which node i is assigned, and $\boldsymbol{x} = (x_1, x_2, ..., x_n)^T \in \mathbb{R}^n$.

3.3 Linear hypergraph embedding

Given a hypergraph, suppose we wish to find node embeddings $\boldsymbol{x} \in \mathbb{R}^n$ such that hyperedges tend to contain nodes that are a small distance apart. To formalize this idea, we can define a linear incoherence function $I_{\text{lin}}(\boldsymbol{x}, R)$ that sums up the squared Euclidean distance between all nodes pairs in tuple R:

$$I_{\text{lin}}(\boldsymbol{x}, R) = \sum_{i,j \in R} (x_i - x_j)^2. \tag{3.1}$$

We may then define the total linear incoherence of the hypergraph, $\eta_{\text{lin}}(G, \boldsymbol{x})$, by aggregating the linear incoherence over all node tuples. Furthermore, we may wish to tune the weights of hyperedges of different cardinalities through a coefficient $c_{|R|} \geq 0$ for node tuple R; that is,

$$\eta_{\text{lin}}(G, \boldsymbol{x}) = \sum_{R \in \mathcal{R}} c_{|R|} A_R I_{\text{lin}}(\boldsymbol{x}, R). \tag{3.2}$$

One justification for these tuning parameters $c_{|R|}$ is that they allow us to avoid the case where high-cardinality hyperedges dominate the expression. For example,

we can choose

$$c_t = \frac{1}{t(t-1)}$$

to balance the contributions from hyperedges with different sizes. A suitable choice of c_t may also depend on the relative importance of hyperedges in the application.

In Proposition 3.3.1 we show that the total linear incoherence maybe be written as a quadratic form involving the hypergraph Laplacian matrix

$$L = \sum_{t=2}^{T} c_t L^{[t]}. \tag{3.3}$$

Proposition 3.3.1. *For any $\boldsymbol{x} \in \mathbb{R}^n$ with L defined in (3.3), and $\eta_{lin}(G, \boldsymbol{x})$ defined in (3.2) we have*

$$\boldsymbol{x}^T L \boldsymbol{x} = \tfrac{1}{2}\eta_{lin}(G, \boldsymbol{x}). \tag{3.4}$$

Proof. It is straightforward to show that

$$\boldsymbol{x}^T L^{[t]} \boldsymbol{x} = \sum_{i,j \in V} x_i (D_{ij}^{[t]} - W_{ij}^{[t]}) x_j = \tfrac{1}{2} \sum_{i,j \in V} W_{ij}^{[t]} (x_i - x_j)^2.$$

Therefore,

$$\begin{aligned}
\boldsymbol{x}^T L \boldsymbol{x} &= \sum_{t=2}^{T} \boldsymbol{x}^T c_t L^{[t]} \boldsymbol{x} \\
&= \tfrac{1}{2} \sum_{t=2}^{T} c_t \sum_{i,j \in V} W_{ij}^{[t]} (x_i - x_j)^2 \\
&= \tfrac{1}{2} \sum_{t=2}^{T} c_t \sum_{i,j \in V} \sum_{R \in \mathcal{R}: |R| = t} A_R \, \mathbb{1}(i \in R) \, \mathbb{1}(j \in R) (x_i - x_j)^2 \\
&= \tfrac{1}{2} \sum_{R \in \mathcal{R}} c_{|R|} A_R \sum_{i,j \in R} (x_i - x_j)^2 = \tfrac{1}{2} \eta_{\text{lin}}(G, \boldsymbol{x}).
\end{aligned}$$

□

We note that each Laplacian $L^{[t]}$ is symmetric and positive semi-definite with smallest eigenvalue 0.

Assumption 3.3.1. *We assume that the unweighted, undirected graph associated with the binarized version of L is connected. It then follows that L has a single eigenvalue equal to 0 with all other eigenvalues positive. We further assume that there is a unique smallest positive eigenvalue, λ_2. (The eigenvector $\boldsymbol{v}^{[2]}$ corresponding to λ_2 is a generalization of the classic Fiedler vector.)*

In minimizing the total linear incoherence (3.2) we must avoid the trivial cases where (a) all nodes are located arbitrarily close to the origin and (b) all nodes are assigned to the same location. Hence it is natural to impose the constraints

$\|\boldsymbol{x}\| = 1$ and $\boldsymbol{x}^T\mathbf{1} = 1$. It then follows from the Rayleigh-Ritz Theorem [55, Theorem 4.2.2] that the quadratic form in Proposition 3.3.1 is solved by $\boldsymbol{x} = \boldsymbol{v}^{[2]}$. This leads us to Algorithm 4 below, which can also be considered as a special case of the algorithm in [36] where the motifs considered are hyperedges.

Algorithm 4: Linear Hypergraph Embedding Algorithm

Result: Node embedding $\boldsymbol{x} \in \mathbb{R}^n$
Input hyperedge adjacency matrices $W^{[2]}, W^{[3]}, ..., W^{[T]}$;
Construct diagonal degree matrices $D^{[t]}_{ii} = \sum_{j \in V} W^{[t]}_{ij}$;
Construct t**-th order Laplacians** $L^{[t]} = D^{[t]} - W^{[t]}$;
Construct hypergraph Laplacian $L = \sum_{t=2}^{T} c_t L^{[t]}$;
Compute second smallest eigenvalue λ_2 **and corresponding eigenvector** $\boldsymbol{v}^{[2]}$;
Embed nodes using $\boldsymbol{x} = \boldsymbol{v}^{[2]}$

Remark 3.3.1. *Algorithm 4 can be extended to higher dimensional embeddings where node i is assigned to $\boldsymbol{x}^{[i]} \in \mathbb{R}^d$ for $d > 1$. In this case we can generalize (3.1) to*

$$I_{lin}(\boldsymbol{x}, R) = \sum_{i,j \in R} \|\boldsymbol{x}^{[i]} - \boldsymbol{x}^{[j]}\|^2. \tag{3.5}$$

If we require the coordinate directions to be orthogonal, then the embedding is found via the eigenvectors corresponding to the d-smallest non-zero eigenvalues; see [52] for details in the graph case.

3.4 Periodic hypergraph embedding

In this section, we look at the periodic analogue of linear hypergraph embedding. Here nodes are embedded into the unit circle rather than along the real line. Such a periodic structure formed the basis of the classic "small world" model of Watts and Strogatz [109]. Results in [45] showed that certain real networks are better represented via this type of "wrap-around" notion of distance. Hence, it is of interest to develop concepts that apply to the hypergraph case.

We may position nodes on the unit circle by mapping them to phase angles $\boldsymbol{\theta} = \{\theta_i\}_{i=1}^n \in [0, 2\pi)$. We may then use a periodic incoherence function to quantify the distance between node pairs in the tuple R:

$$I_{\text{per}}(\boldsymbol{\theta}, R) = \sum_{i,j \in R} |e^{\mathbf{i}\theta_i} - e^{\mathbf{i}\theta_j}|^2. \tag{3.6}$$

Then the total periodic incoherence of the hypergraph becomes

$$\eta_{\text{per}}(G, \boldsymbol{\theta}) = \sum_{R \in \mathcal{R}} c_{|R|} A_R I_{\text{per}}(\boldsymbol{\theta}, R). \tag{3.7}$$

In Proposition 3.4.1 below, we relate the total periodic incoherence to a quadratic form involving the hypergraph Laplacian matrix (3.3).

Proposition 3.4.1. *Let $\boldsymbol{\psi} \in \mathbb{C}^n$ be such that $\psi_j = e^{\mathbf{i}\theta_j}$. Then*

$$\boldsymbol{\psi}^H L \boldsymbol{\psi} = \tfrac{1}{2}\eta_{per}(G, \boldsymbol{\theta}). \tag{3.8}$$

Proof. We have

$$\begin{aligned}
\boldsymbol{\psi}^H L^{[t]} \boldsymbol{\psi} &= \boldsymbol{\psi}^H D^{[t]} \boldsymbol{\psi} - \boldsymbol{\psi}^H W^{[t]} \boldsymbol{\psi} \\
&= \sum_{i \in V} e^{-\mathbf{i}\theta_i} \Big(\sum_{j \in V} W_{ij}^{[t]}\Big) e^{\mathbf{i}\theta_i} - \sum_{i,j \in V} e^{-\mathbf{i}\theta_i} W_{ij}^{[t]} e^{\mathbf{i}\theta_j} \\
&= \sum_{i,j \in V} W_{ij}^{[t]} (1 - e^{-\mathbf{i}\theta_i} e^{\mathbf{i}\theta_j}) \\
&= \tfrac{1}{2} \sum_{i,j \in V} W_{ij}^{[t]} (2 - e^{-\mathbf{i}\theta_i} e^{\mathbf{i}\theta_j} - e^{-\mathbf{i}\theta_j} e^{\mathbf{i}\theta_i}) \\
&= \tfrac{1}{2} \sum_{i,j \in V} W_{ij}^{[t]} (e^{\mathbf{i}\theta_i} - e^{\mathbf{i}\theta_j})(e^{-\mathbf{i}\theta_i} - e^{-\mathbf{i}\theta_j}) \\
&= \tfrac{1}{2} \sum_{i,j \in V} W_{ij}^{[t]} |e^{\mathbf{i}\theta_i} - e^{\mathbf{i}\theta_j}|^2.
\end{aligned}$$

Then the proof may be completed in a similar way to the proof of Proposition 3.3.1. □

Appealing again to the Rayleigh–Ritz theorem [55, Theorem 4.2.2], the quadratic form in (3.8) is minimized over all $\boldsymbol{\psi} \in \mathbb{C}^n$ with $\|\boldsymbol{\psi}\| = 1$ and $\boldsymbol{\psi}^H \mathbf{1} = 1$ by taking $\boldsymbol{\psi} = \boldsymbol{v}^{[2]}$. However, this real-valued eigenvector cannot be proportional to a vector with components of the form $e^{\mathbf{i}\theta_j}$. Hence, following the approach in [45] we will use the heuristic of setting

$$\theta_i = \operatorname{angle}(v_i^{[2]} + \mathbf{i} v_i^{[3]}) \in [-\pi, \pi], \tag{3.9}$$

defined as

$$v_i^{[2]} + \mathbf{i} v_i^{[3]} = |v_i^{[2]} + \mathbf{i} v_i^{[3]}| \cdot e^{i\theta_i},$$

where $\boldsymbol{v}^{[3]}$ is an eigenvector corresponding to the next-smallest eigenvalue of L. Such a heuristic assumption converts two real eigenvectors into a complex vector, which gives an approximate solution to the minimization problem. The resulting workflow is summarized in Algorithm 5.

We also note that for simple unweighted, undirected graphs, finding $\boldsymbol{\theta}$ that minimizes $\eta_{\text{per}}(G, \boldsymbol{\theta})$ is equivalent to the formulation proposed in [45]. This may be shown by letting $u_i = \cos\theta_i$ and $z_i = \sin\theta_i$ and expanding (3.7) as

$$\sum_{i \in V} \sum_{j \in V} W_{ij}^{[2]} \left((u_i - u_j)^2 + (z_i - z_j)^2\right),$$

which simplifies to

$$2\boldsymbol{u}^T (D^{[2]} - W^{[2]}) \boldsymbol{u} + 2\boldsymbol{z}^T (D^{[2]} - W^{[2]}) \boldsymbol{z}. \tag{3.10}$$

This is essentially equation (3.1) in [45], derived from a slightly different perspec-

tive.

We then arrive at Algorithm 5 below.

Algorithm 5: Periodic Hypergraph Embedding Algorithm

Result: Node embedding $\boldsymbol{\theta} = \{\theta_i\}_{i=1}^n \in [0, 2\pi)$
Input adjacency matrices $W^{[2]}, W^{[3]}, ..., W^{[T]}$;
Construct diagonal degree matrices $D_{ii}^{[t]} = \sum_{j \in V} W_{ij}^{[t]}$;
Construct t**-th order Laplacians** $L^{[t]} = D^{[t]} - W^{[t]}$;
Construct hypergraph Laplacian $L = \sum_{t=2}^{T} c_t L^{[t]}$;
Compute second and third smallest eigenvalues λ_2 **and** λ_3 **and corresponding eigenvectors** $\boldsymbol{v}^{[2]}$ **and** $\boldsymbol{v}^{[3]}$;
Calculate phase angles $\theta_i = \text{angle}(v_i^{[2]} + \mathbf{i} v_i^{[3]})$;
Embed nodes using $\boldsymbol{\theta}$

3.5 Generative hypergraph models

We now discuss a connection between the minimization of total incoherence and generative models. Let us consider finding a node embedding $\boldsymbol{x} \in \mathbb{R}^n$ that minimizes a generic total graph incoherence expression

$$\eta(G) = \sum_{R \in \mathcal{R}} c_{|R|} A_R I(\boldsymbol{x}, R), \tag{3.11}$$

for a non-negative incoherence function $I(\boldsymbol{x}, R)$. We consider the case where the $x_i \in \mathbb{R}$ must take distinct values from a discrete set $\{\nu_i\}_{i=1}^n$, where $\nu_i \in \mathbb{R}$; that is, we must have $x_i = \nu_{p_i}$, where $p \in \mathcal{P}$ is a permutation vector. In the linear case, this set may be the integers from 1 to n and in the periodic case this set may be equally spaced angles in $[0, 2\pi)$.

Now consider a random hypergraph model where each hyperedge involving node tuple $R \in \mathcal{R}$ is generated independently with probability

$$\mathbf{P}(A_R = 1) = f_R(\boldsymbol{x}, R), \tag{3.12}$$

for a function f_R that takes values between 0 and 1. We have the following connection.

Theorem 3.5.1. *Suppose $\boldsymbol{x} \in \mathbb{R}^n$ is constrained to take values from a discrete set such that $x_i = \nu_{p_i}$, where $p \in \mathcal{P}$ is a permutation vector. Then minimizing the total incoherence (3.11) over all such $\boldsymbol{x}$ is equivalent to maximizing over all such $\boldsymbol{x}$ the likelihood that the hypergraph is generated by a model of the form (3.12), where*

$$f_R(\boldsymbol{x}, R) = \frac{1}{1 + e^{\gamma c_{|R|} I(\boldsymbol{x}, R)}} \tag{3.13}$$

for any positive γ.

Proof. Using (3.12), the likelihood of the whole hypergraph is

$$\begin{aligned} L(G) &= \prod_{R\in\mathcal{R}:A_R=1} f_R(\boldsymbol{x},R) \prod_{R\in\mathcal{R}:A_R=0} (1-f_R(\boldsymbol{x},R)) \\ &= \prod_{R\in\mathcal{R}:A_R=1} \frac{f_R(\boldsymbol{x},R)}{1-f_R(\boldsymbol{x},R)} \prod_{R\in\mathcal{R}} (1-f_R(\boldsymbol{x},R)), \end{aligned}$$

which leads to the log-likelihood

$$\ln(L(G)) = \sum_{R\in\mathcal{R}:A_R=1} \ln\left(\frac{f_R(\boldsymbol{x},R)}{1-f_R(\boldsymbol{x},R)}\right) + \sum_{R\in\mathcal{R}} \ln\left((1-f_R(\boldsymbol{x},R))\right). \tag{3.14}$$

The second term on the right-hand side, which is the probability of the null hypergraph, is independent of the permutation. Hence, with (3.13), maximizing the log-likelihood of the hypergraph is equivalent to minimizing

$$\sum_{R\in\mathcal{R}:A_R=1} \ln\left(\frac{1-f_R(\boldsymbol{x},R)}{f_R(\boldsymbol{x},R)}\right) = \sum_{R\in\mathcal{R}} c_{|R|}A_R\gamma I(\boldsymbol{x},R) = \gamma\,\eta(G). \tag{3.15}$$

□

Remark 3.5.1. *Theorem 3.5.1 can be extended to the case where node i is assigned to $\boldsymbol{x}^{[i]} \in \mathbb{R}^d$ for $d > 1$, and a higher-dimensional incoherence function in (3.5) is considered. In this scenario, we constrain $\boldsymbol{x}^{[i]} \in \mathbb{R}^d$ to take values from a discrete set $\{\boldsymbol{\nu}^{[i]}\}_{i=1}^n$ where $\boldsymbol{\nu}^{[i]} \in \mathbb{R}^d$, such that $\boldsymbol{x}^{[i]} = \boldsymbol{\nu}^{[p_i]}$ for a permutation vector $p \in \mathcal{P}$. Then we can follow the same arguments as in Theorem 3.5.1 to derive a model described by (3.12) and (3.13), where $\boldsymbol{x} \in \mathbb{R}^{n\times d}$ and $I(\boldsymbol{x},R) = \sum_{i,j\in R}\|\boldsymbol{x}^{[i]} - \boldsymbol{x}^{[j]}\|_2^2$.*

For a hypergraph generated by model (3.13) the number of hyperedges connecting the node tuple R follows a Bernoulli distribution with probability $1/1 + e^{\gamma c_{|R|} I(\boldsymbol{x},R)}$. The log-odds of the hyperedge decay linearly with the incoherence of the node tuple since

$$\ln(f_R(\boldsymbol{x},R)/(1-f_R(\boldsymbol{x},R))) = -\gamma c_{|R|} I(\boldsymbol{x},R),$$

where the factor $\gamma c_{|R|}$ determines the decay rate. The probability of a hyperedge is highest when all nodes overlap, i.e., $I(\boldsymbol{x},R) = 0$, which gives a $1/2$ likelihood. If we generate hyperedges in repeated trials for the node tuple R, the variance of the number of hyperedges is $e^{c_{|R|}\gamma I(\boldsymbol{x},R)}/(1+e^{c_{|R|}\gamma I(\boldsymbol{x},R)})^2$. When $I(\boldsymbol{x},R) = 0$, the largest variance of $1/4$ is achieved. The expected total number of hyperedges of the whole hypergraph G can be expressed as

$$\sum_{R\in\mathcal{R}} f_R(\boldsymbol{x},R) = \sum_{R\in\mathcal{R}} \frac{1}{1+e^{\gamma c_{|R|} I(\boldsymbol{x},R)}}.$$

We note that Theorem 3.5.1 introduces the extra scaling parameter γ. This parameter plays no direct role in Algorithms 4 and 5. However, a value for γ is needed if we wish to compare the likelihoods of the two models having inferred the

embeddings. In principle, we may fit the parameter γ to a given hypergraph by matching the observed number of hyperedges with their expectation. However, from a computational point of view, this is rather challenging in general, since the computational complexity of the expectation calculation is $\mathcal{O}(n^T)$ when the maximum cardinality of a considered hyperedge is T. Hence, given an embedding, in practice we prefer to pick γ by maximizing the likelihood, as described in the following subsection.

3.5.1 The Discrete Constraint

In the model above we have

$$\ln(L(G)) = \ln(L(G_0)) - \gamma\eta(G, \boldsymbol{x}),$$

where $\ln(L(G))$ represents the likelihood of the null hypergraph with no hyperedges. When maximizing the likelihood $L(G)$, we must constrain $\mathbf{x}$ to take discrete values to make it equivalent to minimizing the total incoherence $\gamma\eta(G, \mathbf{x})$, as the likelihood of the null hypergraph is dependent on $\mathbf{x}$.

One way to eliminate the discrete restriction can be, rather than focusing on maximizing the log-likelihood, to consider maximizing the log-odds:

$$\ln\left(\frac{L(G)}{L(G_0)}\right) = -\gamma\eta(G, \boldsymbol{x}).$$

An alternative method to eliminate the discrete restriction is to consider maximizing the posterior probability

$$P(\boldsymbol{x}|G) = \frac{P(G|\boldsymbol{x})P(\boldsymbol{x})}{P(G)}.$$

The dependence on $\boldsymbol{x}$ within the posterior probability can be removed by forming assumptions on the prior probabilities $P(\boldsymbol{x})$ and $P(G)$.

Given that $L(G_0)$ is only dependent on the node location $\boldsymbol{x}$ and is independent of the graph G, we can conveniently express $L(G_0)$ as $1/\Omega(\boldsymbol{x})$ for a function Ω.

Assume that the prior distribution are

$$P(x) = \Omega(\boldsymbol{x}) / \sum_{\boldsymbol{x}} \Omega(\boldsymbol{x}),$$

$$P(G) = \sum_{\boldsymbol{x}} e^{-\gamma\eta(G,\boldsymbol{x})} / \sum_{\boldsymbol{x}} \Omega(\boldsymbol{x}).$$

For the conditional probability $P(G|\boldsymbol{x})$ we can simply use

$$P(G|\boldsymbol{x}) = L(G) = L(G_0)e^{-\gamma\eta(G,\boldsymbol{x})}.$$

This gives us:

$$P(G|\boldsymbol{x}) = e^{-\gamma\eta(G,\boldsymbol{x})}/\Omega(\boldsymbol{x}),$$

From this, we can express the posterior probability as:

$$P(\boldsymbol{x}|G) = \frac{e^{-\gamma\eta(G,\boldsymbol{x})}}{\sum_{\boldsymbol{x}} e^{-\gamma\eta(G,\boldsymbol{x})}}.$$

As the denominator does not rely on $\boldsymbol{x}$, this implies that the most probable $\boldsymbol{x}$ for a given graph G is the one that yields the minimum incoherence $\eta(G, \boldsymbol{x})$

3.5.2 Model comparison

Under the assumption that a given hypergraph arose from a mechanism that favours connections between "nearby" nodes (in some latent, unobservable configuration), it is of interest to know whether a linear or periodic distance provides a better description.We may address this question using a model comparison approach. As in Sections 3.3 and 3.4, we consider one-dimensional embeddings, such that both the linear and periodic version have $n + 1$ parameters given the Laplacian coefficients: n node embeddings and a decay parameter γ. The node embeddings will be estimated from Algorithms 4 and 5. For any choice of γ, we may then calculate the corresponding likelihood for each type of hypergraph, given the embedding. We may then compare the models by reporting plots of likelihood versus γ or by reporting the maximum likelihood over all γ. We note that Theorem 3.5.1 states that node embeddings that minimize the incoherence also maximize the graph likelihood under the given discrete constraints. We note that Algorithms 4 and 5 minimize linear and periodic incoherence after relaxing the discrete constraints in Theorem 3.5.1 for computational feasibility. Such heuristics are often used in discrete programming. Therefore instead of the exact maximum likelihood, we get an estimated maximum likelihood. An overall workflow is shown below in Algorithm 6.

Algorithm 6: Model Comparison

Result: Comparison of possible graph structures
Input hypergraph G;
Compute linear embedding using Algorithm 4;
Compute periodic embedding using Algorithm 5;
Calculate maximum likelihood of linear model over $\gamma > 0$ using (3.14), (3.13), and (3.1) ;
Calculate maximum likelihood of periodic model over $\gamma > 0$ using (3.14), (3.13), and (3.6) ;
Compare likelihoods or report maxima

3.5.3 Weighted Generative Hypergraph Model

While our previous section and numerical experiments focused on unweighted hypergraphs, we will now present a derivation for a weighted hypergraph generative model. In this model, we assume that a hyperedge is independently generated for each unordered node tuple R, and a weight denoted as W_R is randomly assigned to it, with $W_R \in (0, 1)$. The following theorem establishes the relationship between minimizing the incoherence in (3.11) and the weighted random hypergraph

model.

