Learning on Simplicial Complexes
From Convolutions to Generative Models

Yang, Maosheng

DOI
10.4233/uuid:6d221b45-6cb0-42bf-8751-accc25a55469

Publication date
2025

Document Version
Final published version

Citation (APA)
Yang, M. (2025). *Learning on Simplicial Complexes: From Convolutions to Generative Models*. [Dissertation (TU Delft), Delft University of Technology]. https://doi.org/10.4233/uuid:6d221b45-6cb0-42bf-8751-accc25a55469

Delft University of Technology

Learning on Simplicial Complexes

From Convolutions to Generative Models

By Maosheng Yang

Learning on Simplicial Complexes

From Convolutions to Generative Models

Dissertation

for the purpose of obtaining the degree of doctor
at Delft University of Technology
by the authority of the Rector Magnificus Prof.dr.ir. T.H.J.J. van der Hagen,
chair of the Board for Doctorates
to be defended publicly on
Monday 24 November 2025 at 15:00

by

Maosheng YANG

Master of Science in Electrical Engineering,
Delft University of Technology, the Netherlands

This dissertation has been approved by the promotors.

Composition of the Doctorate Committee:

Rector Magnificus,	chairperson
Prof. dr. ir. G. J. T. Leus,	Delft University of Technology, *promotor*
Dr. E. Isufi,	Delft University of Technology, *copromotor*
Independent members:	
Prof. dr. A. Papapantoleon,	Delft University of Technology
Prof. dr. B. Grossenbacher-Rieck,	University of Fribourg, Switzerland
Prof. dr. B. Beferull-Lozano,	Simula Metropolitan Center for Digital Engineering, Norway
Dr. K. Hildebrandt,	Delft University of Technology
Dr. P. Di Lorenzo,	Sapienza University of Rome, Italy
Prof. dr. A. Hanjalic,	Delft University of Technology, reserve member

Keywords: Simplicial Complexes; Simplicial Signals; Convolution; Neural Networks; Gaussian Processes; Schrödinger Bridge; Generative Models

ISBN: 978-94-6518-154-7

An electronic version of this dissertation is available at
http://repository.tudelft.nl/.

Contents

Summary

Machine learning has been growing beyond data living on Euclidean spaces (e.g., texts, audios, images). Graph machine learning models, e.g., graph neural networks (GNNs), succeed in learning from graph-structured data using the graph topological information. In this thesis, we focus on a new domain, simplicial complexes. Not only are they a popular higher-order network model that generalizes graph models encoding pairwise relations between nodes, but they also allow us to support signals on various network objects (nodes, edges, and faces). For example, edge flows defined on a simplicial complex can be studied in terms of both divergent and rotational properties, providing a better model for real-world flows like traffic flows, water flows, money flows, etc.

The main theme of the thesis is to develop *principled* machine learning models for signals on simplicial complexes. By *principled*, we mean that the models leverage the intrinsic priors of the domain and the signals, namely, the *topological* structure of simplicial compelxes and the *Hodge decomposition* of simplicial signals. The latter states that, for example, edge flows can be decomposed into a divergence-free part and a curl-free part, each modeling the distinct properties of real-world flows — the conservation of flows (e.g., water flows) and the rotational properties of flows (e.g., electric currents).

We start by defining a *simplicial convolution* operator. In analogy to the classical convolution operator in Euclidean spaces, the simplicial convolution acts as a fundamental tool for signal processing, and a building block for learning on simplicial complexes. Spatially, it performs a *shift-and-sum* operation within the local simplicial neighborhoods. Spectrally, we show that this operator regulates the different Hodge subspaces individually, allowing for a flexible control on the different Hodge components of the signals. Inspired by convolutional neural networks (CNNs) and GNNs, we then build neural network models on simplicial complexes based on the convolution operator. These models allow us to perform reconstruction, classification and regression tasks on simplicial signals, yet in a deterministic sense.

To enable probabilistic learning of simplicial signals, we later construct Gaussian processes on simplicial complexes owing to its Gaussian nature. We achieve the separate modeling of different Hodge components by once again leveraging the Hodge decomposition. Finally, we investigate the fundamental challenge of matching *arbitrary* distributions of simplicial signals for generative learning on simplicial complexes. By generalizing the classical Schrödinger bridge problem—a dynamic entropy-regularized optimal transport—to simplicial complexes, we formulate the *topological Schrödinger bridge problem* with respect to a topology-aware reference process (e.g., topological heat diffusion). The solution to this problem allows us to build topological Schrödinger bridge models for generative learning and matching of simplicial signals.