Theorem 3.5.2. *Suppose $\boldsymbol{x} \in \mathbb{R}^n$ is constrained to take values from a discrete set such that $x_i = \nu_{p_i}$, where $p \in \mathcal{P}$ is a permutation vector. Then minimizing the total incoherence (3.11) over all such $\boldsymbol{x}$ is equivalent to maximizing over all such $\boldsymbol{x}$ the likelihood that the hypergraph is generated by a model above, where W_R follows a density distribution*

$$f_R(u) = \frac{1}{Z_R(\boldsymbol{x})e^{\gamma u I(\boldsymbol{x})}}, \text{ for } u \in [0,1), \quad \text{and } f(u) = 0 \text{ otherwise.} \tag{3.16}$$

Here, $\gamma > 0$ is a constant, and the normalization factor is given by:

$$Z_R(\boldsymbol{x}) = \int_{y=0}^{1} \frac{1}{e^{\gamma y I(\boldsymbol{x})}}\, dy = \frac{1 - e^{-\gamma I(\boldsymbol{x})}}{\gamma I(\boldsymbol{x})} \text{ for } I(\boldsymbol{x}) > 0, \quad \text{and } Z_R = 1 \text{ for } I(\boldsymbol{x}) = 0.$$

Proof. The likelihood of the graph G is

$$L(G) = \prod_{R \in \mathcal{R}} f_R(W_R) = \prod_{R \in \mathcal{R}} \frac{1}{Z_R(\boldsymbol{x})e^{\gamma W_R I(\boldsymbol{x})}} \tag{3.17}$$

$$\tag{3.18}$$

Therefore the log-likelihood is

$$\ln\left(L(G)\right) = -\sum_{R \in \mathcal{R}} \ln\left(Z_R(\boldsymbol{x})e^{\gamma W_R I(\boldsymbol{x})}\right) \tag{3.19}$$

$$= -\ln\left(\prod_{R \in \mathcal{R}} Z_R(\boldsymbol{x})\right) - \sum_{R \in \mathcal{R}} \gamma W_R I(\boldsymbol{x}) \tag{3.20}$$

$$= -\ln\left(\prod_{R \in \mathcal{R}} Z_R(\boldsymbol{x})\right) - \eta(G, \boldsymbol{x}). \tag{3.21}$$

Note that the first term is a product over all tuples and it depends on $\boldsymbol{x}$. However, since we constrain $\mathbf{x}$ to be a permutation of a discrete set, the first term becomes a constant. Consequently, maximizing the log-likelihood with respect to $\mathbf{x}$ is equivalent to minimizing $\eta(G, \mathbf{x})$. □

Properties of the Weighted Model

Taking the logarithm of the likelihood ratio, assuming $I(\mathbf{x})$ is fixed, yields:

$$\ln(f_R(u_1)/f_R(u_0)) = \ln(e^{\gamma I(\boldsymbol{x})(u_0 - u_1)}) = -\gamma I(\boldsymbol{x})(u_1 - u_0).$$

This indicates that the larger the edge weight, the less probable the occurrence of the hyperedge. Furthermore, the log-likelihood exhibits a linear relationship with u, with a slope of $-\gamma I(\mathbf{x})$. We can imagine that the distribution can be plotted as a exponential decay curve, and the area under the curve is always 1. Therefore when u is held constant, a larger value of $I(\mathbf{x})$ (resulting in faster decay) yields a

steeper curve, with a higher probability density near zero weight. Conversely, for smaller values of $I(\mathbf{x})$, the curve becomes flatter, indicating a higher likelihood of weights closer to one. Thus, the model can generate graphs where nearby nodes (smaller $I(\mathbf{x})$) are more likely to form hyperedges with larger weights. To demonstrate this property rigorously, we can examine the cumulative distribution function $F(u)$ below while assuming $I(\boldsymbol{x}) > 0$ as follows:

$$F(u) = P(W_R \leq u) = \int_{y=0}^{u} f_R(u)\mathrm{d}u = \frac{1 - e^{-\gamma u I(\boldsymbol{x})}}{Z_R \gamma I(\boldsymbol{x})} = \frac{1 - e^{-\gamma u I(\boldsymbol{x})}}{1 - e^{-\gamma I(\boldsymbol{x})}}.$$

It can be easily shown that $dF/dI > 0$, that $dF/dI > 0$, implying the cumulative probability decreases as the incoherence $I(\boldsymbol{x})$ increase.

The expected weight of the hyperedge between tuple R can be calculated as follows:

$$\mathbb{E}(W_R) = \int_0^1 u f_R(u)\mathrm{d}u = \int_0^1 \frac{u}{Z_R(\boldsymbol{x}) e^{\gamma u I(\boldsymbol{x})}}\mathrm{d}u = \frac{1 - (1 + \gamma I(\boldsymbol{x}))e^{-\gamma I(\boldsymbol{x})}}{\gamma I(\boldsymbol{x})(1 - e^{-\gamma I(\boldsymbol{x})})}$$

for $I(\boldsymbol{x}) > 0$, and

$$\int_0^1 u\mathrm{d}u = \frac{1}{2}$$

for $I(\boldsymbol{x}) = 0$. The expected weight decreases as $I(\mathbf{x})$ increases when $I(\mathbf{x}) > 0$, and its maximum value is $1/2$ when $I(\mathbf{x}) = 0$. The expected total weight of the whole graph using a simple sum over all tuples of nodes is

$$\mathbb{E}(\sum_{R \in \mathcal{R}} W_R) = \sum_{R \in \mathcal{R}} (\frac{1 - (1 + \gamma I(\boldsymbol{x}))e^{-\gamma I(\boldsymbol{x})}}{\gamma I(\boldsymbol{x})(1 - e^{-\gamma I(\boldsymbol{x})})}) \mathbb{I}(I(\boldsymbol{x}) \neq 0) + \frac{1}{2}\mathbb{I}(I(\boldsymbol{x}) = 0))$$

Once the node location $\mathbf{x}$ is known, we can calculate the expected edge weight. The computational complexity of this calculation depends on the number of tuples R. For instance, if we only consider simple edges, the complexity will be $\mathcal{O}(n^2)$. However, if we consider hyperedges with a maximum cardinality of T, the complexity will be $\mathcal{O}(n^T)$. The complexity will increase rapidly as T becomes larger.

Another notable property is that the weights are always non-zero, as per our definition, resulting in a dense hypergraph. However, due to the exponential decay of the probability density, the majority of edges will have small weights. To get a sparse hypergraph from the model, one option is to apply a weight threshold to truncate the edges.

3.6 Experiments

3.6.1 Model Comparison

Synthetic Hypergraphs

In this section we test the performance of Algorithms 4, 5 and 6 in a controlled setting. To do this, we generate hypergraphs with either linear or periodic clustered structure using the proposed random model. For simplicity, we only consider dyadic and triadic edges, although the experiments can be extended to include higher-order hyperedges.

Linear hypergraph with clustered nodes We first generate hypergraphs with K planted clusters $C_1, C_2, ..., C_K$ of size m, and $n = mK$ nodes. We embed the nodes using $x_i = \frac{2(l-1)}{K} + \sigma$ if $i \in C_l$, where $\sigma \sim \text{unif}(-a, a)$ is an additive uniform noise. Hyperedges are then drawn randomly according to model (3.13) with the linear incoherence (3.1).

We note that, in practice, the embedding algorithms must choose values c_2 and c_3 in order to form the hypergraph Laplacian, and the model comparison algorithm must choose a value for γ. We are therefore interested in the sensitivity of the process with respect to c_2 and c_3, and in the accuracy with which γ can be estimated. We use c_2, c_3 and γ_0 to denote parameters used by the generative model to create the synthetic data; we also let c_2^* and c_3^* denote the corresponding parameters used in the spectral embedding algorithms and let γ^* represent an inferred value of γ_0. We choose $c_2 = 1$ and $c_3 = 1/3$ so that the weight of a hyperedge is inversely proportional to the number of node pairs involved. We let $m = 50$, $K = 5$, $a = 0.05$, and vary the decay parameter γ_0 from 0 to 10. Figure 3.1a shows an example of the dyadic adjacency matrix, $W^{[2]}$, with $\gamma_0 = 4$, where dots represent non-zeros. A corresponding triadic adjacency matrix, $W^{[3]}$, is shown in Figure 3.1b. In all our tests we discard hypergraphs that do not satisfy Assumption 3.3.1.

For each synthetic hypergraph, we estimate the maximum log-likelihood assuming a linear or a periodic structure using Algorithm 6. For each input decay parameter γ_0, 40 hypergraphs are generated independently and the average maximum log-likelihood is plotted in Figure 3.1c. The shaded regions represent the estimated 80% confidence interval. In this case, the linear model correctly achieves a higher average maximum log-likelihood. The tight bound of the confidence interval suggests that the result is consistent across random trials.

We then perform K-means clustering using the periodic and linear embeddings assuming 5 clusters and plot the Adjusted Rand Index (ARI) [56, 76, 89] in Figure 3.1d. Here, a larger ARI indicates a better clustering result. The dotted line shows the average over 40 independently trials for each γ_0 value and the shaded area is the estimated 80% confidence interval. The plot suggests that the clustering from the linear embedding outperforms the clustering from the periodic embedding.

We are interested in the effect of parameters c_3 and c_3^* that control the weight of triadic edges in the random graph model and spectral embedding algorithm

respectively. To conduct an experiment, we fix the weight of dyadic edges $c_2 = 1$, $c_2^* = 1$, and decay parameter $\gamma_0 = \gamma^* = 1$, while varying c_3 and c_3^*. The maximum log-likelihood of the linear model (Figure 3.1e) and the ARIs using the linear embedding (Figure 3.1f) are shown as heat-maps over c_3 and c_3^*. Values are the average over 40 random trials. Overall, choosing $c_3^* = c_3$, gives the highest maximum likelihood. Therefore, when the true c_3 is not known, it can be estimated using a maximum likelihood method. In terms of the clustering result we note that when c_3 is large, for example, when $c_3 > 0.3$, using information from triadic edges by setting $c_3^* > 0$ achieves a better ARI than using only diadic edges, i.e., $c_3^* = 0$. This is because a large c_3 encourages more triadic edges to be formed within clusters, whereas a small c_3 leads to more triadic edges between clusters. In general the larger the c_3, the less sensitive the ARI is to the choice of c_3^*

Periodic hypergraph with clustered nodes To generate hypergraphs with periodic clusters, we use a node embedding based on a vector of angles $\boldsymbol{\theta} = (\theta_1, \theta_2, ..., \theta_n)^T$ in $[0, 2\pi)$, forming K clusters $C_1, C_2, \ldots, C_K$ of size m. In particular, we let

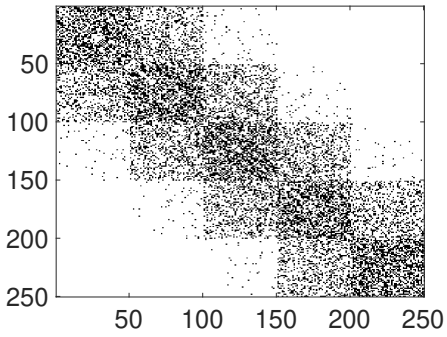
$$\boldsymbol{\theta}_i = \frac{2\pi(l-1)}{K} + \sigma$$

if $i \in C_l$ for $1 \leq l \leq K$, where $\sigma \sim \text{unif}(-a, a)$ is the added noise. The hyperedges are generated using model (3.13), where the incoherence function is defined in (3.6). We choose $a = 0.05\pi$, $c_2 = 1$, $c_3 = 1/3$ and vary the decay parameter γ_0. Examples of the dyadic and triadic adjacency matrices with $\gamma_0 = 1$ are shown in Figure 3.2a and 3.2b.

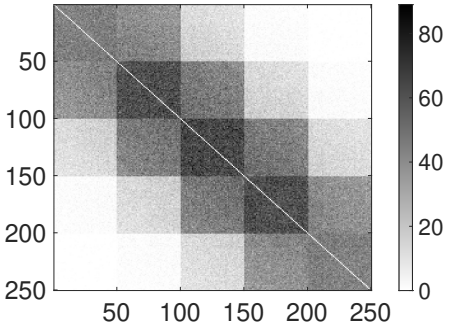
Using the same approach as in the previous section, we compare the maximum log-likelihood and ARIs assuming linear and periodic structures in Figure 3.2c and 3.2d. We see that the periodic model achieves a higher maximum, and on average the periodic embedding produces higher ARIs.

Heat-maps in Figure 3.2e and 3.2f show results for different combinations of c_3 and c_3^* for the periodic embedding algorithm. These results were generated in the same way as for Figures 3.1c and 3.1d. Higher maximum likelihoods are achieved near the diagonal where $c_3^* = c_3$, hence the true parameters c_3 for the underlying hypergraph can be estimated using the maximum likelihood method. As in the previous example, when $c_3 \geq 0.3$, using the triadic edges ($c_3^* > 0$) improves the ARI. When $c_3 < 0.3$, increasing c_3^* leads to an inferior clustering result. However when $c_3 \geq 0.3$, ARI becomes less sensitive to the choice of c_3^* as long as it is positive.

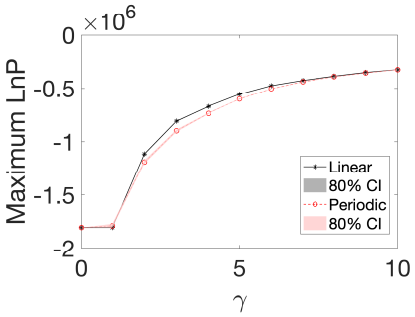
In summary, these tests indicate that the algorithms are able to correctly distinguish between linear and periodic range-dependency when one such structure is present in the data. We observed that setting $c_3^* > 0$ improves the ARI when the triadic edges have a strong structural pattern; that is, when c_3 is large. Moreover, when the true parameter c_3 is unknown we recommend choosing c_3^* based on a maximum likelihood estimation, that is, finding the value c_3^* that returns the largest maxima in Algorithm 6. Such a choice also achieves reasonable ARIs in our synthetic examples as shown in the diagonal entries in Figure 3.1f and 3.2f.



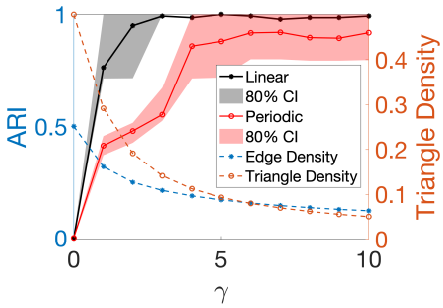
(a)



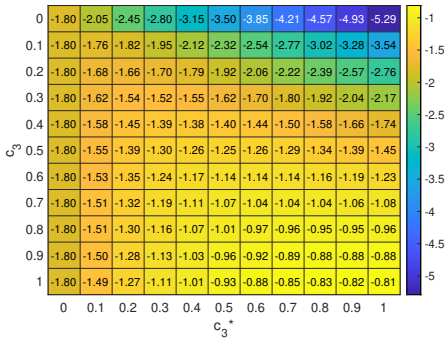
(b)



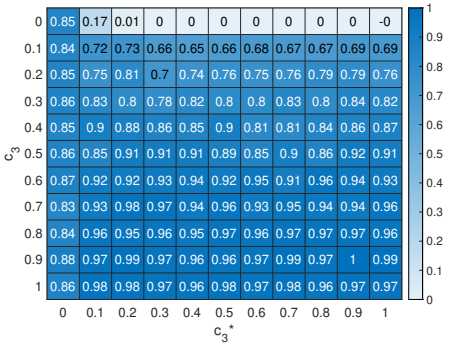
(c)



(d)



(e) Linear log-likelihood of graph $\times 10^{-6}$



(f) Linear ARI

Figure 3.1: Model comparison experiments on synthetic linear hypergraphs. (a) Black pixels in the dyadic adjacency matrix ($\gamma_0 = 4$) represent edges, and they reveal 5 clusters in the diagonal blocks. (b) In the triadic adjacency matrix, colors reflect the number of triangles shared between nodes ($\gamma_0 = 4$). (c) The linear embedding achieves higher log-likelihoods than the periodic version for hypergraphs generated with different γ_0. (d) Adjusted Rand Indices of K-means clustering based on the linear and periodic embedding are plotted against γ_0. (e) Values in the heatmap represent the maximum likelihoods of the linear model $\times 10^{-6}$ from Algorithm 6. The maxima are found along the diagonal when $c_3^* = c_3$. (f) Values represent the Adjusted Rand Indices of K-means clustering based on the linear embedding for different values of c_3^* and c_3.

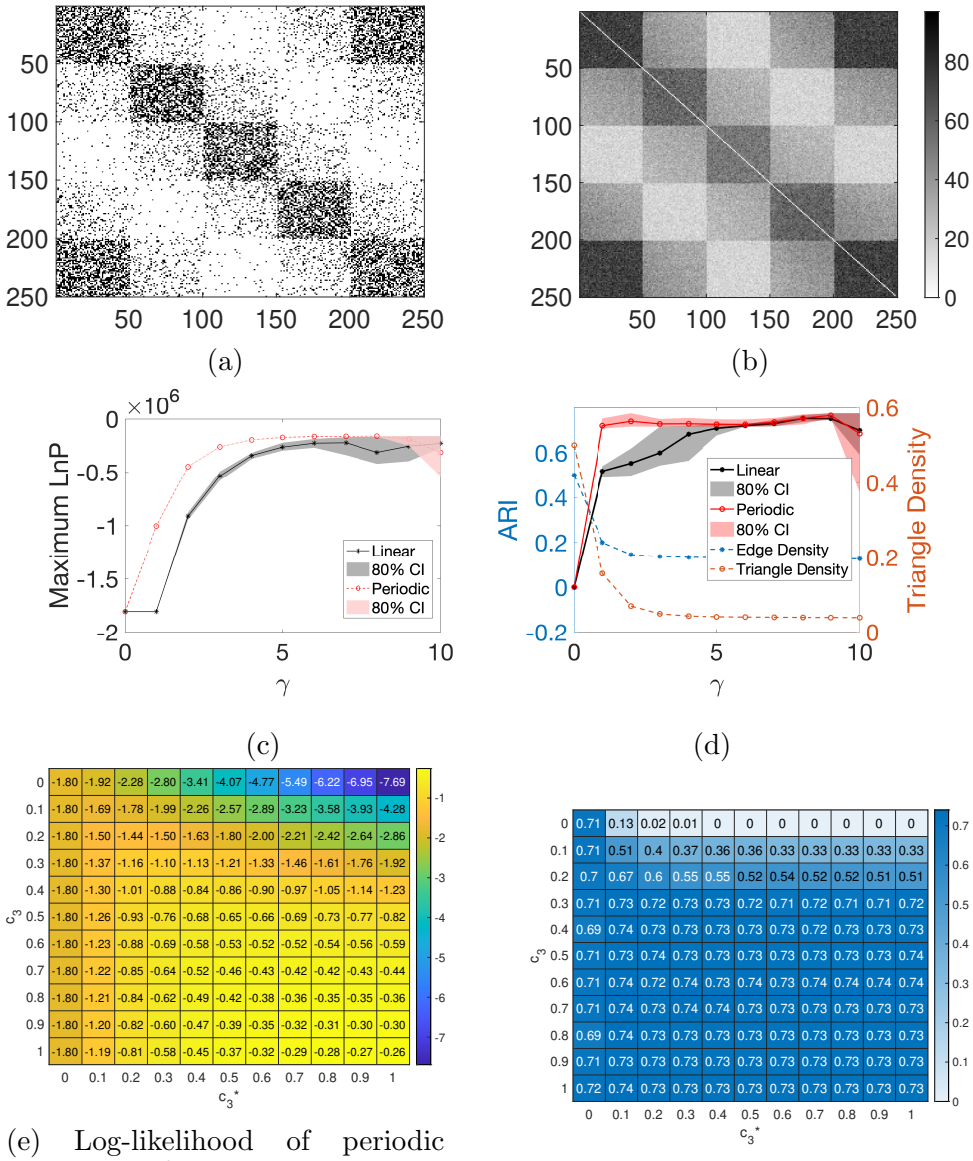


Figure 3.2: Model comparison experiments on synthetic periodic hypergraphs. (a) The bottom-left and top-right blocks in the dyadic adjacency matrix ($\gamma_0 = 1$) reveal a periodic clustered structure. (b) Similar periodic clustered pattern is shown in the triadic adjacency matrix ($\gamma_0 = 1$). (c) The periodic embedding achieves higher log-likelihoods than the linear embedding for different values of γ_0. (d) Adjusted Rand Indices of K-means clustering based on the linear and periodic embedding are plotted against γ_0. (e) Values in the heatmap indicate the maximum likelihoods of the periodic model $\times 10^{-6}$ from Algorithm 6 and the maxima lie on the diagonal where $c_3^* = c_3$. (f) Values represent the Adjusted Rand Indices of K-means clustering based on the periodic embedding for different values of c_3^* and c_3.

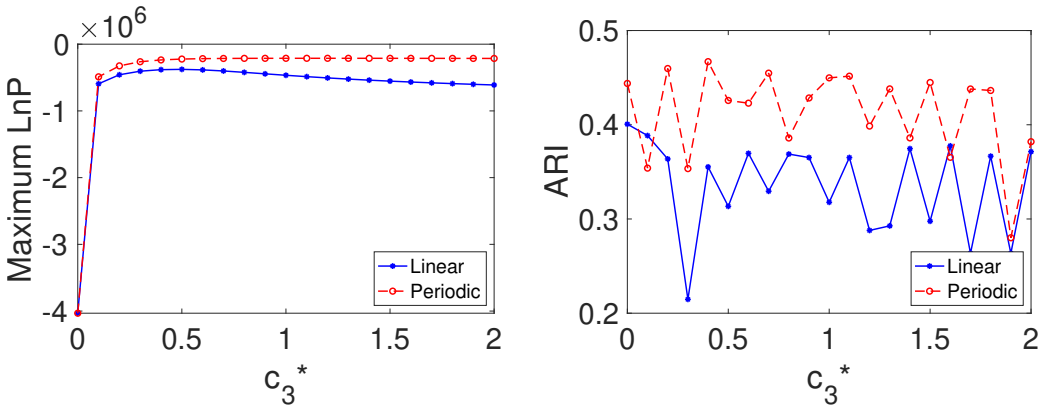


Figure 3.3: Model comparison experiments on the high school contact data. Left side shows the maximum log-likelihoods of the linear and periodic embedding for different values of c_3^* where pentagrams indicate maxima. Right side plots the Adjusted Rand Indices of K-means clustering based on the linear and periodic embedding for various c_3^*.

Real Hypergraphs

High School Contact Data The high school contact data from [75] records the frequency of student interaction. Students are represented as nodes, and contacts between two or three students are registered as dyadic or triadic edges. We retrieved the hypergraph from [16] containing 327 nodes, and only studied its dyadic and triadic edges considering the computational complexity. We construct the hypergraph Laplacian $L = c_2^* L^{[2]} + c_3^* L^{[3]}$ and perform linear and periodic spectral embedding. For the linear embedding we map nodes into 3-dimensional Euclidean space using the eigenvectors corresponding to the three smallest eigenvalues that are larger than 0.01. We make this choice because the eigenvector associated with the smallest non-zero eigenvalue has only a few non-zero entries and leads to trivial clusters. We fix $c_2^* = 1$ and vary c_3^* since only the relative weight c_3^*/c_2^* matters in node embedding.

The maximum likelihoods and ARIs evaluated using various c_3^* are shown in the left column in Figure 3.5. The true clusters are defined by the classes the students came from. Overall the periodic embedding achieves higher likelihoods and ARIs despite the linear embedding involving more parameters. Since linear clusters tend to have more marginalized groups that are far from other clusters, our results may suggest a lack of marginalisation driven by class membership.

We note that setting $c_3^* = 0$ causes the algorithm to ignore triangles, and hence it reduces to classical spectral clustering. For the linear algorithm, we see that incorporating triadic edges by using a positive c_3^* can improve the ARI by up to around 0.09. We note that in [16], modularity maximization-based clustering achieved ARI=1 on the same data. However, those methods have more parameters, which makes the ARI not directly comparable.

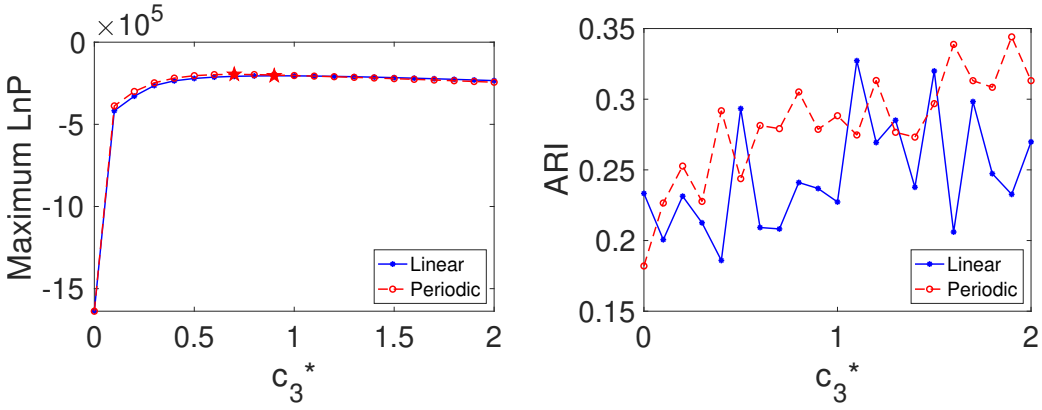


Figure 3.4: Model comparison experiments on the primary school contact data. Left side shows the maximum log-likelihoods of the linear and periodic embedding for different values of c_3^* where pentagrams indicate maxima. Right side plots the Adjusted Rand Indices of K-means clustering based on the linear and periodic embedding for various c_3^*.

Primary School Contact Data The primary school contact hypergraph [16] is derived from the contact patterns among students in 10 classes, as documented in [101]. Nodes represent students or teachers, and hyperedges represent their physical contact. Each node is labelled by the class of the students or as a teacher. The hypergraph contains 242 nodes and 11 classes of labels. We extracted the dyadic and triadic edges from the hypergraph and performed likelihood comparison and clustering with four eigenvectors associated with the four smallest eigenvalues that are greater than 0.01. The middle column in Figure 3.5 suggests the periodic embedding achieves the maximum likelihood at $c_3^* = 0.7$ and overall performs better than the linear embedding in the clustering task. These results may be related to the existence of a teacher group that connects with all student groups. When we arrange dyadic and triadic adjacency matrices by node classes, these connections will appear as off-diagonal entries. As we have shown in Figure 3.2a and 3.2b, the periodic model tends to produce more off-diagonal connections than the linear model.

Senate Bills Data In the senate bills hypergraph [32, 33, 16], nodes are US Congresspersons and hyperedges are the sponsor and co-sponsors of Senate bills. There are in total 294 nodes, and each node is labelled as either Democrat or Republican. We performed likelihood comparison and clustering with only the dyadic and triadic edges. Since the node degree distribution is highly inhomogeneous, we observe many trivial eigenvectors that are close to indicator functions. To address this issue we trimmed off the top and bottom 2% nodes by node degree, and use the eigenvector associated with the smallest eigenvalue that is greater than 0.01. The linear and periodic models have similar maximum likelihoods and clustering ARIs, as shown in the right column in Figure 3.5. In contrast with previous examples, there are only two clusters present in this data set. Hence the difference between the periodic and linear models, which can be reflected in the

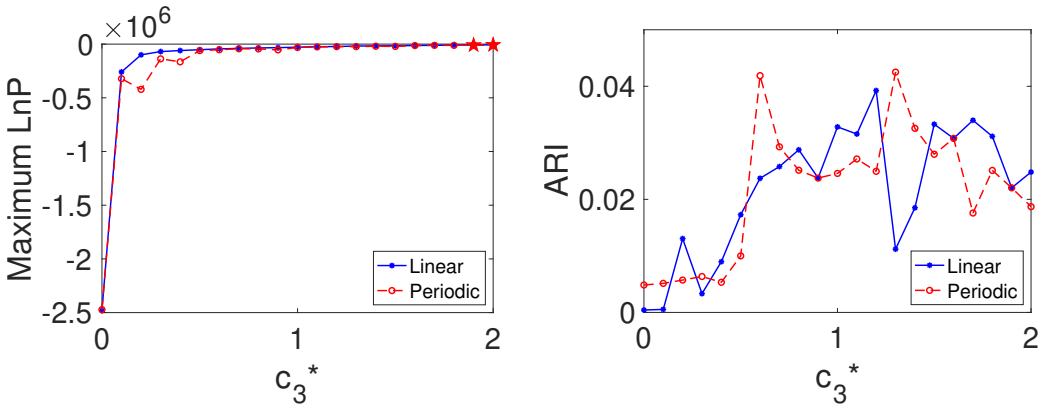


Figure 3.5: Model comparison experiments on the senate bill data. Left side shows the maximum log-likelihoods of the linear and periodic embedding for different values of c_3^* where pentagrams indicate maxima. Right side plots the Adjusted Rand Indices of K-means clustering based on the linear and periodic embedding for various c_3^*.

connection (or disconnection) pattern between the first and the last group, is less prominent.

3.6.2 Hyperedge Prediction

Once the node embeddings are estimated from the spectral algorithms, the probability of hyperedges may be computed from the proposed models. The hyperedge probability can naturally serve as a score for hyperedge prediction. We implement and test such triadic edge prediction on timestamped high school contact data [2, 75] , primary school contact data [2, 101], and synthetic linear hypergraphs. The results will be compared against approaches based on average-scores proposed in [2]. Other hyperedge prediction methods include feature-based prediction [114], model-based prediction [95], and machine learning-based prediction [113].

For high school and primary school contact data, we used three and four eigenvectors respectively corresponding to the smallest eigenvalues that are greater than 0.01 to be consistent with the previous section, and only consider dyadic and triadic edges. The hyperedges are sorted by time stamps and split into training and testing data. For example, an 80 : 20 training/testing splitting ratio means we use the first 80% of the hyperedges to train the model and the last 20% to test the predictions. When the training ratio is low, the subgraph for training may be disconnected thus violating Assumption 3.3.1. Therefore, we only consider nodes in the largest connected component of the graph associated with the binarized version of L of the training subgraph, and test the prediction on the same set of nodes. Note that in real data, the parameters γ_0, c_2 and c_3 for the hypergraph model are unknown. We fix $c_2^* = 1$ and choose c_3^* and γ^* using a maximum likelihood estimate through a grid search on the training data.

On the training set, we assign scores to each triplet using five methods: a

random score as baseline, hyperedge probability from the linear model, arithmetic mean, harmonic mean, and geometric mean from [2]. On the test set, we measure the prediction performance with the area under precision-recall curve (AUC-PR) [25]. A Precision-Recall (PR) curve traces the Precision = True Positive/ (True Positive + False Positive) and Recall = True Positive / (True Positive + False Negative) for different thresholds. The AUC-PR is a measure that balances both Precision and Recall where 1 means perfect prediction at any threshold. Setting $c_3^* = 0$ will assign probability of 0.5 to all triplets, which is equivalent to the random score approach if we break ties randomly.

On the high-school contact data shown in Table 3.1, the harmonic and geometric mean attain the highest AUC-PR for large amounts of training data, see, for example the 80 : 20 data split; while the linear model predictions achieve the best results for small amounts of training data, as seen in the 20 : 80 data split. This could be because when training data is insufficient, there are more unobserved "missing" dyadic and triadic edges. In this case, the node embedding algorithm can infer node proximity based on common neighbors. In other words, it can place nodes with common neighbors nearby even if they haven't been directly linked before. On the other hand, the geometric and harmonic mean will assign a score of zero to a triplet if none of the nodes has been connected previously, and therefore will predict no triadic edges.

We also test the triadic edge prediction on the synthetic linear hypergraphs, generated in the manner described in subsection 3.6.1, with $K = 4$, $m = 60$, $\gamma = 10$, $c_2 = 1$, and $c_3 = 0.3$, such that the clustered pattern resembles the high-school contact data. We consider three eigenvectors associated with the smallest eigenvalues that are greater than 0.01 for the synthetic linear model. Since defining a periodic model with more than one eigenvector is beyond the scope of this work, we only test the linear hypergraphs. We randomly select a portion of the hyperedges as the training set, while ensuring the sampled hypergraph is connected, and test the performance on the rest of the hyperedges. The AUC-PR averaged over 20 random hypergraphs is shown in Table 3.1. We observe that the linear model outperforms random score and average scores for various training data sizes.

3.7 Conclusion

In this chapter we have developed new random models and embedding algorithms for hypergraphs, and investigated their equivalence. In particular, we focused on two spectral embedding algorithms customized for hypergraphs, which aim to reveal linear and periodic structures, respectively. We also described random hypergraph models associated with these algorithms, which allow us to quantify the relative strength of linear and periodic structures based on maximum likelihood. We demonstrated the model comparison approach on synthetic linear and periodic hypergraphs, showing that the results are consistent with the generating mechanism. When applied to high school and primary school contact hypergraphs, the model comparison suggests the periodic structure is more prominent. On this data set we also showed that the "spectral embedding plus random hypergraph"

Data (Train:Test)	Random	Linear	Arith Mean	Geo Mean	Harm Mean
Highschool (80:20)	5.3e-5	9.3e-4	9.4e-4	**6.2e-3**	6.0e-3
Highschool (60:40)	1.2e-4	1.2e-3	2.1e-3	**1.1e-2**	**1.1e-2**
Highschool (20:80)	2.6e-4	**9.8e-3**	3.9e-3	7.6e-3	7.4e-3
Primary School (80:20)	3.2e-4	8.4e-3	3.3e-3	1.6e-2	**1.7e-2**
Primary School (60:40)	1.1e-3	1.2e-2	9.9e-3	**2.2e-2**	**2.2e-2**
Primary School (20:80)	1.6e-3	**3.0e-2**	1.6e-2	2.2e-2	2.1e-2
Linear Model (80:20)	6.4e-3	**1.6e-1**	4.1e-2	7.8e-2	7.7e-2
Linear Model (60:40)	1.3e-2	**2.8e-1**	8.5e-2	1.8e-1	1.8e-1
Linear Model (20:80)	3.2e-2	**3.0e-1**	1.1e-1	1.4e-1	1.3e-1

Table 3.1: AUC-PR for triangle prediction on high-school contact data and synthetic hypergraphs from the linear model. Highest values are indicated in bold.

approach gives a useful strategy for predicting new hyperedges.

Chapter 4

Magnetic Hodge Laplacian for Simplicial Complexes

4.1 Introduction

A simplicial complex is an extension of a graph that captures interactions between any number of nodes. It comprises simplexes of various dimensions, which can be seen as a generalization of edges to any dimensions. A simplicial complex can also be viewed as a hypergraph with downward face closure properties. This concept will be discussed in further depth in the upcoming section.

Originally developed in the field of algebraic topology, simplicial complexes provide a powerful toolkit for analyzing the topology of data [80]. Recently, simplicial complexes have received growing interest due to increased availability of complex structured data and improved computation power, and have found applications in fields such as neuroscience [39, 87], biology [11, 13, 81, 112], signal processing [92, 94], sensor networks [27, 37], computer graphics [9, 116] and social contagion [57].

Similar to hypergraphs, simplicial complexes are capable of encoding relations between more than two components, a feature not readily achievable with graphs due to their limitation to pairwise relationships. A primary benefit of choosing simplicial complexes over hypergraphs is the possibility of applying existing theories and computational methodologies, drawn from algebraic topology to real-world data sets [12, 14, 83]. A popular metric in this domain is the homology group, which, in simple terms, measures the number of loops, holes, and cavities within the data. The homology group can be computed using the Hodge Laplacian on simplicial complexes [35], an extension of the graph Laplacian to any dimension. Beyond providing insights into the shape of the data, the Hodge Laplacian can also be linked with higher-order random walks and used to examine the dynamical process on simplicial complexes [93].

Despite the rapid progress in the research of simplicial complexes, the majority of the existing methods consider undirected simplexes. Note that although simplexes are oriented, such orientations are induced by an arbitrary choice of node order, which does not influence the computational results. Therefore, Hodge Laplacian with oriented simplexes can not capture the directional information.

So far theories on directed simplicial complexes are still less explored, partly due to their intricate combinatorial nature. In Chapter 2, we introduced the Magnetic Laplacian on directed graphs defined in terms of the adjacency matrix and a phase angle associated with edge directions. Such a Hermitian operator can be useful for node embedding with directed graphs [30], identification of directed clusters [31], and synchronization of connected sensors [100]. In this chapter, our goal is to define and analyze a new first-order Magnetic Hodge Laplacian that extends the Magnetic Laplacian to simplicial complexes respecting the direction of flows on both edges and triangles. This chapter is exploratory in nature, aiming to lay the groundwork for future investigations and research.

The main contributions of this chapter are as follows.

- We introduce the first-order Magnetic Hodge Laplacian which can represent directional flow on both edges and triangles.
- We analyze the quadratic form and spectrum of the newly introduced first-order Magnetic Hodge Laplacian across four distinct cases of directed triangles.
- We demonstrate the tool by applying it to two different types of triangulated tori, where the eigenvectors of the Magnetic Hodge Laplacian recover directional information.

The rest of the chapter is structured as follows. In the subsequent section, we provide the theoretical background for simplicial complexes. Following that, in Section 4.3, we introduce the new concept of the Magnetic Hodge Laplacian. Basic notations used throughout this chapter are presented in Section 4.3. We then proceed to define a new boundary operator to construct the Magnetic Hodge Laplacian in Section 4.3.1. Although this construction is consistent with the zero-order Magnetic Laplacian, it fails to distinguish certain configurations involving the direction of triangles. Therefore, in Section 4.3.2, we shift gears and utilize an element-by-element formalism to construct the first-order Magnetic Laplacian. In Section 4.4 and 4.5, we present case studies focused on directed triangles and tori. In these case studies, we demonstrate the application of the first-order Magnetic Hodge Laplacian and analyze its spectral properties. Finally, we conclude the chapter with a summary and discussion in Section 4.6.

4.2 Background

4.2.1 Simplicial Complex

In this thesis, we will use the notion of abstract simplicial complexes without reference to their realization in Euclidean space. However, here we will omit the word “abstract” in the majority of the thesis for readability. We will first define an abstract simplex, which serves as the fundamental building block of a simplicial complex. Most definitions are based on textbooks on algebraic topology such as [10, 27, 38, 80].

Definition 4.2.1 ((Abstract) Simplex)**.** *Let $V = \{v_0, v_1, ..., v_n\}$ be a finite set of vertices. A k-simplex σ_k is a subset of V with $k+1$ distinct elements.*

We refer to k as the **dimension** of σ_k. Simplexes with dimensions 0, 1, 2, and 3 are commonly referred to as **vertices**, **edges**, **triangles**, and **tetrahedra**, respectively.

Definition 4.2.2 (Face and Co-face)**.** *A **face** of σ_k is a $(k-1)$-simplex comprised of a subset of vertices of σ_k. Similarly, σ_k is called a **co-face** of σ_{k-1} when σ_k contains σ_{k-1} as one of its faces. We use $deg_U(\sigma)$ to represent the number of $(k+1)$-simplices in which σ_k is a face.*

Definition 4.2.3 (Upper and Lower Adjacent)**.** *Two distinct k-simplices are **upper adjacent** if they share a common co-face. Two distinct k-simplices are **lower adjacent** if they share a common face.*

Now we define a simplicial complex, which is constructed from a set of simplexes following certain rules.

Definition 4.2.4 ((Abstract) Simplicial Complex)**.** *A simplicial complex $\mathcal{S}$ is a collection of finite simplexes such that (1) if $\mathcal{A} \in \mathcal{S}$, every nonempty face of $\mathcal{A}$ is also an element of $\mathcal{S}$ and (2) the intersection of any two simplexes is a face of both of those simplexes.*

The **dimension** of $\mathcal{S}$ is the maximum dimension of its elements. The vertex set V of $\mathcal{S}$ is the union of the elements of $\mathcal{S}$ with dimension 0.

Corollary 4.2.1. *Let σ_k^1, σ_k^2 be two distinct k-simplexes for an integer $k > 0$ in a simplicial complex $\mathcal{S}$, then the following statements hold:*

1. *If σ_k^1 and σ_k^2 are upper-adjacent, they must be lower adjacent.*
2. *If σ_k^1 and σ_k^2 are lower-adjacent, their common face is unique.*
3. *If σ_k^1 and σ_k^2 are upper adjacent, their common co-face is unique.*

Proof. The first statement is a direct consequence of property (1) in Definition 4.2.4. The second statement can be verified using property (2) in Definition 4.2.4. We can prove the third statement by noting that two faces uniquely determine a simplex. □

For numerical computation, it is convenient to associate each simplex with an orientation. This is analogous to assigning a label to each node for graphs. This leads to the following definition for an oriented simplex.

Definition 4.2.5 (Oriented Simplex)**.** *Let σ_k be a k-simplex, an orientation for σ_k is an ordering for its vertices. Two orderings are equivalent if they differ by an even permutation. An oriented k-simplex is a k-simplex σ_k together with its orientation.*

In this thesis, we write $\sigma_k = [v_0, ..., v_k]$ to denote that σ_k is an oriented k-simplex with vertices $\{v_0, ..., v_k\}$ and orientation given by the shown ordering and all even permutations of it. When $k > 0$, we denote $-\sigma_k$ as the oriented simplex consisting of the same set of nodes as σ_k subject to an odd permutation of the nodes, that is,

$$[v_0, ..., v_k] = -[v_{\pi(0)}, ..., v_{\pi(k)}] \text{ for an odd permutation } \pi,$$

therefore $-(-\sigma_k) = \sigma_k$ by definition. Similarly, an **oriented simplicial complex** is a simplicial complex where each simplex has an assigned orientation.

Definition 4.2.6 (Chain Group)**.** *[38] Let $\mathcal{S}$ be an oriented simplicial complex, with k-simplexes $\sigma_k^1, ..., \sigma_k^n$. The k-th chain group $C_k(\mathcal{S})$ is the free abelian group on the set $\{\sigma_k^1, ..., \sigma_k^n\}$. An element c_k of $C_k(\mathcal{S})$, called called a k-chain, can be written as a linear combination of k-simplexes in $\mathcal{S}$*

$$c_k = \lambda_1 \sigma_k^1 + ... + \lambda_n \sigma_k^n,$$

for some integers $\lambda_1, ..., \lambda_n$. Furthermore, we define the sum of two k-chains $c_k = \sum_{j=1}^n \lambda_j \sigma_k^j$ and $c_k' = \sum_{j=1}^n \lambda_j' \sigma_k^j$ as

$$c_k + c_k' = \sum_{j=1}^{n} (\lambda_j + \lambda_j') \sigma_k^j,$$

When writing down a k-chain, we often drop the simplexes with coefficient 0 and denote a k-chain in which all coefficients are zero as 0. More details on the free abelian groups can be found in, for example, the Appendix of [38].

Definition 4.2.7 (Boundary Map)**.** *Given the spaces of chains C_k, we define the k-boundary map $\delta_k : C_k \to C_{k-1}$ as follows:*

$$\delta_k([v_0, ..., v_k]) = \sum_{j=0}^{k} (-1)^j [v_0, ..., v_{j-1}, v_{j+1}, ..., v_k], \tag{4.1}$$

for $k \geq 1$. For $k = 0$, δ_k is defined to be zero.

Definition 4.2.8. *Let σ_k^1, σ_k^2 be two distinct upper-adjacent k-simplexes. If the signs of the coefficients of these two simplices in $\delta_{k+1}(\sigma_{k+1}^3)$ are the same, we say that σ_k^1 and σ_k^2 are similarly oriented with respect to σ_{k+1}^3, denoted as $\sigma_k^1 \sim_U \sigma_k^2$; if the signs of the coefficients are different, we say the simplices are dissimilarly oriented with respect to σ_{k+1}^3, denoted as $\sigma_k^1 \not\sim_U \sigma_k^2$.*

Definition 4.2.9. *Let σ_k^1, σ_k^2 be two distinct lower-adjacent k-simplexes. If the signs of the coefficients of σ_{k-1}^3 in $\delta_k(\sigma_k^1)$ and $\delta_k(\sigma_k^2)$ are the same, we say that σ_k^1 and σ_k^2 are similarly oriented with respect to σ_{k-1}^3, denoted as $\sigma_k^1 \sim_L \sigma_k^2$; if the signs of the coefficients are different, we say the simplices are dissimilarly oriented with respect to σ_{k-1}^3, denoted as $\sigma_k^1 \not\sim_L \sigma_k^2$.*

Lemma 4.2.1. *Let us consider two upper adjacent simplexes. They are similarly oriented with respect to their common co-face if and only if they are dissimilarly oriented with respect to their common face, and vice versa.*

The boundary map exhibits the property of a homomorphism; that is, the boundary map commutes with the group operation:

$$\delta_k(c_k + c'_k) = \delta_k c_k + \delta_k c'_k.$$

Therefore the boundary map is also called the **boundary homomorphism**. For each boundary map, there exists a coboundary map $\delta_k^* : C_{k-1} \to C_k$, which is the adjoint of the boundary map. From (4.1) we can verify that

$$\delta_k \delta_{k+1} = 0, \tag{4.2}$$

Thus we say the boundary of a boundary is zero. Equation (4.2) is sometimes referred to as the fundamental theorem of topology. Consequently, the image of δ_{k+1} is a subspace of the kernel of δ_k, or

$$\mathrm{im}(\delta_{k+1}) \subseteq \ker(\delta_k).$$

Here $\mathrm{im}(\cdot)$ denotes the image of an operator, and we call $\mathrm{im}(\delta_{k+1})$ the space of k-boundaries. $\mathrm{im}(\cdot)$ represents the kernel of an operator. It is straightforward to verify that $c_k \in C_k$ is a cyclic chain whose first and last vertices are identical, if and only if $\delta_k c_k = 0$. Therefore, $\ker(\delta_k)$ is the space of k-cycles. This leads to the definition of the k-th homology group as those elements in the null space of δ_k which are not in the image of δ_{k+1}.

Definition 4.2.10 (Homology Group)**.** *The k-th homology group of the chain complex $\mathcal{C}$ is defined as the quotient vector space*

$$H_k(\mathcal{C}) = ker(\delta_k)/im(\delta_{k+1}),$$

and its elements are called homology classes.

Note that $\ker(\delta_k)/\mathrm{im}(\delta_{k+1})$ has the structure of an abelian group under addition, hence the word "group". Loosely speaking, the group $H_k(\mathcal{C})$ measures the number of independent k-dimensional holes in $\mathcal{C}$.

As we have shown earlier, a boundary map δ_k is a linear map between the space of k-chains to the space of $(k-1)$-chains. If we choose the right basis vectors, the boundary map can be represented as matrices, simplifying computations. Each row of such matrix is indexed by a $(k-1)$-simplex, while each column is indexed by a k-simplex. The matrix entries are the coefficients in Equation (4.1).

Definition 4.2.11 (Matrix Representation of Boundary Operator)**.** *Let n_k and n_{k-1} be the number of k-simplexes and $(k-1)$-simplexes in the simplicial complex $\mathcal{S}$. The matrix representation of δ_k, denoted as $B_k \in \mathbb{R}^{n_{k-1} \times n_k}$, is given by*

$$[B_k]_{ij} = \begin{cases} 1, & \text{if } \sigma_{k-1}^i \sim \sigma_k^j \\ -1, & \text{if } \sigma_{k-1}^i \nsim \sigma_k^j \\ 0, & \text{otherwise,} \end{cases}$$

where $\sigma_{k-1}^i \sim \sigma_k^j$ denotes the $(k-1)$-simplex σ_{k-1}^i is a face of k-simplex σ_k^j with the same orientation induced by δ_k, and $\sigma_{k-1}^i \nsim \sigma_k^j$ denotes σ_{k-1}^i is a face of σ_k^j with the opposite orientation induced by δ_k.

Consequently, the matrix representation of δ_k^*, the Hermitian adjoint operator for δ_k, is B_k^T.

4.2.2 Hodge Laplacian on Simplicial Complex

The Hodge Laplacian on simplicial complexes [28, 67] generalizes the graph Laplacian to higher orders. It is also a discrete counterpart of the Laplace-de Rham operator in differential geometry [53].

Definition 4.2.12. *The Hodge k-Laplacian is defined as*

$$\Delta_k = \delta_k^* \delta_k + \delta_{k+1} \delta_{k+1}^*.$$

In a matrix representation, this can be written as

$$\mathcal{L}_k = B_{k+1} B_{k+1}^T + B_k^T B_k.$$

Definition 4.2.13. *For convenience, we define the k-up Hodge Laplacian as*

$$\mathcal{L}_k^{up} = B_{k+1} B_{k+1}^T$$

and the k-down Hodge Laplacian as

$$\mathcal{L}_k^{down} = B_k^T B_k,$$

such that

$$\mathcal{L}_k = \mathcal{L}_k^{up} + \mathcal{L}_k^{down}.$$

Proposition 4.2.1. *Consider a finite oriented simplicial complex $\mathcal{S}$, let $k \geq 0$ be an integer, and $\{\sigma_k^1, \sigma_k^2, ..., \sigma_k^n\}$ be the k-dimensional simplexes of $\mathcal{S}$. For $i, j \in \{1, 2, ..., n\}$, the corresponding entry in the Hodge Laplacian matrix can be expressed as follows:*

$$\mathcal{L}_k^{up} = \begin{cases} deg_U(\sigma_k^i), & \text{if } i = j \\ 1, & \text{if } \sigma_k^i \sim_U \sigma_k^i \\ -1, & \text{if } \sigma_k^i \not\sim_U \sigma_k^i \\ 0, & \text{otherwise;} \end{cases}$$

$$\mathcal{L}_k^{down} = \begin{cases} k+1, & \text{if } i = j \\ 1, & \text{if } \sigma_k^i \sim_L \sigma_k^i \\ -1, & \text{if } \sigma_k^i \not\sim_L \sigma_k^i \\ 0, & \text{otherwise.} \end{cases}$$

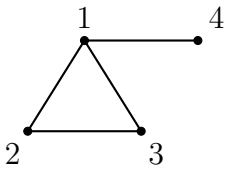


Figure 4.1: Examples of 2-dimensional simplicial complex

For $k > 0$ we have

$$\mathcal{L}_k = \begin{cases} deg_U(\sigma_k^i) + k + 1, & \text{if } i = j \\ 1, & \text{if } \sigma_k^i \sim_L \sigma_k^i \text{ and they are not upper adjacent} \\ -1, & \text{if } \sigma_k^i \not\sim_L \sigma_k^i \text{ and they are not upper adjacent} \\ 0, & \text{otherwise.} \end{cases}$$

Proof. We can compute $\mathcal{L}_k^{up}$ and $\mathcal{L}_k^{dow}$ directly using the boundary operators. Combined with Lemma 4.2.1 we can calculate $\mathcal{L}_k$. We leave out the detailed proof here and refer the interested reader to [41]. □

Example 4.2.1. *Let us consider an oriented simplicial complex shown in Figure 4.1 consists of node $[1], [2], [3], [4]$, edges $[1,2], [1,3], [3,4], [1,4]$, and triangle $[1,2,3]$. The boundary operators and Hodge Laplacians are given below.*

$$B_1 = \begin{array}{c} \\ [1] \\ [2] \\ [3] \\ [3] \end{array} \begin{array}{c} \begin{array}{cccc} [1,2] & [1,3] & [2,3] & [1,4] \end{array} \\ \begin{pmatrix} -1 & -1 & 0 & -1 \\ 1 & 0 & -1 & 0 \\ 0 & 1 & 1 & 0 \\ 0 & 0 & 0 & 1 \end{pmatrix} \end{array},$$

and

$$B_2 = \begin{array}{c} \\ [1,2] \\ [1,3] \\ [2,3] \\ [1,4] \end{array} \begin{array}{c} \begin{array}{c} [1,2,3] \end{array} \\ \begin{pmatrix} 1 \\ -1 \\ 1 \\ 0 \end{pmatrix} \end{array}.$$

Therefore

$$\mathcal{L}_1^{up} = B_2 B_2^T = \begin{pmatrix} 1 \\ -1 \\ 1 \\ 0 \end{pmatrix} \begin{pmatrix} 1 & -1 & 1 & 0 \end{pmatrix} = \begin{pmatrix} 1 & -1 & 1 & 0 \\ -1 & 1 & -1 & 0 \\ 1 & -1 & 1 & 0 \\ 0 & 0 & 0 & 0 \end{pmatrix},$$

and

$$\mathcal{L}_1^{down} = B_1^T B_1 = \begin{pmatrix} -1 & 1 & 0 & 0 \\ -1 & 0 & 1 & 0 \\ 0 & -1 & 1 & 0 \\ -1 & 0 & 0 & 1 \end{pmatrix} \begin{pmatrix} -1 & -1 & 0 & -1 \\ 1 & 0 & -1 & 0 \\ 0 & 1 & 1 & 0 \\ 0 & 0 & 0 & 1 \end{pmatrix} = \begin{pmatrix} 2 & 1 & -1 & 1 \\ 1 & 2 & 1 & 1 \\ -1 & 1 & 2 & 0 \\ 1 & 1 & 0 & 2 \end{pmatrix},$$

which sum up to

$$\mathcal{L}_1 = \mathcal{L}_1^{up} + \mathcal{L}_1^{down} = \begin{pmatrix} 3 & 0 & 0 & 1 \\ 0 & 3 & 0 & 1 \\ 0 & 0 & 3 & 0 \\ 1 & 1 & 0 & 2 \end{pmatrix}.$$

The Hodge decomposition theorem is a central result in the study of differential forms on smooth manifolds [78], and it has been adapted to discrete settings as below [59, 67].

Theorem 4.2.1 (Hodge Decomposition)**.** *The space of k-simplex signals can be decomposed into three orthogonal subspaces as*

$$\mathbb{R}^{n_k} = im(B_k^T) \oplus ker(\mathcal{L}_k) \oplus im(B_{k+1}).$$

This implies that, every $x \in \mathbb{R}^{n_k}$ can be decomposed uniquely as

$$x = B_{k+1}v + x_H + B_k^T w,$$

such that

$$\langle B_{k+1}w, x_H \rangle = \langle x_H, B_k^T v \rangle = \langle B_{k+1}w, B_k^T v \rangle = 0,$$

for some $v \in \mathbb{R}^{n_{k+1}}$ and $w \in \mathbb{R}^{n_{k-1}}$.

4.2.3 2-Manifolds

This thesis will consider examples of simplicial complexes abstracted from 2-dimensional manifolds, or 2-manifolds, which are some of the most common 2-dimensional spaces. Usually, we visualize them by embedding them into 3-dimensional spaces.

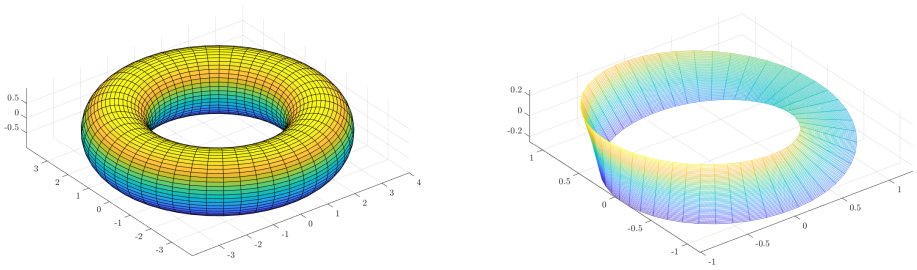
Definition 4.2.14 (Open Disk)**.** *Any subset of a topological space that is homeomorphic to*

$$D = \{x \in \mathbb{R}^2 \,|\, \|x\| < 1\} \tag{4.3}$$

is called an open disk.

This leads to the definition of the 2-manifold below.

Definition 4.2.15 (2-Manifold)**.** *A topological space $\mathbb{M}$ whose points all lie in open disks is a **2-manifold without boundary**. By removing open disks from 2-manifolds without boundary, one can get **2-manifolds with boundary**.*



(a) Torus (b) Möbius strip

Figure 4.2: Examples of 2-manifolds

The definition above means that a 2-manifold looks like a plane locally. Examples of 2-manifolds without boundary are the sphere, S^2 and the torus (Figure 4.2a), T^2; examples of 2-manifolds with boundary are the disk, the cylinder and the Möbius strip (Figure 4.2b).

Among the examples above, the Möbius strip has the intriguing property that it seems to have two sides locally but only one side globally. Such property can be demonstrated by considering a small, oriented circle inside the strip. If we move the small circle once around the strip, its orientation will be reversed. This path is therefore called a *orientation-reversing* closed curve. If we move the circle twice around the strip, the orientation will be preserved – such path is *orientation-preserving*. A 2-manifold is *orientable* if all closed curves in it are orientation-preserving; otherwise, it is called non-orientable.

For computation, we can triangulate a 2-manifold. The boundary circle of each disk is cut at three points and converted into triangular regions. All triangular regions can be represented as geometric triangles, which are arranged in a way that they share vertices and edges the same way as the triangular regions in the 2-manifold. Such representation is called a *triangulation* if it is homeomorphic to the 2-manifold. We also require that any two triangles are either disjoint, sharing an edge, or sharing one vertex. It is not allowed to have identical vertices in one triangle.

Once we get a triangulation of a 2-manifold $\mathbb{M}$, we can orient the triangles. If two triangles sharing an edge induce opposite orientations on the shared edge, they are *consistently oriented*. There is a way to arrange triangles such that all adjacent pairs are consistently oriented if and only if $\mathbb{M}$ is orientable.

4.3 Magnetic Hodge Laplacian

As described in Chapter 2, the Magnetic Laplacian is a Hermitian operator that incorporates phase angles to encode the direction of links, allowing for the visualization of directed flows on graphs. In the rest of the section, we aim to extend the Magnetic Laplacian to the first-order Magnetic Hodge Laplacian. The zero-order

Magnetic Hodge Laplacian should align with the Magnetic Laplacian of directed graphs formed solely by the nodes and directed links of the simplicial complex.

Notation

Let us consider an unweighted directed 2-dimensional simplicial complex $\mathcal{S}$ comprised of n nodes, links, and triangles. Let $V = \{1, ..., n\}$ be a finite set of vertices and $E \subseteq \binom{V}{2}$ is the set of edges, or 2-simplexes. Additionally, the set of triangles, or 2-simplexes, $T \subseteq \binom{V}{3}$. Furthermore, edges and triangles also have directions concordant or discordant with their orientation. Let $w_\sigma > 0$ indicate the weight associated with the direction aligned with the orientation of a simplex σ while $w_{-\sigma} > 0$ indicates the weight associated to the direction opposite to the orientation of σ. Then let

$$\phi_\sigma = w_\sigma - w_{-\sigma}$$

represent the net weight of the flow aligned with the orientation. If simplexes are *unweighted*, w_σ and $w_{-\sigma}$ take value of either 0 or 1. Therefore,

$$\phi_\sigma = \begin{cases} 1, & \text{if the direction of } \sigma \text{ aligns with its orientation} \\ -1, & \text{if the direction of } \sigma \text{ is opposite to its orientation} \\ 0, & \text{if both directions exist.} \end{cases}$$

Furthermore, we let

$$W_\sigma = (w_\sigma + w_{-_\sigma})/2$$

be the symmetrized edge weight.

4.3.1 Boundary Operator Formalism

First-order Magnetic Boundary Operator

Let us start by defining the zero-order Magnetic Hodge Laplacian L_0^M of a 2-dimensional simplicial complex in terms of the first-order Magnetic Boundary Operator B_1^M whose rows are indexed by nodes and columns indexed by edges. Similar to the Boundary Operator for the Hodge Laplacian, the signs of the matrix elements depend on the relative orientations between the nodes and edges. In addition, we would like to incorporate the direction of edges using a phase angle. We use $\ell = [v_1, v_2] \in E$ to denote an edge whose positive orientation is given by the node ordering $[v_1, v_2]$, and define the first-order Magnetic Boundary Operator as

$$[B_1^M]_{i\ell} = \begin{cases} -e^{\mathbf{i}\phi_\ell/2}, & \text{if } \exists j \in V/i : \ell = [i, j] \in E \\ e^{-\mathbf{i}\phi_\ell/2}, & \text{if } \exists j \in V/i : \ell = [j, i] \in E \\ 0, & \text{otherwise.} \end{cases}$$

Then the zero-order Magnetic Hodge Laplacian is defined as

$$\mathcal{L}_0^M = B_1^M (B_1^M)^\dagger.$$

Example 4.3.1. *Let us consider a network consisting of a single edge* $[1, 2]$. *The*

weight $w_{[1,2]}$ *is associated with the direction* $1 \to 2$*, while the weight* $w_{[2,1]}$ *is associated with the weight* $2 \to 1$*. We then have*

$$B_1^M = \begin{matrix} \\ [1] \\ [2] \end{matrix} \begin{matrix} [1,2] \\ \begin{pmatrix} -e^{\mathbf{i}(w_{[1,2]}-w_{[2,1]})} \\ e^{-\mathbf{i}(w_{[1,2]}-w_{[2,1]})} \end{pmatrix} \end{matrix},$$

and the zero-order Magnetic Laplacian is then given by

$$\begin{aligned} \mathcal{L}_0^M = B_1^M (B_1^M)^\dagger &= \begin{pmatrix} -e^{\mathbf{i}(w_{[1,2]}-w_{[2,1]})} \\ e^{-\mathbf{i}(w_{[1,2]}-\alpha_{[2,1]})} \end{pmatrix} \begin{pmatrix} -e^{-\mathbf{i}(w_{[1,2]}-w_{[2,1]})} & e^{\mathbf{i}(w_{[1,2]}-w_{[2,1]})} \end{pmatrix} \\ &= \begin{pmatrix} 1 & -e^{2\mathbf{i}(w_{[1,2]}-w_{[2,1]})} \\ -e^{-2\mathbf{i}(w_{[1,2]}-w_{[2,1]})} & 1 \end{pmatrix}. \end{aligned}$$

Consistency with the Magnetic Laplacian

The zero-order Magnetic Hodge Laplacian defined above is consistent with the Magnetic Laplacian defined in [30] shown in 2.2.1 up to some scaling factors. For example, let us consider the unweighted case where edge weights w_ℓ and $w_{-\ell}$ are either 0 or one. The node degree can be calculated as

$$d_i = \sum_{\ell \in E : i \in \ell} W_\ell.$$

Then if we modify the coefficient and phase angle in Equation (4.3.1) as

$$[B_1^M]_{i\ell} = \begin{cases} -\sqrt{W_\ell} e^{\mathbf{i}\phi_\ell \pi g}, & \text{if } \exists j \in V/i : \ell = [j,i] \in E \\ \sqrt{W_\ell} e^{-\mathbf{i}\phi_\ell \pi g}, & \text{if } \exists j \in V/i : \ell = [i,j] \in E \\ 0, & \text{otherwise,} \end{cases}$$

we will get the same Laplacian as Equation (2.4). This can be shown by writing out each term in $\mathcal{L}_0^M$

$$[\mathcal{L}_0^M]_{ij} = [B_1^M (B_1^M)^\dagger]_{ij} = \begin{cases} -d_i, & \text{if } i = j \\ -\frac{1}{2} e^{-2\mathbf{i}\pi g}, & \text{if } i \to j \\ -\frac{1}{2} e^{2\mathbf{i}\pi g}, & \text{if } j \to i \\ -1, & \text{if } i \leftrightarrow j \\ 0, & \text{otherwise.} \end{cases}$$

Higher-order Magnetic Boundary Operator

Now we extend the Magnetic boundary operator above to higher order. Let σ_{k-1}^p and σ_k^q be a $(k-1)$ and k-simplex respectively. If we extend Equation (4.3.1) to

higher order, we get

$$[B_k^M]_{pq} = \begin{cases} -e^{\mathbf{i}\phi_q/2}, & \text{if } \sigma_{k-1}^p \not\sim_U \sigma_k^q \\ e^{-\mathbf{i}\phi_q/2}, & \text{if } \sigma_{k-1}^p \sim_U \sigma_k^q \\ 0, & \text{otherwise.} \end{cases}$$

Then we can define the n-order Magnetic Laplacian as

$$\mathcal{L}_k^M = B_{k+1}^M (B_{k+1}^M)^\dagger + (B_k^M)^\dagger B_k.$$

We refer to the first part $B_{k+1}^M (B_{k+1}^M)^\dagger$ as the k-up Magnetic Hodge Laplacian, which can be written as

$$[B_{k+1}^M (B_{k+1}^M)^\dagger]_{ij} = \begin{cases} \sum_{\sigma_{k+1}^q : \sigma_k^i \subset \sigma_{k+1}^q} 1, & \text{if } i = j \\ -e^{\mathbf{i}\phi_q}, & \text{if } \sigma_k^i \not\sim_U \sigma_{k+1}^q, \sigma_k^j \sim_U \sigma_{k+1}^q \\ -e^{-\mathbf{i}\phi_q}, & \text{if } \sigma_k^i \sim_U \sigma_{k+1}^q, \sigma_k^j \not\sim_U \sigma_{k+1}^q \\ 1 & \text{if } \sigma_k^i, \sim_U \sigma_{k+1}^q, \sigma_k^j \sim_U \sigma_{k+1}^q \text{ or } \sigma_k^i \not\sim_U \sigma_{k+1}^q, \sigma_k^j \not\sim_U \sigma_{k+1}^q \\ 0, & \text{otherwise.} \end{cases} \tag{4.4}$$

Similarly, we define the k-down Magnetic Hodge Laplacian as follows:

$$[(B_k^M)^\dagger B_k^M]_{ij} = \begin{cases} \sum_{\sigma_{k-1}^q : \sigma_{k-1}^q \subset \sigma_k^i} 1, & \text{if } i = j \\ -e^{-2\mathbf{i}(\phi_i + \phi_j)}, & \text{if } \sigma_{k-1}^q \not\sim_L \sigma_k^i,\ \sigma_{k-1}^q \sim_L \sigma_k^j \\ -e^{2\mathbf{i}(\phi_i + \phi_j)}, & \text{if } \sigma_{k-1}^q \sim_L \sigma_k^i,\ \sigma_{k-1}^q \not\sim_L \sigma_k^j \\ 1, & \text{if } \sigma_{k-1}^q \sim_L \sigma_k^i,\ \sigma_{k-1}^q \sim_L \sigma_k^j \quad \text{or} \quad \sigma_{k-1}^q \not\sim_L \sigma_k^i, \sigma_{k-1}^q \not\sim_L \sigma_k^j \\ 0, & \text{otherwise.} \end{cases} \tag{4.5}$$

By examining Equation (4.4) and (4.5), it becomes apparent that both the up and down Magnetic Hodge Laplacians, defined using the boundary operator, have a limitation. Specifically, they are unable to differentiate between situations where two simplexes both share the same (or opposite) orientations with their shared face (or co-face). In order to overcome this limitation, we present a novel framework in the following section that is independent of the boundary operator.

4.3.2 Element by Element Formalism

In a directed graph, nodes possess a trivial direction; hence, we only need to consider the directions of the links. For instance, two possible edge directions exist between nodes i and j: $i \to j$ or $i \leftarrow j$. It is also worth noting that we allow these two directions to occur concurrently, that is, $i \leftrightarrow j$.

When dealing with a 2-dimensional simplicial complex, the number of possible scenarios increases. It is no longer solely about the directions of the edges; the directions of the 2-simplices, or triangles, also come into play. In the case of $i, j, k \in V$, where $[i, j, k]$ or any of its permutations form a 2-simplex, we represent its orientation as $\Delta_{ijk} = 1$ (Case 1, 3, 5, and 7 in Table 4.1), if, when we curl the fingers of our right hand along the flow, our thumb points outward from the

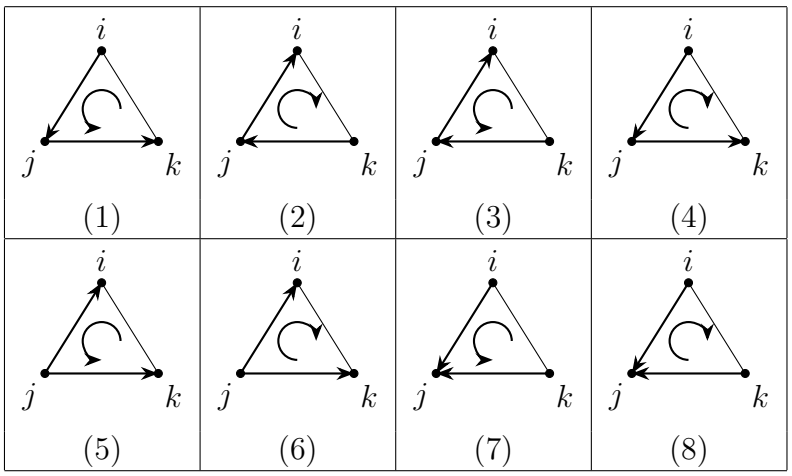


Table 4.1: 8 scenarios for $\mathcal{L}_1^{M,up}$

paper. Conversely, if the thumb points inward, we assign $\Delta_{ijk} = -1$ (Case 2, 4, 6, and 8 in Table 4.1). By considering all combinations of edge and triangle orientations, we encounter a total of eight potential configurations, illustrated in Table 4.1.

To distinguish these configurations, we employ the phase of complex numbers combined with the "higher-order rotation" induced by the Pauli matrices. This significantly enhances the complexity of our definition of the first-order Magnetic Hodge Laplacian. If we aim to construct a k-order Magnetic Hodge Laplacian of an order $d > 1$, the number of configurations will still be eight regardless of d.

1-up Magnetic Hodge Laplacian Consider a simplicial complex $\mathcal{S}$ with dimension $d \geq 2$. Let $\{\sigma_1^1, \sigma_1^2, ..., \sigma_1^n\}$ be its 1-simplexes. To construct the 1-up Hodge Magnetic Laplacian, we first define the *1-up degree matrix* $D_1^{up} \in \mathbb{R}^{n \times n}$ where

$$[D_1^{up}]_{ii} = 2 \deg_U(\sigma_1^i).$$

The value of $[D_1^{up}]_{ii}$ represents the number of edges that are upper-adjacent to edge i. The coefficient 2 arises from the fact that $\deg_U(\sigma_1^i)$ corresponds to the number of triangles that contain the edge σ_1^i. In each of these triangles, there are two other edges that are upper-adjacent to σ_1^i. There is no double counting because, as stated in Corollary 4.2.1, two upper-adjacent edges share a unique common triangle. Next, we isolate the off-diagonal entries of the Hodge Laplacian $\mathcal{L}_1^{up}$ to capture the orientations of edges. This is achieved by constructing the *1-up adjacency matrix*

$$A_1^{up} = D_1^{up} - \mathcal{L}_1^{up}.$$

From Proposition 4.2.1 we have

$$[A_1^{up}]_{lm} = \begin{cases} 0, & \text{if } l = m \\ -1, & \text{if } \sigma_1^l \sim_U \sigma_1^m \\ 1, & \text{if } \sigma_1^l \not\sim_U \sigma_1^m \\ 0, & \text{otherwise.} \end{cases}$$

Definition 4.3.1 (1-up Hodge Magnetic Laplacian)**.** *We define the 1-up Hodge Magnetic Laplacian as* $\mathcal{L}_1^{M,up}$ *as*

$$\mathcal{L}_1^{M,up} = D_1^{up} \otimes I_2 - T^{[1],up} \circ (A_1^{up} \otimes \mathbf{1}_2),$$

where $\mathbf{1}_2$ *is the* 2×2 *matrix having all elements equal to one, and the rotation matrix* $T^{[1],up}$ *is defined as*

$$T^{[1],up}_{(ij),(jk)} = \begin{cases} e^{-\mathbf{i}\delta} I_2, & \text{if } i \to j, j \to k, \Delta_{ijk} = 1 \quad (1) \\ e^{\mathbf{i}\delta} I_2, & \text{if } j \to i, k \to j, \Delta_{ijk} = -1 \quad (2) \\ e^{-\mathbf{i}\delta} \boldsymbol{\sigma}_x, & \text{if } j \to i, k \to j, \Delta_{ijk} = 1 \quad (3) \\ e^{\mathbf{i}\delta} \boldsymbol{\sigma}_x, & \text{if } i \to j, j \to k, \Delta_{ijk} = -1 \quad (4) \\ e^{-\mathbf{i}\delta} \boldsymbol{\sigma}_y, & \text{if } j \to i, j \to k, \Delta_{ijk} = 1 \quad (5) \\ e^{\mathbf{i}\delta} \boldsymbol{\sigma}_y, & \text{if } j \to i, j \to k, \Delta_{ijk} = -1 \quad (6) \\ e^{-\mathbf{i}\delta} \boldsymbol{\sigma}_z, & \text{if } i \to j, k \to j, \Delta_{ijk} = 1 \quad (7) \\ e^{\mathbf{i}\delta} \boldsymbol{\sigma}_z, & \text{if } i \to j, k \to j, \Delta_{ijk} = -1 \quad (8) \\ I_2, & \text{otherwise,} \end{cases}$$

for a constant $\delta \in [0, 2\pi)$*. Here,* $\boldsymbol{\sigma}$ *are the Pauli matrices*

$$\sigma_x = \begin{pmatrix} 0 & 1 \\ 1 & 0 \end{pmatrix}, \quad \sigma_y = \begin{pmatrix} 0 & -i \\ i & 0 \end{pmatrix} \quad \sigma_z = \begin{pmatrix} 1 & 0 \\ 0 & -1 \end{pmatrix}.$$

Theorem 4.3.1. *If we assign a complex vector* $\nu_l \in \mathbb{C}^2$ *to edge index by* l*, or* σ_1^l*, and let* $\boldsymbol{\nu} = \{\nu_1, ..., \nu_n\}$ *, then we have:*

$$\boldsymbol{\nu}^H \mathcal{L}_1^{M,up} \boldsymbol{\nu} = \frac{1}{2} \left(\sum_{\sigma_1^l \not\sim_U \sigma_1^m} \|\nu_l - T^{[1],up}_{lm} \nu_m\|^2 + \sum_{\sigma_1^l \sim_U \sigma_1^m} \|\nu_l + T^{[1],up}_{lm} \nu_m\|^2 \right).$$

Proof. This can be proven using a similar approach as outlined in Theorem 2.2.1 while taking into account the symmetric nature of A_1^{up} and the Hermitian property of $T^{[1],up}$. □

Remark 4.3.1. *Based on the theorem above, it becomes apparent that* $\mathcal{L}_1^{M,up}$ *is semi-definite positive. We can immediately see that* $\boldsymbol{\nu}^H \mathcal{L}_1^{M,up} \boldsymbol{\nu} = 0$ *when*

$$\nu_l = T^{[1],up}_{lm} \nu_m, \text{ for all } l, m \in \{1, ..., n\} \text{ such that } \sigma_1^l \not\sim_U \sigma_1^m, \tag{4.6}$$

and

$$\nu_l = -T^{[1],up}_{lm} \nu_m, \text{ for all } l, m \in \{1, ..., n\} \text{ such that } \sigma_1^l \sim_U \sigma_1^m. \tag{4.7}$$

Let us assign $\nu_p = (e^{\mathbf{i}\theta_p}, e^{\mathbf{i}\phi_p})^T \in \mathbb{C}^2$ *to edge* σ_1^p*. Now, consider two edges* $l = [i, j]$*,* $m = [j, k]$ *such that* $\sigma_1^l \not\sim_U \sigma_1^m$*. We use* $a \equiv b$ *to denote* $a = b + 2\pi q$ *where* q *can be any integer. Under these circumstances, the first condition (4.6) implies that*

1. $\theta_l \equiv \theta_m - \delta$*,* $\phi_l \equiv \phi_m - \delta$ *when* $i \to j, j \to k$*, and* $\Delta_{ijk} = 1$*;*

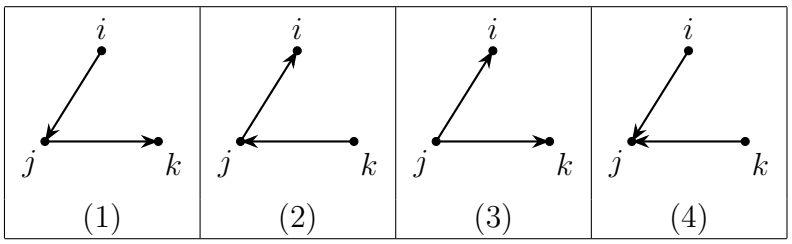


Table 4.2: 4 scenarios for $L_1^{M,down}$

2. $\theta_l \equiv \theta_m + \delta$, $\phi_l \equiv \phi_m + \delta$ *when* $j \to i, k \to j$, *and* $\Delta_{ijk} = -1$;

3. $\theta_l \equiv \phi_m - \delta$, $\phi_l \equiv \theta_m - \delta$ *when* $j \to i, k \to j$, *and* $\Delta_{ijk} = 1$;

4. $\theta_l \equiv \phi_m + \delta$, $\phi_l \equiv \theta_m + \delta$ *when* $i \to j, j \to k$, *and* $\Delta_{ijk} = -1$;

5. $\theta_l \equiv \phi_m - \frac{\pi}{2} - \delta$, $\phi_l \equiv \theta_m + \frac{\pi}{2} - \delta$ *when* $j \to i, j \to k$, *and* $\Delta_{ijk} = 1$;

6. $\theta_l \equiv \phi_m - \frac{\pi}{2} + \delta$, $\phi_l \equiv \theta_m + \frac{\pi}{2} + \delta$ *when* $j \to i, j \to k$, *and* $\Delta_{ijk} = -1$;

7. $\theta_l \equiv \theta_m - \delta$, $\phi_l \equiv \phi_m + \pi - \delta$ *when* $i \to j, k \to j$, *and* $\Delta_{ijk} = 1$;

8. $\theta_l \equiv \theta_m + \delta$, $\phi_l \equiv \phi_m + \pi + \delta$ *when* $i \to j, k \to j$, *and* $\Delta_{ijk} = -1$.

Similar results can be obtained for the case when $\sigma_1^l \sim_U \sigma_1^m$, *simply by adding* π *to the right-hand side.*

1-down Magnetic Hodge Laplacian Now, let us define a 1-down Magnetic Hodge Laplacian following a similar procedure considering 4 cases visualized in Table 4.2. First we extract the off-diagonal entries of $\mathcal{L}_1^{down}$ by defining the 1-down adjacency matrix

$$A_1^{down} = D_1^{down} - \mathcal{L}_1^{down},$$

where the 1-down degree matrix $D_1^{down} \in \mathbb{R}^{n \times n}$ is a diagonal matrix with entries

$$[D_1^{down}]_{ll} = \sum_{m=1}^{n} |[A_1^{down}]_{lm}|.$$

From Proposition 4.2.1 we have

$$[A_1^{down}]_{lm} = \begin{cases} -1, & \text{if } \sigma_1^l \sim_L \sigma_1^l \\ 1, & \text{if } \sigma_1^l \not\sim_L \sigma_1^m \\ 0, & \text{otherwise.} \end{cases}$$

Finally, the 1-down Magnetic Hodge Laplacian is defined by incorporating both the edge orientations with A_1^{down} and the edge directions using a rotation matrix $T^{[1],down}$.

Definition 4.3.2. *The 1-down Magnetic Hodge Laplacian denoted as* $\mathcal{L}_1^{M,down}$ *is defined as follows:*

$$\mathcal{L}_1^{M,down} = D_1^{down} \otimes I_2 - T^{[1],down} \circ (A_1^{down} \otimes \mathbf{1}_2),$$

where the 1-down rotation matrix $T^{[1],down}$ *is given by*

$$T^{[1],down}_{(ij),(jk)} = \begin{cases} e^{\mathbf{i}\delta} I_2, & \text{if } i \to j, j \to k \quad (1) \\ e^{-\mathbf{i}\delta} I_2, & \text{if } j \to i, k \to j \quad (2) \\ \boldsymbol{\sigma}_y, & \text{if } j \to i, j \to k \quad (3) \\ \boldsymbol{\sigma}_z, & \text{if } i \to j, k \to j \quad (4) \\ I_2, & \text{otherwise,} \end{cases}$$

for a constant $\delta \in [0, 2\pi)$.

The Hermitian property of $\mathcal{L}^{M,down}1$ can be easily shown since D_1^{down} and A_1^{down} are real and symmetric; and $T^{[1],down}$ is Hermitian.

Theorem 4.3.2. *If we assign a complex vector* $\nu_l \in \mathbb{C}^2$ *to edge* l*, and let* $\boldsymbol{\nu} = \{\nu_1, ..., \nu_n\}$ *, then we have:*

$$\boldsymbol{\nu}^H \mathcal{L}_1^{M,down} \boldsymbol{\nu} = \frac{1}{2} \left(\sum_{\sigma_1^l \not\sim_L \sigma_1^m} \|\nu_l - T^{[1],down}_{lm} \nu_m\|^2 + \sum_{\sigma_1^l \sim_L \sigma_1^m} \|\nu_l + T^{[1],down}_{lm} \nu_m\|^2 \right).$$

Proof. This statement can be proven in a similar manner as Theorem 2.2.1 considering the fact that A_1^{down} is symmetric and $T^{[1],down}$ is Hermitian. □

Remark 4.3.2. *According to the theorem mentioned earlier, we can conclude that* $\mathcal{L}_1^{M,down}$ *is positive semi-definite. The quadratic form* $\boldsymbol{\nu}^H \mathcal{L}_1^{M,down} \boldsymbol{\nu} = 0$ *when the following condition holds*

$$\nu_l = T^{[1],down}_{lm} \nu_m, \text{ for all } l, m \in \{1, ..., n\} \text{ such that } \sigma_1^l \not\sim_U \sigma_1^m, \tag{4.8}$$

and

$$\nu_l = -T^{[1],down}_{lm} \nu_m, \text{ for all } l, m \in \{1, ..., n\} \text{ such that } \sigma_1^l \sim_U \sigma_1^m. \tag{4.9}$$

Let us assign $\nu_p = (e^{\mathbf{i}\theta_p}, e^{\mathbf{i}\phi_p})^T \in \mathbb{C}^2$ *to edge* σ_1^p *and consider two edges* $l = [i, j]$, $m = [j, k]$ *such that* $\sigma_1^l \not\sim_L \sigma_1^m$*. In this scenario, the condition (4.8) leads to*

1. $\theta_l \equiv \theta_m + \delta$, $\phi_l \equiv \phi_m + \delta$ *when* $i \to j$, $j \to k$;
2. $\theta_l \equiv \theta_m - \delta$, $\phi_l \equiv \phi_m - \delta$ *when* $j \to i$, $k \to j$;
3. $\theta_l \equiv \phi_m - \frac{\pi}{2}$, $\phi_l \equiv \theta_m - \delta$ *when* $j \to i$, $j \to k$;
4. $\theta_l \equiv \theta_m$, $\phi_l \equiv \phi_m + \pi$ *when* $i \to j$, $k \to j$.

We can derive similar results for the situation when $\sigma_1^l \sim_L \sigma_1^m$ *by adding* π *to the right-hand side.*

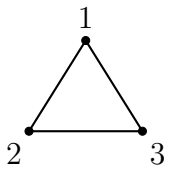


Figure 4.3

Figure 4.4: Undirected 2-simplex

4.4 Case Studies on Directed Triangles

Let us now explore some examples of directed triangles. We start by studying an undirected 2-dimensional simplicial complex as depicted in Figure 4.3, which consists of three nodes $[1], [2], [3]$, three edges $[1,2]$, $[1,3]$, $[2,3]$, and one triangle $[1,2,3]$. Subsequently, we can compute the boundary operators and Hodge Laplacians as illustrated below, where each row and column corresponds to a positively oriented simplex.

$$B_1 = \begin{array}{c} \\ [1] \\ [2] \\ [3] \end{array} \begin{array}{c} \begin{array}{ccc} [1,2] & [1,3] & [2,3] \end{array} \\ \begin{pmatrix} -1 & -1 & 0 \\ 1 & 0 & -1 \\ 0 & 1 & 1 \end{pmatrix} \end{array},$$

and

$$B_2 = \begin{array}{c} \\ [1,2] \\ [1,3] \\ [2,3] \end{array} \begin{array}{c} \begin{array}{c} [1,2,3] \end{array} \\ \begin{pmatrix} 1 \\ -1 \\ 1 \end{pmatrix} \end{array}.$$

Therefore

$$\mathcal{L}_1^{up} = B_2 B_2^T = \begin{pmatrix} 1 & -1 & 1 \\ -1 & 1 & -1 \\ 1 & -1 & 1 \end{pmatrix},$$

and

$$\mathcal{L}_1^{down} = B_1^T B_1 = \begin{pmatrix} 2 & 1 & -1 \\ 1 & 2 & 1 \\ -1 & 1 & 2 \end{pmatrix},$$

which sum up to

$$\mathcal{L}_1 = \mathcal{L}_1^{up} + \mathcal{L}_1^{down} = \begin{pmatrix} 3 & 0 & 0 \\ 0 & 3 & 0 \\ 0 & 0 & 3 \end{pmatrix}.$$

The 1-up degree matrix D_1^{up} and adjacency matrix A_1^{up} are

$$D_1^{up} = \begin{array}{c} \\ [1,2] \\ [1,3] \\ [2,3] \end{array} \begin{array}{c} \begin{array}{ccc} [1,2] & [1,3] & [2,3] \end{array} \\ \begin{pmatrix} 2 & 0 & 0 \\ 0 & 2 & 0 \\ 0 & 0 & 2 \end{pmatrix} \end{array},$$

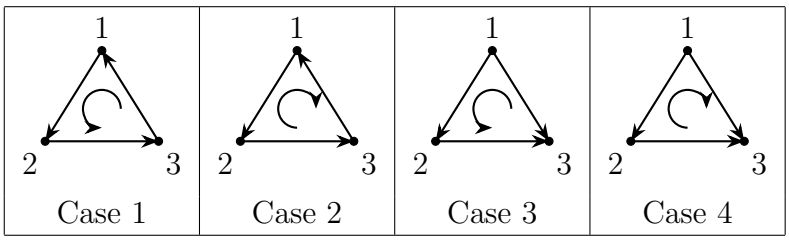


Table 4.3: 4 examples for directed triangles

and

$$A_1^{up} = \begin{array}{c} \\ [1,2] \\ [1,3] \\ [2,3] \end{array} \begin{array}{c} \begin{array}{ccc} [1,2] & [1,3] & [2,3] \end{array} \\ \begin{pmatrix} 0 & -1 & 1 \\ -1 & 0 & -1 \\ 1 & -1 & 0 \end{pmatrix} \end{array}.$$

The diagonal entries in D_1^{up} are twos since each edge shares the same triangle with two other edges. The elements of A_1^{up} are non-zero iff the two links are incident to the same triangle. Similarly, using the 1-down Hodge Laplacian $\mathcal{L}_1^{down}$ in Example 4.2.1 we have:

$$D_1^{down} = \begin{array}{c} \\ [1,2] \\ [1,3] \\ [2,3] \end{array} \begin{array}{c} \begin{array}{ccc} [1,2] & [1,3] & [2,3] \end{array} \\ \begin{pmatrix} 2 & 0 & 0 \\ 0 & 2 & 0 \\ 0 & 0 & 2 \end{pmatrix} \end{array},$$

and

$$A_1^{down} = \begin{array}{c} \\ [1,2] \\ [1,3] \\ [2,3] \end{array} \begin{array}{c} \begin{array}{ccc} [1,2] & [1,3] & [2,3] \end{array} \\ \begin{pmatrix} 0 & 1 & -1 \\ 1 & 0 & 1 \\ -1 & 1 & 0 \end{pmatrix} \end{array}.$$

The diagonal entries in D_1^{down} are twos because each edge shares a node with exactly two other edges. The elements of A_1^{down} are non-zero if and only if the two links are incident to the same node.

We have yet to take into account the directions of edges and triangles. In the subsequent section, we will build the first-order Magnetic Hodge Laplacian defined in Section 4.3.2 for the four potential scenarios in which direction becomes relevant, as illustrated in Table 4.3. We will plot their eigenvalues for various values of δ and analyze the commutator

$$[\mathcal{L}_1^{M,up}, \mathcal{L}_1^{M,down}] = \mathcal{L}_1^{M,up}\mathcal{L}_1^{M,down} - \mathcal{L}_1^{M,down}\mathcal{L}_1^{M,up}.$$

We are also interested in examining the quadratic form of the Magnetic Hodge Laplacian and determining the conditions under which it can be minimized to zero. To compute the quadratic form, we assign $\boldsymbol{\nu} = (\nu_1, \nu_2, \nu_3)^T \in \mathbb{C}^6$ to the three edges $[1,2]$, $[1,3]$, and $[2,3]$. Here, $\nu_p = (r_p e^{\mathbf{i}\theta_p}, s_p e^{\mathbf{i}\phi_p})^T \in \mathbb{C}^2$ corresponds to the p-th edge, with r_p, s_p being non-negative real numbers, and $\theta_p, \phi_p \in [0, 2\pi)$.

Laplacian matrices frequently appear in diffusion [64]. We will now examine

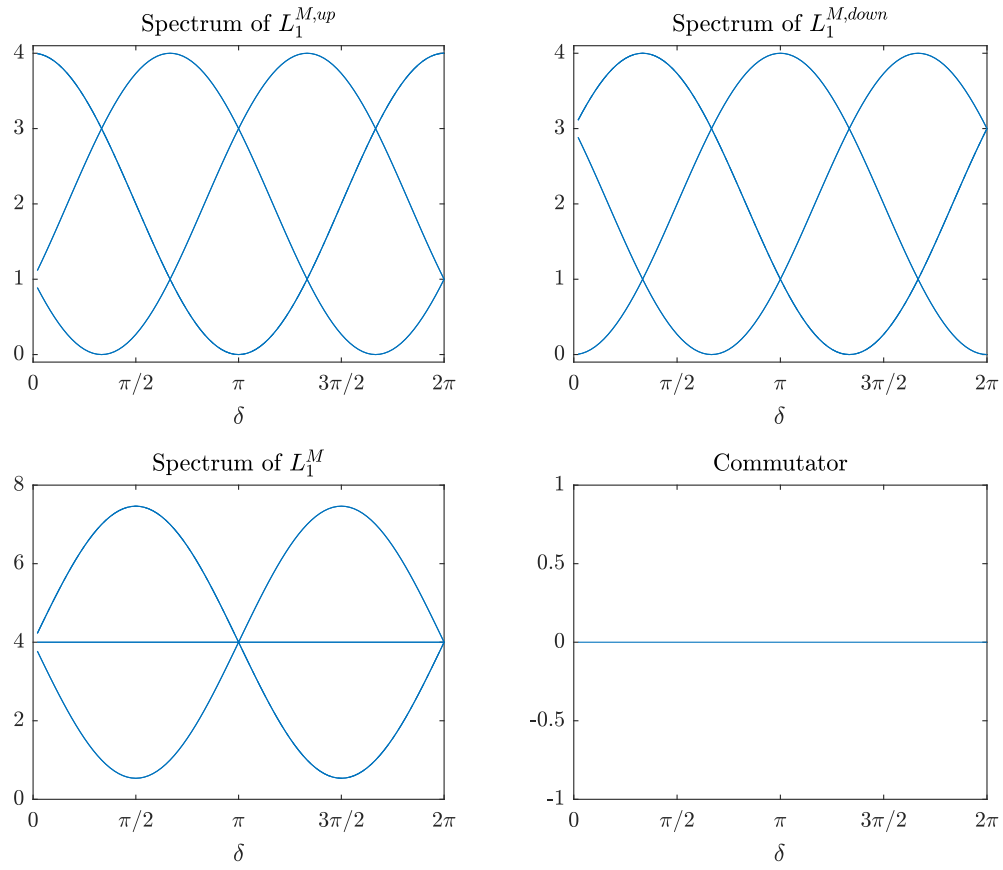


Figure 4.5: Eigenvalues of $\mathcal{L}_1^{M,up}$ (top-left), $\mathcal{L}_1^{M,down}$ (top-right), and $\mathcal{L}_1^{M}$ (bottom-left), and commutator $[\mathcal{L}_1^{M,up}, \mathcal{L}_1^{M,down}]$ (bottom-right) in Case 1

the dynamics of the diffusion governed by the following equations:

$$\frac{d\boldsymbol{\nu}(t)}{dt} = -\mathcal{L}_1^{M,up}\boldsymbol{\nu}(t),$$

$$\frac{d\boldsymbol{\nu}(t)}{dt} = -\mathcal{L}_1^{M,down}\boldsymbol{\nu}(t),$$

and

$$\frac{d\boldsymbol{\nu}(t)}{dt} = -(\mathcal{L}_1^{M,up} + \mathcal{L}_1^{M,down})\boldsymbol{\nu}(t).$$

We will plot the evolution of some initial states to study the diffusion process.

4.4.1 Case 1

Let us first consider the directed 2-simplicial complex displayed in the first column of Table 4.3, with edge directions given by $1 \to 2$, $2 \to 3$, and $3 \to 1$; and triangle direction $\Delta_{123} = 1$. This represents the scenario where all edge directions conform to the direction of the triangle. Hence, if we traverse from one edge to another, we either align with both the directions of the edge and the triangle, as shown in the first row of Table 4.3.1, or we go against both the edge and triangle directions, corresponding to the second row in Table 4.3.1. For the 1-up Magnetic Hodge

Laplacian, we obtain the following:

$$\mathcal{L}_1^{M,up} = \begin{matrix} \\ [1,2] \\ [1,3] \\ [2,3] \end{matrix} \begin{matrix} \begin{matrix} [1,2] & [1,3] & [2,3] \end{matrix} \\ \begin{pmatrix} 2I_2 & -I_2 e^{\mathbf{i}\delta} & I_2 e^{-\mathbf{i}\delta} \\ -I_2 e^{-\mathbf{i}\delta} & 2I_2 & -I_2 e^{\mathbf{i}\delta} \\ I_2 e^{\mathbf{i}\delta} & -I_2 e^{-\mathbf{i}\delta} & 2I_2 \end{pmatrix} \end{matrix}.$$

In the top left panel of Figure 4.5, we display the eigenvalues of $\mathcal{L}_1^{M,up}$ in relation to δ. The corresponding eigenvalues are:

$$\{2 + 2\cos(\delta), 2 - 2\cos(\delta - \pi/3), 2 + 2\cos(\delta - 2\pi/3)\}.$$

Moreover, its quadratic form can be computed as

$$\boldsymbol{\nu}^H \mathcal{L}_1^{M,up} \boldsymbol{\nu} = \|\nu_1 - I_2 e^{\mathbf{i}\delta}\nu_2\|^2 + \|\nu_1 + I_2 e^{-\mathbf{i}\delta}\nu_3\|^2 + \|\nu_2 - I_2 e^{\mathbf{i}\delta}\nu_3\|^2.$$

The quadratic form above equals zero when $r_1 = r_2 = r_3$, $s_1 = s_2 = s_3$, and

$$\begin{aligned} &\theta_1 \equiv \theta_2 + \delta,\ \phi_1 \equiv \phi_2 + \delta, \\ &\theta_1 \equiv \theta_3 - \delta + \pi,\ \phi_1 \equiv \phi_3 - \delta + \pi, \\ &\theta_2 \equiv \theta_3 + \delta,\ \phi_2 \equiv \phi_3 + \delta. \end{aligned}$$

The solution exists only when $\delta = \pi/3, \pi$, or $5\pi/3$, as depicted in the top-left panel of Figure 4.5. When δ assumes the aforementioned values, $\boldsymbol{\nu}$, which satisfy the above conditions, becomes an eigenvector of $\mathcal{L}_1^{M,up}$ associated with an eigenvalue of zero, according to the Rayleigh-Ritz Theorem. For the 1-down Magnetic Hodge Laplacian, we have

$$\mathcal{L}_1^{M,down} = \begin{matrix} \\ [1,2] \\ [1,3] \\ [2,3] \end{matrix} \begin{matrix} \begin{matrix} [1,2] & [1,3] & [2,3] \end{matrix} \\ \begin{pmatrix} 2I_2 & I_2 e^{-\mathbf{i}\delta} & -I_2 e^{\mathbf{i}\delta} \\ I_2 e^{\mathbf{i}\delta} & 2I_2 & I_2 e^{-\mathbf{i}\delta} \\ -I_2 e^{-\mathbf{i}\delta} & I_2 e^{\mathbf{i}\delta} & 2I_2 \end{pmatrix} \end{matrix}.$$

Eigenvalues of $\mathcal{L}_1^{M,down}$, as shown in the top-right plot in Figure 4.5, are

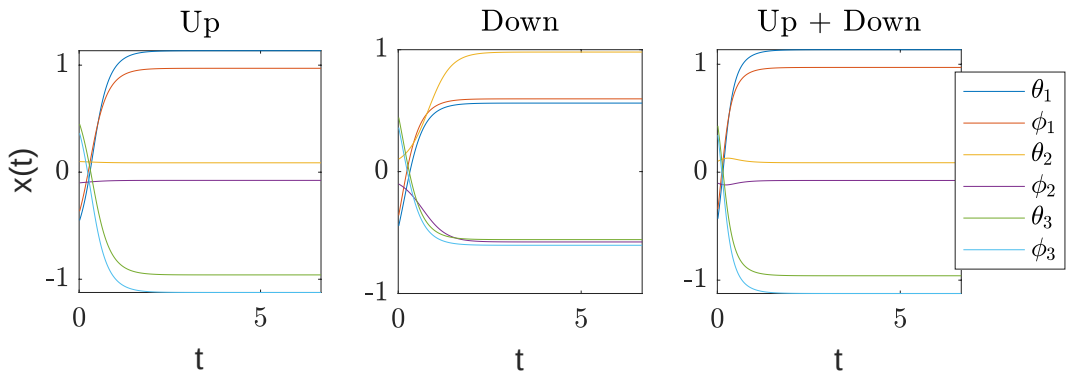
$$\{2 - 2\cos(\delta + \pi/3), 2 - 2\cos(\delta), 2 - 2\cos(\delta - \pi/3)\}.$$

The corresponding quadratic form is

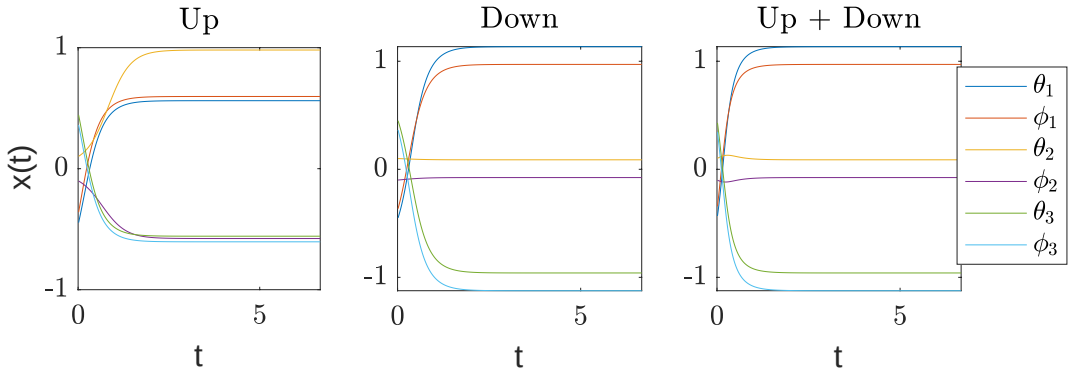
$$\boldsymbol{\nu}^H \mathcal{L}_1^{M,down} \boldsymbol{\nu} = \|\nu_1 + I_2 e^{-\mathbf{i}\delta}\nu_2\|^2 + \|\nu_1 - I_2 e^{\mathbf{i}\delta}\nu_3\|^2 + \|\nu_2 + I_2 e^{-\mathbf{i}\delta}\nu_3\|^2.$$

It becomes zero when $r_1 = r_2 = r_3$, $s_1 = s_2 = s_3$, and

$$\begin{aligned} &\theta_1 \equiv \theta_2 - \delta + \pi,\ \phi_1 \equiv \phi_2 - \delta + \pi, \\ &\theta_1 \equiv \theta_2 - \delta + \pi,\ \phi_1 \equiv \phi_3 + \delta, \\ &\theta_1 \equiv \theta_2 - \delta + \pi,\ \phi_2 \equiv \phi_3 - \delta + \pi. \end{aligned}$$



(a) $\delta = \pi/3$



(b) $\delta = 2\pi/3$

Figure 4.6: Diffusion on triangle in Case 1

The solution exists only when $\delta = 0, 2\pi/3$, or $4\pi/3$, as shown in the top-right panel in Figure 4.5. Upon combining the 1-up and 1-down Magnetic Hodge Laplacians, we obtain the eigenvalues of their sum, as shown in the bottom-left panel of Figure 4.5. The corresponding eigenvalues are

$$\{3, 3 + 2\sqrt{3}\sin(\delta), 3 - 2\sqrt{3}\sin(\delta)\}.$$

$\mathcal{L}_1^{M,up}$ and $\mathcal{L}_1^{M,down}$ commute since the commutator is zero (bottom-right panel in Figure 4.5). Furthermore, it can be shown that $\mathcal{L}_1^{down}\mathcal{L}_1^{up} \neq 0$, which implies that the Hodge decomposition does not hold. Figure 4.6 plots two examples when the dynamics are described by $\mathcal{L}_1^{M,up}$ (left), $\mathcal{L}_1^{M,down}$ (middle), and their sum (right) for various value of δ. For visualization purposes, we only display the phase angles. Setting $\delta = \pi/3$ (Figure 4.6a) and examining the plot for $\mathcal{L}_1^{M,up}$, we notice that it converges towards an eigenvector of $\mathcal{L}_1^{M,up}$ associated with an eigenvalue of zero, which satisfies the condition we previously discussed. Both $\mathcal{L}_1^{M,down}$ and $\mathcal{L}_1^{up} + \mathcal{L}_1^{M,down}$ only possess positive eigenvalues when $\delta = \pi/3$, therefore the vectors converge into their slow eigenmodes, associated with the smallest eigenvalue. We observe similar results when $\delta = 2\pi/3$ (Figure 4.6b),

where the final state in the middle plot lies in the null space of $\mathcal{L}_1^{M,down}$.

4.4.2 Case 2

Now, let us consider Case 2 in Table 4.3, where the edge directions are as follows: $1 \to 2$, $2 \to 3$, and $3 \to 1$; and the triangle direction is $\Delta_{123} = -1$. This scenario represents a situation where all edge directions are opposite to the direction of the triangle. Consequently, when traversing from one edge to another, we either move against the edge direction while aligning with the triangle direction, as shown in the third row of Table 4.3.1, or align with the edge direction while moving against the triangle direction, as depicted in the fourth row of Table 4.3.1. The corresponding 1-up Magnetic Laplacian is

$$\mathcal{L}_1^{M,up} = \begin{matrix} \\ [1,2] \\ [1,3] \\ [2,3] \end{matrix} \begin{matrix} \begin{matrix} [1,2] & [1,3] & [2,3] \end{matrix} \\ \begin{pmatrix} 2I_2 & -\boldsymbol{\sigma}_x e^{-\mathbf{i}\delta} & \boldsymbol{\sigma}_x e^{\mathbf{i}\delta} \\ -\boldsymbol{\sigma}_x e^{\mathbf{i}\delta} & 2I_2 & -\boldsymbol{\sigma}_x e^{-\mathbf{i}\delta} \\ \boldsymbol{\sigma}_x e^{-\mathbf{i}\delta} & -\boldsymbol{\sigma}_x e^{\mathbf{i}\delta} & 2I_2 \end{pmatrix} \end{matrix}.$$

Then the quadratic form $\boldsymbol{\nu}^H \mathcal{L}_1^{M,up} \boldsymbol{\nu}$ can be calculated as

$$\boldsymbol{\nu}^H \mathcal{L}_1^{M,up} \boldsymbol{\nu} = \|\nu_1 - \boldsymbol{\sigma}_x e^{-\mathbf{i}\delta} \nu_2\|^2 + \|\nu_1 + \boldsymbol{\sigma}_x e^{\mathbf{i}\delta} \nu_3\|^2 + \|\nu_2 - \boldsymbol{\sigma}_x e^{-\mathbf{i}\delta} \nu_3\|^2.$$

We plot its eigenvalues against δ in the top left panel of Figure 4.5. The corresponding eigenvalues are:

$$\{2 + 2\cos(\delta), 2 + 2\cos(\delta - \frac{\pi}{3}), 2 + 2\cos(\delta - \frac{2\pi}{3}), 2 + 2\cos(\delta - \pi),$$
$$2 + 2\cos(\delta - \frac{4\pi}{3}), 2 + 2\cos(\delta - \frac{5\pi}{3})\}.$$

It may be shown that $\boldsymbol{\nu}^H \mathcal{L}_1^{M,up} \boldsymbol{\nu} = 0$ when $r_1 = r_2 = r_3 = s_1 = s_2 = s_3$, and

$$\theta_1 \equiv \phi_2 - \delta,\ \phi_1 \equiv \theta_2 - \delta,$$
$$\theta_1 \equiv \phi_3 + \delta + \pi,\ \phi_1 \equiv \theta_3 + \delta + \pi,$$
$$\theta_2 \equiv \phi_3 - \delta,\ \phi_2 \equiv \theta_3 - \delta.$$

The equations above have a solution only when $\delta \in \{0, \pi/3, 2\pi/3, \pi, 4\pi/3, 5\pi/3\}$, as can be seen in the plot. In Case 2, $\mathcal{L}_1^{down}$ remains the same as in Case 1 since the edge directions are identical to the previous case. The eigenvalues of $\mathcal{L}_1^{up} + \mathcal{L}_1^{down}$ are

$$\{4 + 4\cos(\delta - \pi/3), 4 + 4\cos(\delta - \pi), 4 + 4\cos(\delta - 5\pi/3), 4\}.$$

The commutator $[\mathcal{L}_1^{down}, \mathcal{L}_1^{up}]$ is 0 as shown in the bottom-right in Figure 4.7. Hodge decomposition does not hold since it can be checked that $\mathcal{L}_1^{down}\mathcal{L}_1^{up} \neq 0$ for all values of δ. Figure 4.8 displays two examples of vector diffusion when $\delta = \pi/3$ and $2\pi/3$. When $\delta = 1\pi/3$, the vectors on the left converge towards an eigenvector of $\mathcal{L}_1^{up}$ related to eigenvalue zero. When $\delta = 2\pi/3$, the vectors

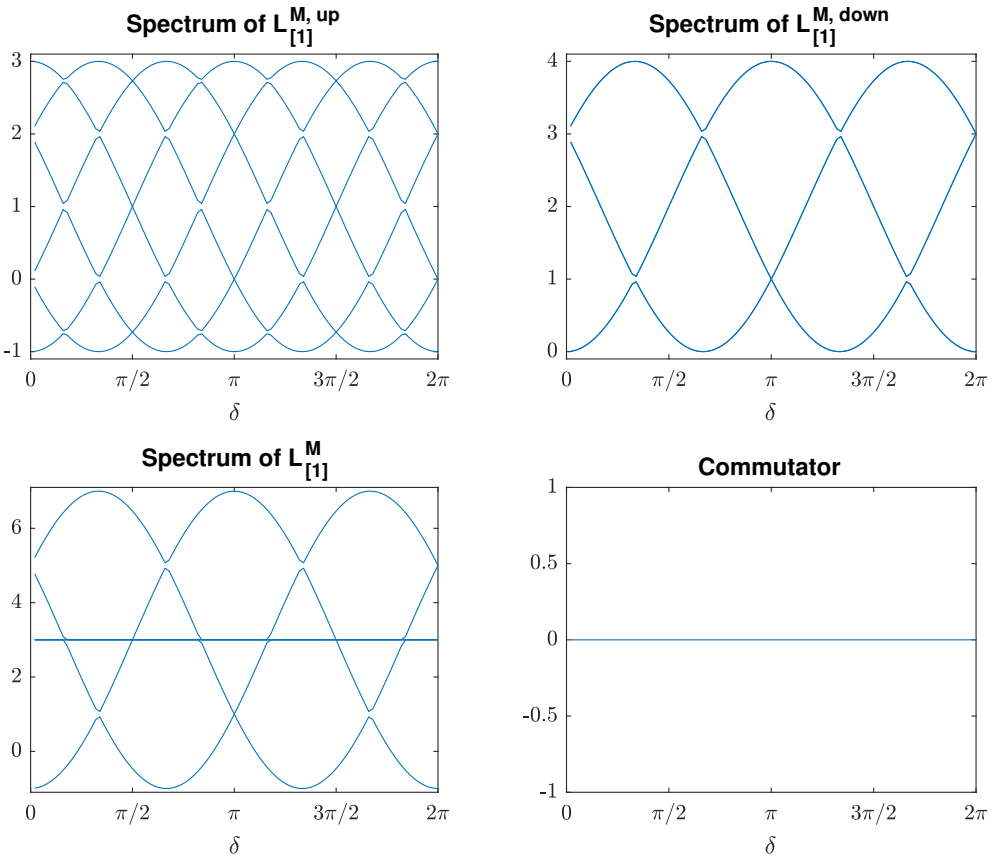


Figure 4.7: Eigenvalues of $\mathcal{L}_1^{M,up}$ (top-left), $\mathcal{L}_1^{M,down}$ (top-right), and $\mathcal{L}_1^{M,up}$ (bottom-left), and commutator $[\mathcal{L}_1^{M,up}, \mathcal{L}_1^{M,down}]$ (bottom-right) in Case 2

converge towards some eigenvectors for both $\mathcal{L}_1^{up}$ (on the left) and $\mathcal{L}_1^{down}$ (in the middle), each of which is linked to a zero eigenvalue.

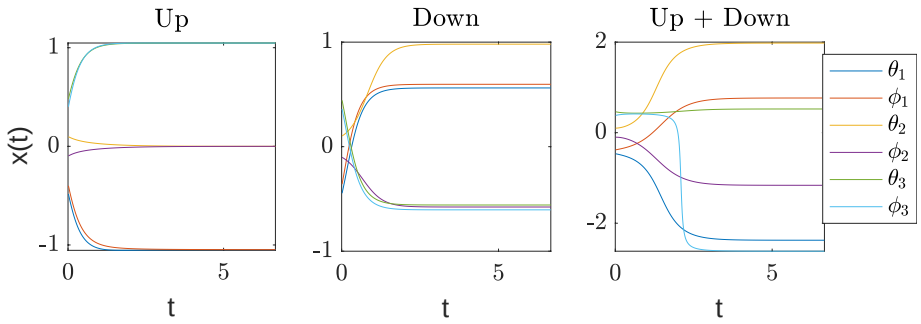
4.4.3 Case 3

Now, let us look at the case of a 2-simplicial complex with triangle and link direction given by $\Delta_{123} = 1$ and $1 \to 2$, $2 \to 3$ and $1 \to 3$, which is illustrated in the third column of Table 4.3. First, we can write down the 1-up and 1-down Magnetic Hodge Laplacian as

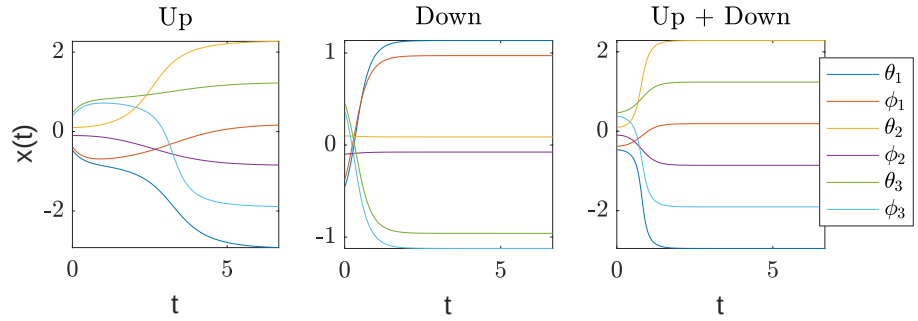
$$\mathcal{L}_1^{M,up} = \begin{matrix} \\ [1,2] \\ [1,3] \\ [2,3] \end{matrix} \begin{matrix} \begin{matrix} [1,2] & [1,3] & [2,3] \end{matrix} \\ \begin{pmatrix} 2I_2 & -\boldsymbol{\sigma}_y e^{\mathbf{i}\delta} & I_2 e^{-\mathbf{i}\delta} \\ -\boldsymbol{\sigma}_y e^{-\mathbf{i}\delta} & 2I_2 & -\boldsymbol{\sigma}_z e^{\mathbf{i}\delta} \\ I_2 e^{\mathbf{i}\delta} & -\boldsymbol{\sigma}_z e^{-\mathbf{i}\delta} & 2I_2 \end{pmatrix} \end{matrix}.$$

Its quadratic form is

$$\boldsymbol{\nu}^H \mathcal{L}_1^{M,up} \boldsymbol{\nu} = \|\nu_1 - \boldsymbol{\sigma}_y e^{\mathbf{i}\delta} \nu_2\|^2 + \|\nu_1 + I_2 e^{-\mathbf{i}\delta} \nu_3\|^2 + \|\nu_2 - \boldsymbol{\sigma}_z e^{\mathbf{i}\delta} \nu_3\|^2,$$



(a) $\delta = \pi/3$



(b) $\delta = 2\pi/3$

Figure 4.8: Diffusion on triangle in Case 2

and it becomes zero when $r_1 = r_2 = r_3 = s_1 = s_2 = s_3$ and

$$\theta_1 \equiv \phi_2 + \delta - \frac{\pi}{2},\ \phi_1 \equiv \theta_2 + \delta + \frac{\pi}{2},$$
$$\theta_1 \equiv \theta_3 - \delta + \pi,\ \phi_1 \equiv \phi_3 - \delta + \pi,$$
$$\theta_2 \equiv \phi_3 + \delta,\ \phi_2 \equiv \phi_3 + \delta + \pi.$$

The above equation system has a solution iff $\delta \in \{0, \pi/3, 2\pi/3, \pi, 4\pi/3, 5\pi/3\}$. This can be confirmed by the eigenvalue top-left plot in Figure 4.9. Furthermore, from the plot, we see that eigenvalues are given by

$$\{2 + 2\cos(\delta - \frac{\pi}{6}), 2 + 2\cos(\delta - \frac{\pi}{2}), 2 + 2\cos(\delta - \frac{5\pi}{6}), 2 + 2\cos(\delta - \frac{7\pi}{6}),$$
$$2 + 2\cos(\delta - \frac{3\pi}{2}), 2 + 2\cos(\delta - \frac{11\pi}{6})\}.$$

For the 1-down Magnetic Laplacian, we have

$$\mathcal{L}_1^{M,down} = \begin{matrix} \\ [1,2] \\ [1,3] \\ [2,3] \end{matrix} \begin{matrix} \begin{matrix} [1,2] & [1,3] & [2,3] \end{matrix} \\ \begin{pmatrix} 2I_2 & \boldsymbol{\sigma}_y & -I_2 e^{\mathbf{i}\delta} \\ \boldsymbol{\sigma}_y & 2I_2 & \boldsymbol{\sigma}_z \\ -I_2 e^{-\mathbf{i}\delta} & \boldsymbol{\sigma}_z & 2I_2 \end{pmatrix} \end{matrix},$$

and

$$\boldsymbol{\nu}^H \mathcal{L}_1^{M,down} \boldsymbol{\nu} = \|\nu_1 + \boldsymbol{\sigma}_y \nu_2\|^2 + \|\nu_1 - I_2 e^{\mathbf{i}\delta} \nu_3\|^2 + \|\nu_2 + \boldsymbol{\sigma}_z \nu_3\|^2.$$

The quadratic form is zero when $r_1 = r_2 = r_3 = s_1 = s_2 = s_3$ and

$$\begin{aligned}
&\theta_1 \equiv \phi_2 + \frac{\pi}{2},\ \phi_1 \equiv \theta_2 - \frac{\pi}{2}, \\
&\theta_1 \equiv \theta_3 + \delta,\ \phi_1 \equiv \phi_3 + \delta, \\
&\theta_2 \equiv \theta_3 + \pi,\ \phi_2 \equiv \phi_3.
\end{aligned}$$

The equation above can be solved iff $\delta \in \{\pi/2, 3\pi/2\}$ as shown in the top-right plot in Figure 4.9. Based on the plot, we can deduce that the eigenvalues are

$$\begin{aligned}
&\{2 + 2\cos(\frac{\delta}{3} - \frac{\pi}{6}), 2 - 2\cos(\frac{\delta}{3} - \frac{\pi}{6}), 2 + 2\cos(\frac{\delta}{3} - \frac{\pi}{2}), 2 - 2\cos(\frac{\delta}{3} - \frac{\pi}{2}), \\
&2 + 2\cos(\frac{\delta}{3} - \frac{5\pi}{6}), 2 - 2\cos(\frac{\delta}{3} - \frac{5\pi}{6})\}.
\end{aligned}$$

$\mathcal{L}_1^{M,up}$ and $\mathcal{L}_1^{M,up}$ do not commute except for the special case when $\delta = 0$ as shown in the bottom-right panel in Figure 4.9. Two examples of vector diffusion are displayed in Figure 4.10 for $\delta = \pi/2$ and $3\pi/2$. In the plots for $\mathcal{L}_1^{M,up}$ and $\mathcal{L}_1^{M,up}$, the vectors approach some eigenvectors that correspond to eigenvalue zero.

4.4.4 Case 4

Finally, we examine the directed 2-simplicial complex with triangle and link direction given by $\Delta_{123} = -1$ and $1 \to 2$, $2 \to 3$ and $1 \to 3$ as shown in the last column in Figure 4.11. As the edge directions are precisely the same as in Case 3, $\mathcal{L}_1^{M,down}$ in this scenario is identical to the previous case. Therefore, we only need to discuss $\mathcal{L}_1^{M,up}$, which can be computed as follows:

$$\mathcal{L}_1^{M,up} = \begin{matrix} \\ [1,2] \\ [1,3] \\ [2,3] \end{matrix} \begin{matrix} \begin{matrix} [1,2] & [1,3] & [2,3] \end{matrix} \\ \begin{pmatrix} 2I_2 & -\boldsymbol{\sigma}_y e^{-\mathbf{i}\delta} & \boldsymbol{\sigma}_x e^{\mathbf{i}\delta} \\ -\boldsymbol{\sigma}_y e^{\mathbf{i}\delta} & 2I_2 & -\boldsymbol{\sigma}_z e^{-\mathbf{i}\delta} \\ \boldsymbol{\sigma}_x e^{-\mathbf{i}\delta} & -\boldsymbol{\sigma}_z e^{\mathbf{i}\delta} & 2I_2 \end{pmatrix} \end{matrix}.$$

This leads to

$$\boldsymbol{\nu}^H \mathcal{L}_1^{M,up} \boldsymbol{\nu} = \|\nu_1 - \boldsymbol{\sigma}_y e^{-\mathbf{i}\delta} \nu_2\|^2 + \|\nu_1 + \boldsymbol{\sigma}_x e^{\mathbf{i}\delta} \nu_3\|^2 + \|\nu_2 - \boldsymbol{\sigma}_z e^{-\mathbf{i}\delta} \nu_3\|^2.$$

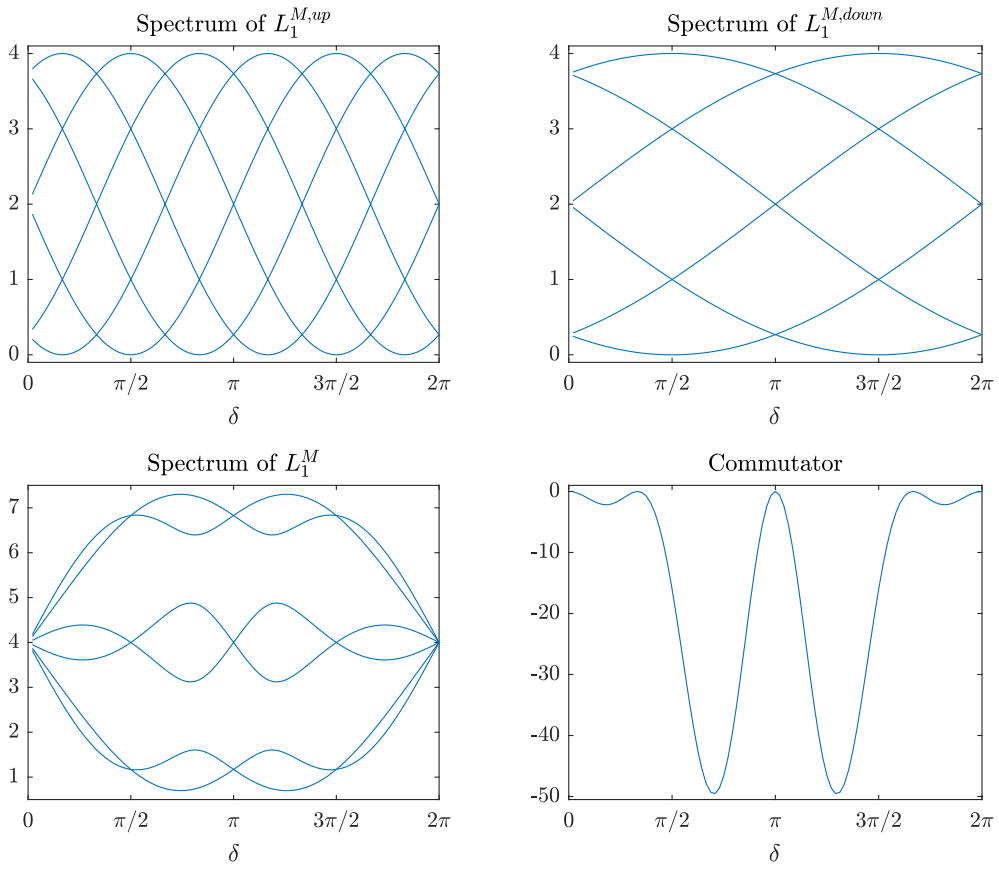
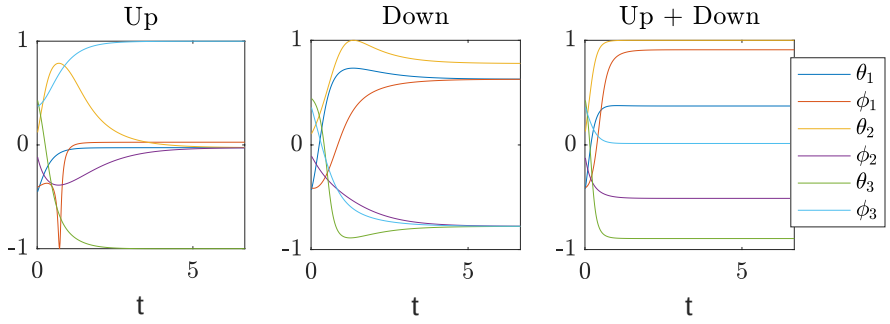
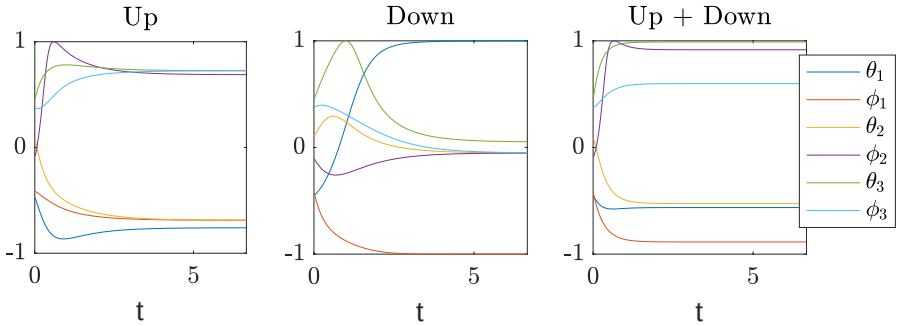


Figure 4.9: Eigenvalues of $\mathcal{L}_1^{M,up}$ (top-left), $\mathcal{L}_1^{M,down}$ (top-right), and $\mathcal{L}_1^{M,up}$ (bottom-left), and commutator $[\mathcal{L}_1^{M,up}, \mathcal{L}_1^{M,down}]$(bottom-right) in Case 3



(a) $\delta = \pi/2$



(b) $\delta = 3\pi/2$

Figure 4.10: Diffusion on triangle in Case 3

We can further prove that $\boldsymbol{\nu}^H \mathcal{L}_1^{M,up} \boldsymbol{\nu} = 0$ when $r_1 = s_2 = s_3$, $s_1 = r_2 = s_3$, and

$$\theta_1 \equiv \phi_2 - \delta - \frac{\pi}{2},\ \phi_1 \equiv \theta_2 - \delta + \frac{\pi}{2},$$
$$\theta_1 \equiv \phi_3 + \delta + \pi,\ \phi_1 \equiv \theta_3 + \delta + \pi,$$
$$\theta_2 \equiv \phi_3 - \delta,\ \phi_2 \equiv \phi_3 - \delta + \pi.$$

The solution to the equations exists iff $\delta \in \{\pi/2, 7\pi/6, 11\pi/6\}$ as shown in the top-left plot in Figure 4.11. The eigenvalues shown in the top-left subplot are

$$\{2 + 2\cos(\delta + \frac{\pi}{2}), 2 + 2\cos(\delta - \frac{5\pi}{6}), 2 + 2\cos(\delta - \frac{\pi}{6})\}.$$

Furthermore, we observe that $\mathcal{L}_1^{M,up}$ and $\mathcal{L}_1^{M,down}$ only commute when $\delta = 0$ or π. In Figure 4.12, we illustrate two vector diffusion processes where $\delta = \pi$ and $3\pi/2$.

4.5 Case Study on Triangulated Torus

In the previous examples, we only considered simplicial complexes with a single triangle. Now, we will examine examples where the simplicial complexes contain

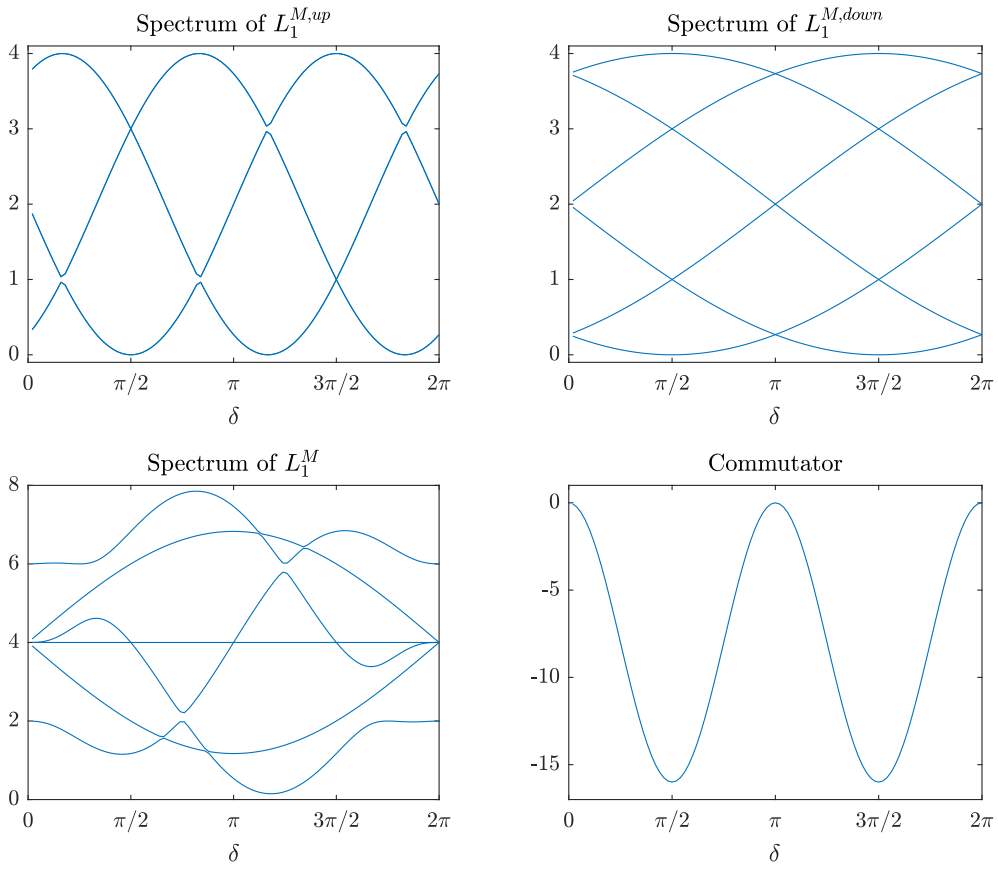
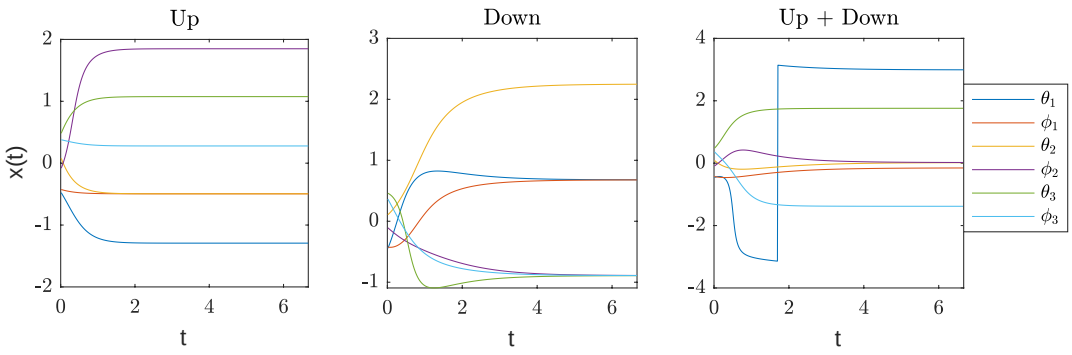
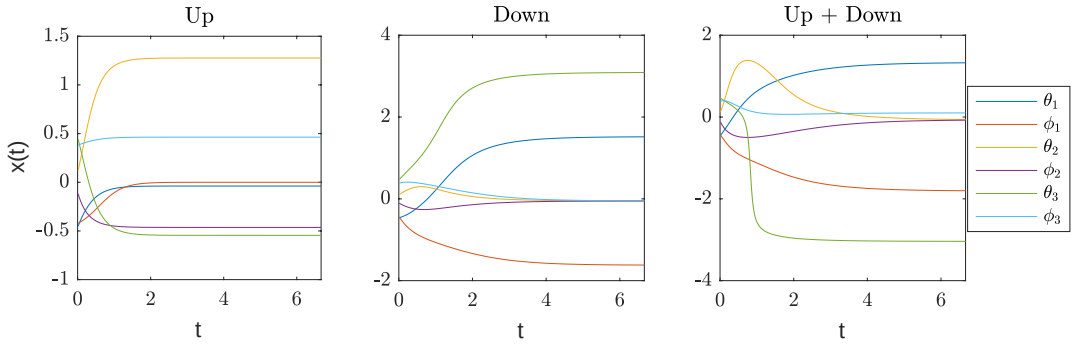


Figure 4.11: Eigenvalues of $\mathcal{L}_1^{M,up}$ (top-left), $\mathcal{L}_1^{M,down}$ (top-right), and $\mathcal{L}_1^{M,up}$ (bottom-left), and commutator $[\mathcal{L}_1^{M,up}, \mathcal{L}_1^{M,down}]$(bottom-right) in Case 4



(a) $\delta = \pi/2$



(b) $\delta = 3\pi/2$

Figure 4.12: Diffusion on triangle in Case 4

multiple triangles and edges. Specifically, we will focus on the triangulated torus as our example. The torus, as depicted in Figure 4.2a, is a 2-manifold without boundary. To perform computations, we triangulate the torus into a simplicial complex and consider two cases that differ only in the direction of the triangles. For each case, we will investigate the spectrum and eigenvectors of the 1-up and 1-down Laplacian.

4.5.1 Type 1 Torus

The first type of torus we are examining is illustrated in Figure 4.13. In this example, all edge directions align with the triangle directions. Therefore the small triangles fall under Case 1 of directed triangles depicted in Figure 4.3. We visualize the eigenvalues of $\mathcal{L}_1^{M,up}$, $\mathcal{L}_1^{M,down}$, and their sum for $\delta = \pi/3$ in Figure 4.14. In Figures 4.19 and 4.20, the edges are colored based on the phase angle of the eigenvector corresponding to the smallest eigenvalue for $\delta = \pi/3$ and $\delta = 2\pi/3$.

As we have previously discussed in Section 4.4.1, when $\delta = \pi/3$, $\mathcal{L}_1^{M,up}$ possesses a zero eigenvalue. In the top two subplots of Figure 4.19, associated with $\mathcal{L}_1^{M,up}$, the edges are differentiated by three distinct colors. If we traverse a single step following the path of the edges and triangles, we notice that the phase angle always increments by $2\pi/3$. The same pattern can be observed for $\mathcal{L}_1^{M,up}$ when $\delta = 2\pi/3$, as illustrated in Figure 4.20. This can be explained by the presence of the zero eigenvalues and its associated eigenvector, as we have analyzed in Section 4.4.1.

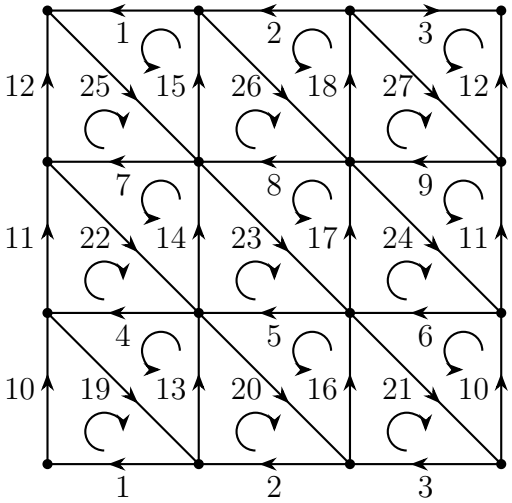


Figure 4.13: Triangulated Torus type 1

4.5.2 Type 2 Torus

In the second torus example, we alter the flow direction for all the upper triangles above the diagonal edges, compared with the previous scenario. As a result, these upper triangles align with Case 2 of directed triangles while the lower triangles

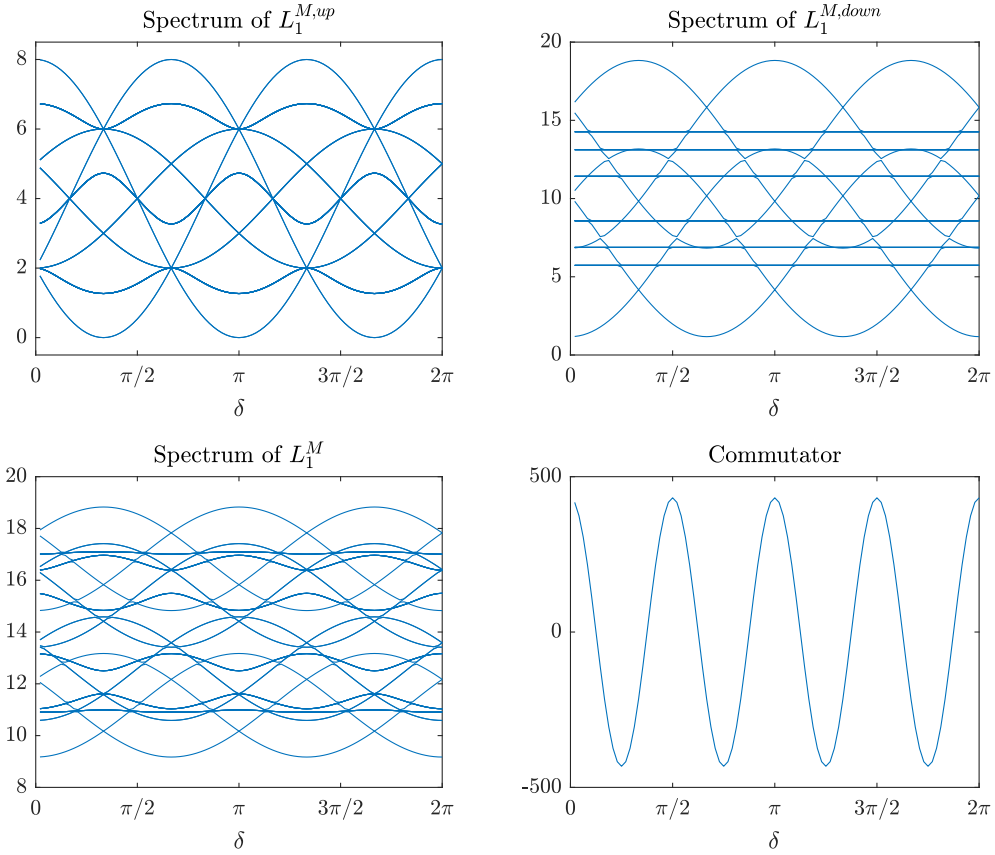


Figure 4.14: Spectrum of Type 1 torus

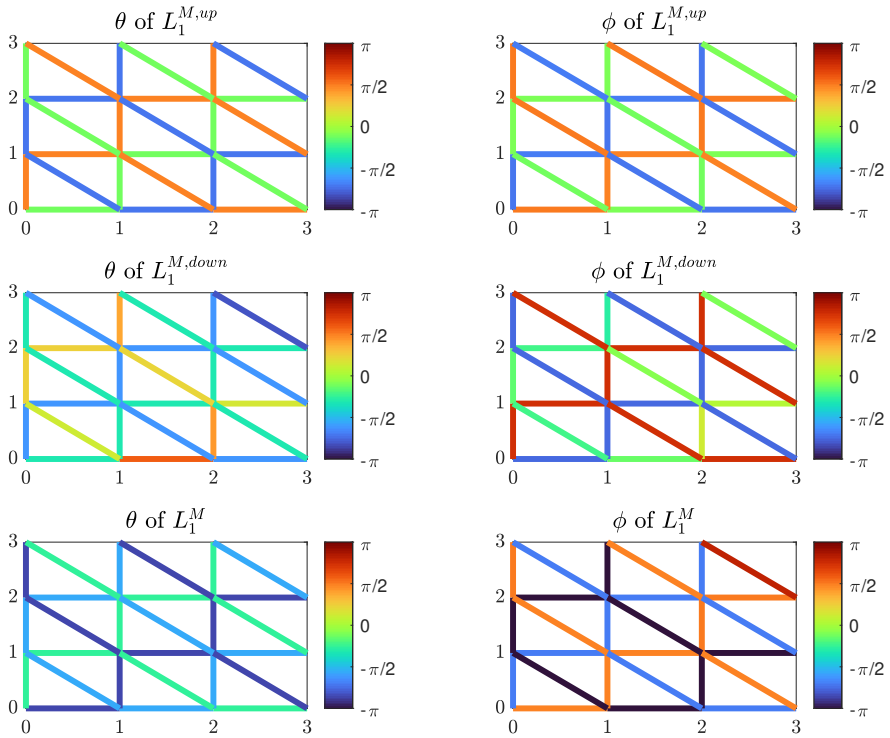


Figure 4.15: Eigenvectors of Type 1 torus, $\delta = \pi/3$

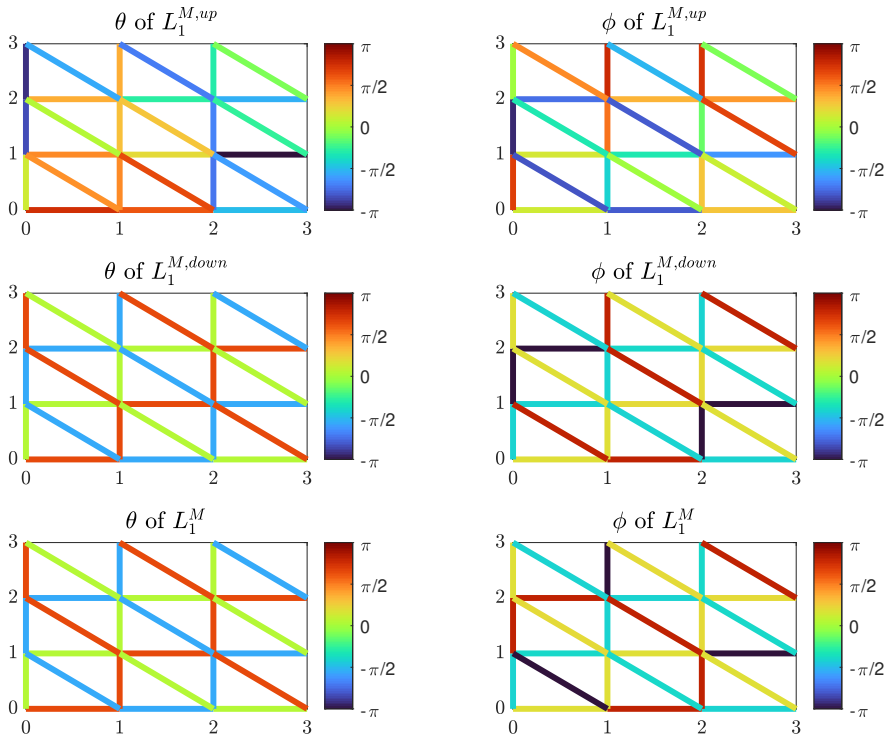


Figure 4.16: Eigenvectors of type 1 Torus, $\delta = 2\pi/3$

correspond to Case 1 illustrated in Figure 4.3. We observe that at $\delta = \pi/3$, a condition where $\mathcal{L}_1^{M,up}$ has a zero eigenvalue for both Case 1 and Case 2, the phase angles of the corresponding eigenvector are allocated such that they increase by $2\pi/3$ with each step taken along the direction of the triangles. Next, let us look at the eigenvectors when δ is set to $2\pi/3$, in which scenario the smallest eigenvalue of $\mathcal{L}_1^{M,down}$ is zero for Case 1 and 2. In this situation, the phase angles for $\mathcal{L}_1^{M,down}$ increase by $\pi/6$ with each step along the edge direction.

4.6 Conclusion

In this chapter, we have investigated the extension of Hodge Laplacians to directed simplicial complexes. The primary contribution lies in the introduction and mathematical analysis of the first-order Magnetic Hodge Laplacian. This tool offers a mathematical framework that accounts for directional flows within higher-dimensional simplexes, extending the Hodge Laplacian to directed simplicial complexes. Through our case studies on directed triangles and tori, we demonstrated that the Magnetic Laplacian can account for the direction of flow in higher-dimensional simplexes. We also showed that, on different types of tori, the eigenvectors of the 1-up and 1-down Magnetic Hodge Laplacian can help reveal interesting direction-related patterns.

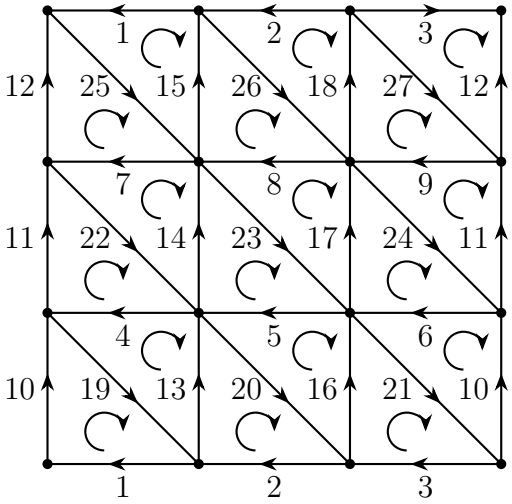


Figure 4.17: Triangulated Torus type 2

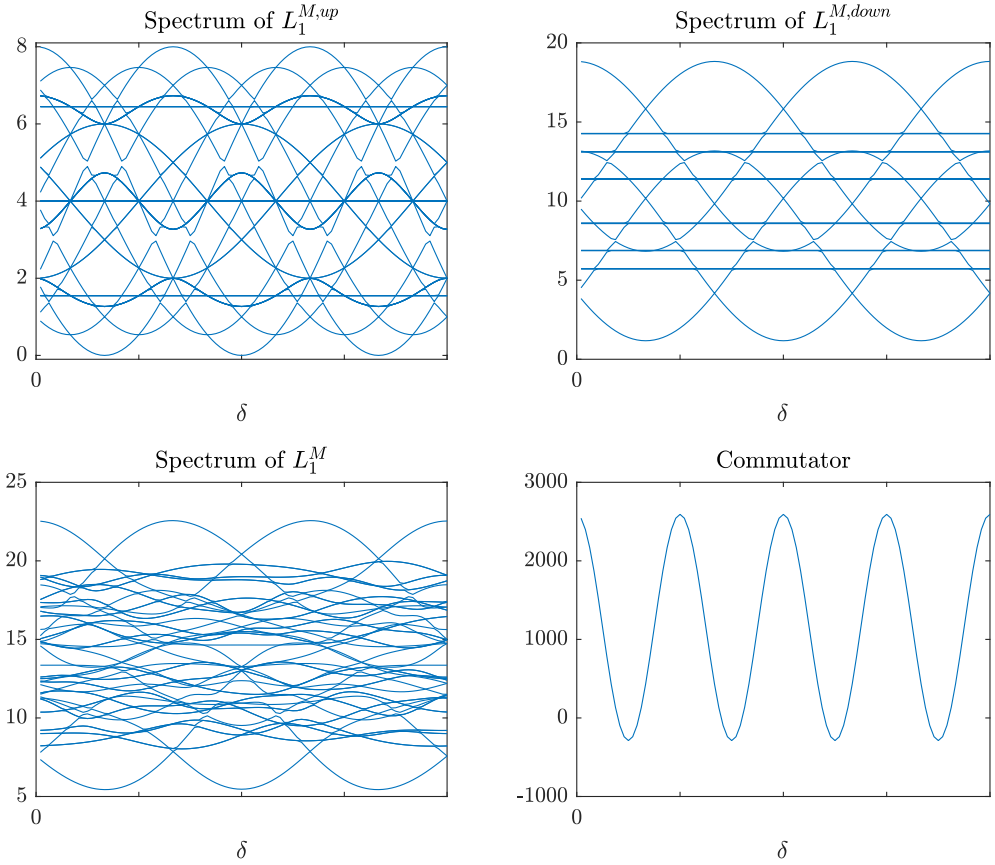


Figure 4.18: Spectrum of Type 2 torus

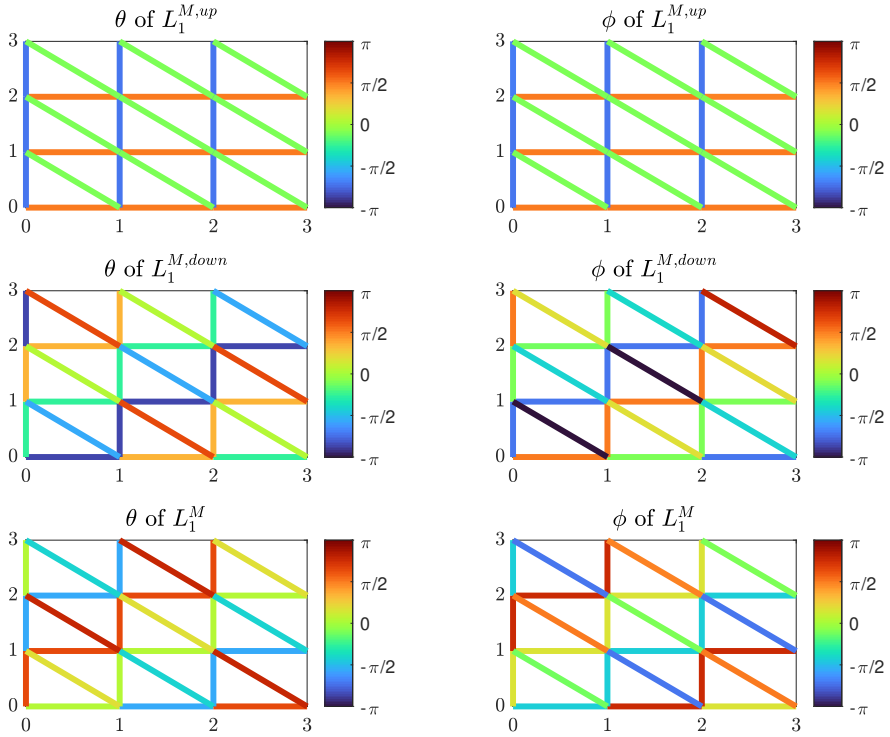


Figure 4.19: Eigenvectors of Type 2 torus, $\delta = \pi/3$

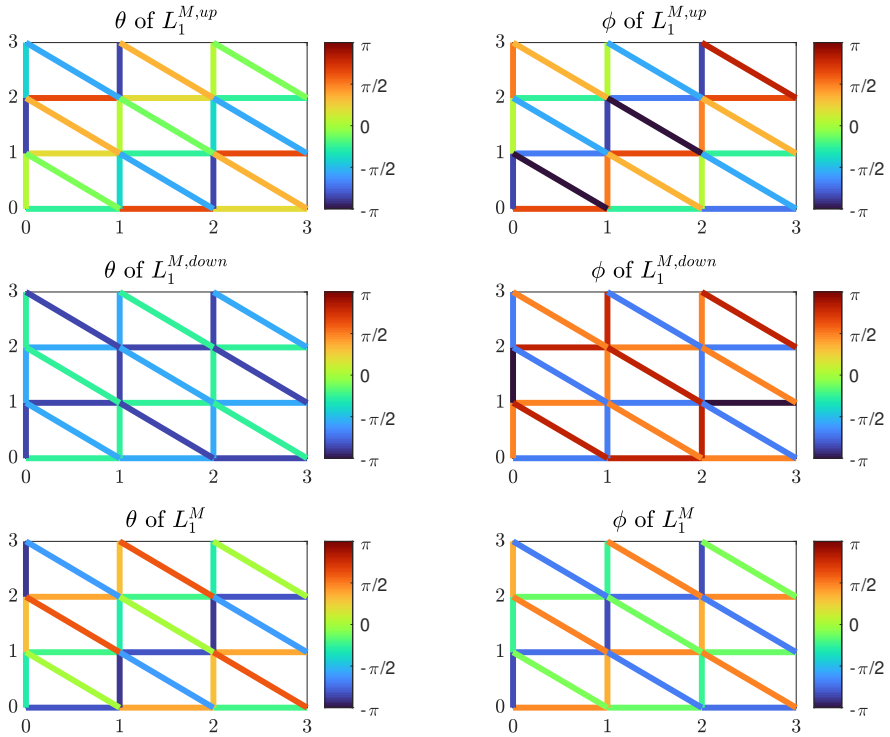


Figure 4.20: Eigenvectors of type 2 Torus, $\delta = 2\pi/3$

Chapter 5

Conclusion and Future Work

Laplacian matrices provide invaluable insights into the connectivity and clustering patterns of underlying graphs. While the standard undirected Laplacian has been extensively examined and widely applied in various learning tasks on graphs, there remains a research gap when it comes to more complex graphs. In this thesis, we explored mathematical frameworks for analyzing directed networks, hypergraphs, and directed simplicial complexes using graph Laplacians.

In Chapter 2, we examined two existing frameworks for directed graphs, namely the Magnetic Laplacian and the Trophic Laplacian, both of which factor in the direction of edges. The Magnetic Laplacian is devised to identify periodic clusters when the edges form directed cycles. On the other hand, the Trophic Laplacian focuses on recovering hierarchical patterns, which is particularly applicable when the graph is acyclic. We analyzed these two Laplacians by formulating their corresponding random graph models, which generate directed edges grounded on node embeddings. These models also provide an inference framework for comparing the prominence of periodic and linear structures. We demonstrated the application of this framework on diverse datasets from social science, biology, and ecology.

Chapter 3 focuses on hypergraph embedding, introducing customized embedding algorithms and new random models for hypergraphs. We define a hypergraph Laplacian that can adjust the significance of edges of different sizes. The corresponding spectral algorithms for generating periodic and linear embeddings are then analyzed. Furthermore, we establish related random hypergraph models and propose an inference workflow to quantify the relative strength of two types of structures using maximum likelihood. This proposed approach proves helpful in predicting new hyperedges on real hypergraphs, especially when dealing with a small training size.

There are several promising directions for future work related to Chapters 2 and 3. It would be of interest to use the likelihood ratios to compare this network feature across a well-defined category to address questions such as "are results between top chess players more or less periodic than results between top tennis players?" and "does an organism that is more advanced in an evolutionary sense have more periodic connectivity in the brain?" An extension of the comparison tool to weighted networks should also be possible; here, there are notable, and perhaps application-specific, issues about how to generalize and interpret the

Magnetic Laplacian. Also, the comparison can be extended to include other types of structure, including stochastic block and core-periphery versions [104]. This introduces further challenges of (a) accounting for different numbers of model parameters, and (b) dealing with nonlinear spectral methods. Further, by introducing an appropriate null model, it may be possible to quantify the presence of linear or periodic hierarchies in absolute rather than relative terms.

Chapter 4 extends Hodge Laplacians to directed simplicial complexes. We introduce a new Laplacian, the first-order Magnetic Hodge Laplacian, which accounts for directional flows within 2-dimensional simplicial complexes. Case studies on directed triangles and tori demonstrate its ability to capture flow direction, while spectral analysis reveals direction-related patterns. These findings may open up opportunities for potential applications in various fields that deal with directed higher-order simplexes from areas such as sensor networks, computational biology, and neuroscience. We note that the current work is exploratory in nature and carries some limitations: for instance, the Hodge decomposition does not hold for the Magnetic Hodge Laplacian. Additionally, we have not examined certain directional patterns on a torus, such as the triangles in Case 3 and Case 4. There is also further work to be done in analyzing other types of triangulated 2-manifolds, such as tetrahedra. The theoretical groundwork we laid in this chapter can be further refined and tested in various synthetic examples and real-world applications to refine the methodology and develop efficient algorithms.

Bibliography

[1] Lada A. Adamic and Natalie Glance. The political blogosphere and the 2004 US election: divided they blog. In *Proceedings of the 3rd International Workshop on Link Discovery*, pages 36–43, 2005.

[2] Austin R. Benson, Rediet Abebe, Michael T. Schaub, Ali Jadbabaie, and Jon Kleinberg. Simplicial closure and higher-order link prediction. *Proceedings of the National Academy of Sciences*, 115(48):E11221–E11230, 2018.

[3] Austin R. Benson, David F. Gleich, and Desmond J. Higham. Higher-order network analysis takes off, fueled by classical ideas and new data. *SIAM News*, 2021.

[4] Austin R. Benson, David F. Gleich, and Jure Leskovec. Higher-order organization of complex networks. *Science*, 353(6295):163–166, 2016.

[5] Austin R. Benson, David F. Gleich, and Jure Leskovec. Higher-order organization of complex networks. *Science*, pages 163–166, 2016.

[6] Seth G. Benzell, Avinash Collis, and Christos Nicolaides. Rationing social contact during the COVID-19 pandemic: Transmission risk and social benefits of US locations. *Proceedings of the National Academy of Sciences*, 117(26):14642–14644, 2020.

[7] Ginestra Bianconi. *Higher order networks : An introduction to simplicial complexes*. Cambridge University Press, 2021.

[8] Iwan Boksebeld and Amir Vaxman. High-order directional fields. *ACM Transactions on Graphics*, 41(6):1–17, 2022.

[9] Iwan Boksebeld and Amir Vaxman. High-order directional fields. *ACM Transactions on Graphics*, 41(6), nov 2022.

[10] Glen E Bredon. *Topology and geometry*, volume 139. Springer Science and Business Media, 2013.

[11] Zixuan Cang, Lin Mu, Kedi Wu, Kristopher Opron, Kelin Xia, and Guo-Wei Wei. A topological approach for protein classification. *Computational and Mathematical Biophysics*, 3(1), 2015.

[12] Gunnar Carlsson. Topology and data. *Bulletin of the American Mathematical Society*, 46(2):255–308, 2009.

[13] Joseph Minhow Chan, Gunnar Carlsson, and Raul Rabadan. Topology of viral evolution. *Proceedings of the National Academy of Sciences*, 110(46):18566–18571, 2013.

[14] Frédéric Chazal and Bertrand Michel. An introduction to topological data analysis: fundamental and practical aspects for data scientists. *Frontiers in Artificial Intelligence*, 4:667963, 2021.

[15] Madhukar Chhimwal, Saurabh Agrawal, and Girish Kumar. Measuring circular supply chain risk: A Bayesian network methodology. *Sustainability*, 13:8448, 2021.

[16] Philip S. Chodrow, Nate Veldt, and Austin R. Benson. Generative hypergraph clustering: From blockmodels to modularity. *Science Advances*, 7(28):eabh1303, 2021.

[17] Fan Chung. *Spectral Graph Theory*. Regional conference series in mathematics; no. 92. American Mathematical Society, Providence, R.I., 1997.

[18] Fan Chung. Laplacians and the cheeger inequality for directed graphs. *Annals of Combinatorics*, 9(1):1–19, 2005.

[19] Fan Chung and Mark Kempton. A local clustering algorithm for connection graphs. *Lecture Notes in Computer Science (including subseries Lecture Notes in Artificial Intelligence and Lecture Notes in Bioinformatics)*, 8305:26–43, 2013.

[20] Fan Chung and Wenbo Zhao. Ranking and sparsifying a connection graph. In *Algorithms and Models for the Web Graph*, volume 7323 of *Lecture Notes in Computer Science*, pages 66–77. Springer Berlin Heidelberg, Berlin, Heidelberg, 2012.

[21] Giulio Cimini, Rossana Mastrandrea, and Tiziano Squartini. *Reconstructing networks*. Cambridge University Press, 2021.

[22] Anthony Cole. The influence matrix methodology: A technical report. *Landcare Research Contract Report: LC0506/175*, 2006.

[23] Mihai Cucuringu, Huan Li, He Sun, and Luca Zanetti. Hermitian matrices for clustering directed graphs: insights and applications. In *International Conference on Artificial Intelligence and Statistics*, pages 983–992. PMLR, 2020.

[24] Mihai Cucuringu and Hemant Tyagi. An extension of the angular synchronization problem to the heterogeneous setting. *Foundations of Data Science*, 4(1):71–122, 2020.

[25] Jesse Davis and Mark Goadrich. The relationship between precision-recall and roc curves. In *Proceedings of the 23rd International Conference on Machine Learning*, pages 233–240, 2006.

[26] Henry-Louis De Kergorlay and Desmond J. Higham. Consistency of anchor-based spectral clustering. *Information and Inference: A Journal of the IMA*, 11:801—822, 2022.

[27] Vin De Silva and Robert Ghrist. Coverage in sensor networks via persistent homology. *Algebraic and Geometric Topology*, 7(1):339–358, 2007.

[28] Mathieu Desbrun, Eva Kanso, and Yiying Tong. Discrete differential forms for computational modeling. In *ACM SIGGRAPH 2006 Courses*, pages 39–54. 2006.

[29] Markus Deserno. How to generate equidistributed points on the surface of a sphere, 2004. Accessed on Month Day, Year.

[30] Michaël Fanuel, Carlos M Alaiz, Angela Fernandez, and Johan A.K Suykens. Magnetic eigenmaps for the visualization of directed networks. *Applied and Computational Harmonic Analysis*, 44(1):189–199, 2018.

[31] Michaël Fanuel, Carlos M Alaíz, and Johan A. K. Suykens. Magnetic eigenmaps for community detection in directed networks. *Physical Review E*, 95(2):022302–022302, 2017.

[32] James H. Fowler. Connecting the congress: A study of cosponsorship networks. *Political Analysis*, 14(04):456–487, 2006.

[33] James H. Fowler. Legislative cosponsorship networks in the US house and senate. *Social Networks*, 28(4):454–465, oct 2006.

[34] Brendan J. Frey and Delbert Dueck. Clustering by passing messages between data points. *Science*, 315(5814):972–976, 2007.

[35] Joel Friedman. Computing betti numbers via combinatorial laplacians. In *Proceedings of the twenty-eighth annual ACM symposium on Theory of Computing*, pages 386–391, 1996.

[36] Francesco Galuppi, Raffaella Mulas, and Lorenzo Venturello. Spectral theory of weighted hypergraphs via tensors. *Linear and Multilinear Algebra*, 0(0):1–31, 2022.

[37] Robert Ghrist and Abubakr Muhammad. Coverage and hole-detection in sensor networks via homology. In *IPSN 2005. Fourth International Symposium on Information Processing in Sensor Networks, 2005.*, pages 254–260. IEEE, 2005.

[38] Peter Giblin. *Graphs, surfaces and homology: an introduction to algebraic topology*. Springer Science & Business Media, 2013.

[39] Chad Giusti, Robert Ghrist, and Danielle S Bassett. Two's company, three (or more) is a simplex: Algebraic-topological tools for understanding higher-order structure in neural data. *Journal of Computational Neuroscience*, 41:1–14, 2016.

[40] David F. Gleich. PageRank beyond the web. *SIAM Review*, 57(3):321–363, 2015.

[41] Timothy E. Goldberg. Combinatorial laplacians of simplicial complexes. *Senior Thesis, Bard College*, 6, 2002.

[42] Xue Gong, Desmond J. Higham, and Konstantinos Zygalakis. Directed network laplacians and random graph models. *Royal Society Open Science*, 8(10):211144, 2021.

[43] Xue Gong, Desmond J. Higham, and Konstantinos Zygalakis. Generative hypergraph models and spectral embedding. *Scientific Reports*, 13(1):540, 2023.

[44] Peter Grindrod. Range-dependent random graphs and their application to modeling large small-world proteome datasets. *Physical Review E*, 66(6):066702/7–066702, 2002.

[45] Peter Grindrod, Desmond J. Higham, and Gabriela Kalna. Periodic reordering. *IMA Journal of Numerical Analysis*, 30(1):195–207, 2010.

[46] William L. Hamilton. Graph representation learning. *Synthesis Lectures on Artificial Intelligence and Machine Learning*, 14(3):1–159, 2020.

[47] William L Hamilton, Rex Ying, and Jure Leskovec. Representation learning on graphs: Methods and applications. *IEEE Data Engineering Bulletin*, 2017.

[48] Desmond J. Higham. Unravelling small world networks. *Journal of Computational and Applied Mathematics*, 158(1):61–74, 2003.

[49] Desmond J. Higham. Spectral reordering of a range-dependent weighted random graph. *IMA Journal of Numerical Analysis*, 25(3):443–457, 2005.

[50] Desmond J. Higham. Spectral clustering and its use in bioinformatics. *Journal of Computational and Applied Mathematics*, 204:25–37, 2007.

[51] Desmond J. Higham and Henry-Louis De Kergorlay. Epidemics on hypergraphs: Spectral thresholds for extinction. *Proceedings of the Royal Society A*, 477(2252):20210232, 2021.

[52] Desmond J. Higham, Gabriela Kalna, and Milla J. Kibble. Spectral clustering and its use in bioinformatics. *Journal of Computational and Applied Mathematics*, 204:25–37, 2007.

[53] Anil Nirmal Hirani. *Discrete exterior calculus*. California Institute of Technology, 2003.

[54] Peter D. Hoff, Adrian E Raftery, and Mark S Handcock. Latent space approaches to social network analysis. *Journal of the American Statistical Association*, 97(460):1090–1098, 2002.

[55] Roger A. Horn and Charles R. Johnson. *Matrix Analysis*. Cambridge University Press, Cambridge, 1985.

[56] Lawrence Hubert and Phipps Arabie. Comparing partitions. *Journal of Classification*, 2(1):193–218, 1985.

[57] Iacopo Iacopini, Giovanni Petri, Alain Barrat, and Vito Latora. Simplicial models of social contagion. *Nature Communications*, 10(1):2485, 2019.

[58] Lorien Jasny and Dana R Fisher. Echo chambers in climate science. *Environmental Research Communications*, 1(10):101003, 2019.

[59] Xiaoye Jiang, Lek-Heng Lim, Yuan Yao, and Yinyu Ye. Statistical ranking and combinatorial hodge theory. *Mathematical Programming*, 127(1):203–244, 2011.

[60] Samuel Johnson. Digraphs are different: Why directionality matters in complex systems. *Journal of Physics: Complexity*, 1:015003, 2020.

[61] Marcus Kaiser and Claus C. Hilgetag. Nonoptimal component placement, but short processing paths, due to long-distance projections in neural systems. *PLOS Computational Biology*, 2(7):e95, 2006.

[62] Jon Kleinberg. Navigation in a small world. *Nature*, 406:845, 2000.

[63] Renaud Lambiotte, Martin Rosvall, and Ingo Scholtes. From networks to optimal higher-order models of complex systems. *Nature Physics*, 15(4):313–320, 2019.

[64] Renaud Lambiotte and Michael T. Schaub. *Modularity and Dynamics on Complex Networks*. Elements in the Structure and Dynamics of Complex Networks. Cambridge University Press, 2022.

[65] Jure Leskovec and Andrej Krevl. SNAP Datasets: Stanford large network dataset collection. http://snap.stanford.edu/data, June 2014.

[66] Stephen Levine. Several measures of trophic structure applicable to complex food webs. *Journal of Theoretical Biology*, 83:195–207, 1980.

[67] Lek-Heng Lim. Hodge laplacians on graphs. *SIAM Review*, 62(3):685–715, 2020.

[68] Maxime Lucas, Giulia Cencetti, and Federico Battiston. Multiorder laplacian for synchronization in higher-order networks. *Physical Review Research*, 2:033410, Sep 2020.

[69] Helmut Lütkepohl. *Handbook of Matrices*. Wiley, Chichester, 1996.

[70] Ulrike Luxburg. A tutorial on spectral clustering. *Statistics and Computing*, 17(4):395–416, 2007.

[71] Peter Macgregor and He Sun. A tighter analysis of spectral clustering, and beyond. In *International Conference on Machine Learning*, pages 14717–14742. PMLR, 2022.

[72] Robert S. MacKay. Incomplete pairwise comparison. *Mathematics Today*, 132, 2020.

[73] Robert S. Mackay, Samuel Johnson, and Benedict Sansom. How directed is a directed network? *Royal Society Open Science*, 7(9):201138–201138, 2020.

[74] Fragkiskos D. Malliaros and Michalis Vazirgiannis. Clustering and community detection in directed networks: A survey. *Physics Reports*, 533(4):95–142, 2013.

[75] Rossana Mastrandrea, Julie Fournet, and Alain Barrat. Contact patterns in a high school: a comparison between data collected using wearable sensors, contact diaries and friendship surveys. *PLOS One*, 10(9):e0136497, 2015.

[76] Chris McComb. Adjusted rand index. *GitHub*, 2022. Retrieved June 29, 2022.

[77] Ron Milo, Shai Shen-Orr, Shalev Itzkovitz, Nadav Kashtan, Dmitri Chklovskii, and Uri Alon. Network motifs: Simple building blocks of complex networks. *Science*, 298(5594):824–827, 2002.

[78] Shigeyuki Morita. *Geometry of differential forms*. Number 201. American Mathematical Soc., 2001.

[79] Giannis Moutsinas, Choudhry Shuaib, Weisi Guo, and Stephen Jarvis. Graph hierarchy: A novel approach to understanding hierarchical structures in complex networks. *Scientific Reports*, 11:13943, 2019.

[80] James R. Munkres. *Elements of algebraic topology*. CRC press, 2018.

[81] Vidit Nanda and Radmila Sazdanović. Simplicial models and topological inference in biological systems. In *Discrete and topological models in molecular biology*, pages 109–141. Springer, 2013.

[82] Mark E.J Newman. Finding community structure in networks using the eigenvectors of matrices. *Physical Review E*, 74(3):036104–036104, 2006.

[83] Nina Otter, Mason A Porter, Ulrike Tillmann, Peter Grindrod, and Heather A Harrington. A roadmap for the computation of persistent homology. *EPJ Data Science*, 6:1–38, 2017.

[84] William R Palmer and Tian Zheng. Spectral clustering for directed networks. In *International Conference on Complex Networks and Their Applications*, pages 87–99. Springer, 2020.

[85] Tiago P. Peixoto. Ordered community detection in directed networks. *Physical Review E*, 106:024305, Aug 2022.

[86] Richard Peng, He Sun, and Luca Zanetti. Partitioning well-clustered graphs: Spectral clustering works! *SIAM Journal on Computing*, 46(2):710–743, 2017.

[87] Giovanni Petri, Paul Expert, Federico Turkheimer, Robin Carhart-Harris, David Nutt, Peter J Hellyer, and Francesco Vaccarino. Homological scaffolds of brain functional networks. *Journal of The Royal Society Interface*, 11(101):20140873, 2014.

[88] Emad Ramadan, Arijit Tarafdar, and Alex Pothen. A hypergraph model for the yeast protein complex network. In *18th International Parallel and Distributed Processing Symposium, 2004. Proceedings.*, pages 189–, 2004.

[89] William M. Rand. Objective criteria for the evaluation of clustering methods. *Journal of the American Statistical Association*, 66(336):846 – 850, 1971.

[90] Ryan A. Rossi, Nesreen K. Ahmed, Eunyee Koh, Sungchul Kim, Anup Rao, and Yasin Abbasi-Yadkori. A structural graph representation learning framework. In *Proceedings of the 13th International Conference on Web Search and Data Mining*, WSDM 2020, page 483–491, New York, NY, USA, 2020. Association for Computing Machinery.

[91] Benedict Sansom, Samuel Johnson, and Robert S. MacKay. Trophic incoherence drives systemic risk in financial exposure networks. Technical Report Working Paper Number 39, National Institute of Economic and Social Research, Westminster, London, 2021.

[92] Stefania Sardellitti, Sergio Barbarossa, and Lucia Testa. Topological signal processing over cell complexes. In *2021 55th Asilomar Conference on Signals, Systems, and Computers*, pages 1558–1562. IEEE, 2021.

[93] Michael T. Schaub, Austin R. Benson, Paul Horn, Gabor Lippner, and Ali Jadbabaie. Random walks on simplicial complexes and the normalized hodge 1-laplacian. *SIAM Review*, 62(2):353–391, 2020.

[94] Michael T. Schaub and Santiago Segarra. Flow smoothing and denoising: Graph signal processing in the edge-space. In *2018 IEEE Global Conference on Signal and Information Processing (GlobalSIP)*, pages 735–739. IEEE, 2018.

[95] Ingo Scholtes. When is a network a network? multi-order graphical model selection in pathways and temporal networks. In *Proceedings of the 23rd ACM SIGKDD international conference on knowledge discovery and data mining*, pages 1037–1046, 2017.

[96] Bernhard Schölkopf, John Platt, and Thomas Hofmann. Learning with hypergraphs: Clustering, classification, and embedding. In *Advances in Neural Information Processing Systems 19: Proceedings of the 2006 Conference*, pages 1601–1608, 2007.

[97] Nicholas Sharp, Yousuf Soliman, and Keenan Crane. The vector heat method. *ACM Transactions on Graphics*, 38(3):1–19, 2019.

[98] Shai S Shen-Orr, Ron Milo, Shmoolik Mangan, and Uri Alon. Network motifs in the transcriptional regulation network of escherichia coli. *Nature Genetics*, 31(1):64–68, 2002.

[99] Jianbo Shi and J. Malik. Normalized cuts and image segmentation. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 22(8):888–905, 2000.

[100] Amit Singer. Angular synchronization by eigenvectors and semidefinite programming. *Applied and Computational Harmonic Analysis*, 30(1):20–36, 2011.

[101] Juliette Stehlé, Nicolas Voirin, Alain Barrat, Ciro Cattuto, Lorenzo Isella, Jean-François Pinton, Marco Quaggiotto, Wouter Van den Broeck, Corinne Régis, Bruno Lina, and Philippe Vanhems. High-resolution measurements of face-to-face contact patterns in a primary school. *PLOS One*, 6(8):e23176, 2011.

[102] Gilbert Strang. *Linear Algebra and Learning from Data*. Wellesley-Cambridge Press, 2019.

[103] Leo Torres, Ann S. Blevins, Danielle S. Bassett, and Tina Eliassi-Rad. The why, how, and when of representations for complex systems. *SIAM Review*, 63:435–485, 2021.

[104] Francesco Tudisco and Desmond J. Higham. A nonlinear spectral method for core-periphery detection in networks. *SIAM Journal on Mathematics of Data Science*, 1:269–292, 2019.

[105] Francesco Tudisco and Desmond J. Higham. Core-periphery detection in hypergraphs. *SIAM Journal on Mathematics of Data Science*, 5(1):1–21, 2023.

[106] Robert E. Ulanowicz and Donald L. DeAngelis. Network analysis of trophic dynamics in South Florida ecosystems. *US Geological Survey Program on the South Florida Ecosystem*, 114:45, 2005.

[107] Ulrike Von Luxburg, Mikhail Belkin, and Olivier Bousquet. Consistency of spectral clustering. *The Annals of Statistics*, pages 555–586, 2008.

[108] Chang-Dong Wang and Jian-Huang Lai. *Nonlinear Clustering: Methods and Applications*, pages 253–302. Springer International Publishing, Cham, 2016.

[109] Duncan J. Watts and Steven H. Strogatz. Collective dynamics of 'small-world' networks. *Nature*, 393(6684):440–442, 1998.

[110] Scott White and Padhraic Smyth. A spectral clustering approach to finding communities in graphs. In *Proceedings of the 2005 SIAM international conference on data mining*, pages 274–285. SIAM, 2005.

[111] William B. Wood, editor. *The nematode Caenorhabditis Elegans*. Cold Spring Harbor monograph series; 17, 1988.

[112] Kelin Xia and Guo-Wei Wei. Persistent homology analysis of protein structure, flexibility, and folding. *International Journal for Numerical Methods in Biomedical Engineering*, 30(8):814–844, 2014.

[113] Naganand Yadati, Vikram Nitin, Madhav Nimishakavi, Prateek Yadav, Anand Louis, and Partha Talukdar. Nhp: Neural hypergraph link prediction. In *Proceedings of the 29th ACM International Conference on Information & Knowledge Management*, pages 1705–1714, 2020.

[114] Seeun Yoon, Hyungseok Song, Kijung Shin, and Yung Yi. How much and when do we need higher-order information in hypergraphs? a case study on hyperedge prediction. In *Proceedings of The Web Conference 2020*, pages 2627–2633, 2020.

[115] Jun Yu, Dacheng Tao, and Meng Wang. Adaptive hypergraph learning and its application in image classification. *IEEE Transactions on Image Processing*, 21(7):3262–3272, 2012.

[116] Rundong Zhao, Mathieu Desbrun, Guo-Wei Wei, and Yiying Tong. 3d hodge decompositions of edge- and face-based vector fields. *ACM Transactions on Graphics*, 38(6), nov 2019.