We have evaluated these models on networks involved with edge flows, such as road/water

networks, collaboration networks, financial networks, brain networks and so on. Across various tasks, we show the effectiveness of the proposed models and the importance of leveraging both the topological structure and the Hodge decomposition of simplicial signals.

Symbols

Sets

- $\mathcal{G}$: a graph
- SC_2: a simplicial 2-complex (this is a set)
- $\mathcal{V}$: a set of nodes/vertices
- $\mathcal{E}$: a set of edges
- $\mathcal{F}$: a set of faces, or $\mathcal{T}$: a set of triangular faces
- $1, 2, \ldots, N_0$: nodes in $\mathcal{V}$ and N_0: the number of nodes
- $e_1, e_2, \ldots, e_{N_1}$: edges in $\mathcal{E}$ and N_1: the number of edges
- $t_1, t_2, \ldots, t_{N_2}$: faces in $\mathcal{F}$ and N_2: the number of faces
- $\mathcal{S} := \cup_k \mathcal{S}^k$: a set of simplices for $k = 0, \ldots, K$
- $\mathcal{S}^k$: a set of k-simplices and $N_k := |\mathcal{S}^k|$ the number of simplices
- $s_1^k, s_2^k, \ldots, s_{N_k}^k$: simplices in $\mathcal{S}^k$
- $\mathcal{N}$: a set of neighbors

Signals

- $\mathbf{x}^k$: k-simplicial signals
- $\mathbf{v} = [v_1, \ldots, v_{N_0}]^\top \in \mathbb{R}^{N_0}$: node signals
- $\mathbf{f} = [f_1, \ldots, f_{N_1}]^\top \in \mathbb{R}^{N_1}$: edge flows
- $\mathbf{t} = [t_1, \ldots, t_{N_2}]^\top \in \mathbb{R}^{N_2}$: triangle flows
- $\mathbf{X}^k \in \mathbb{R}^{N_k \times D}$: D-dimensional k-simplicial signals

OPERATORS

- $\ker(\cdot)$: the kernel space of a matrix (linear operator).
- $\mathrm{im}(\cdot)$: the image space of a matrix (linear operator).
- $\mathbf{I}$: identity matrix of appropriate size.
- $\mathbf{B}_k$: the incidence matrix between $(k-1)$- and k-simplices, a.k.a., the k-th boundary operator. The transpose $\mathbf{B}_k^\top$ is the coboundary operator.
- $\mathbf{B}_1$: the node-edge incidence matrix of size $N_0 \times N_1$.
- $\mathbf{B}_2$: the edge-triangle incidence matrix of size $N_1 \times N_2$.
- $\mathbf{L}_k$: the k-th Hodge Laplacian operator. Sometimes, we denote the 1-Hodge Laplacian as $\mathbf{L}$ for brevity. $\mathbf{L}_0$ is the graph Laplacian.
- $\mathbf{L}_{k,\mathrm{d}}$ is the down (or lower) Laplacian and $\mathbf{L}_{k,\mathrm{u}}$ is the up Laplacian.
- $\mathbf{L}_k = \mathbf{U}_k \mathbf{\Lambda}_k \mathbf{U}_k^\top$: the eigendecomposition of the k-th Hodge Laplacian. $\mathbf{U}_k = [\mathbf{u}_1, \ldots, \mathbf{u}_{N_1}]^\top$ is the eigenvector matrix and $\mathbf{\Lambda}_k = \mathrm{diag}([\lambda_1, \ldots, \lambda_{N_1}])$ is the eigenvalue diagonal matrix, where $(\lambda_i, \mathbf{u}_i)$ is the i-th eigenpair.
- $\mathbf{H}_k$: the k-th simplicial convolutional filter. $\mathbf{H}_0$: graph filters. $\mathbf{H}_1$: edge filters, sometimes, breviated as $\mathbf{H}$.
- Subscripts $\cdot_{\mathrm{H}}, \cdot_{\mathrm{G}}, \cdot_{\mathrm{C}}$ denote mathematical objects associated with the *harmonic, gradient* and *curl* subspaces, respectively, given by the *Hodge decomposition*.

1

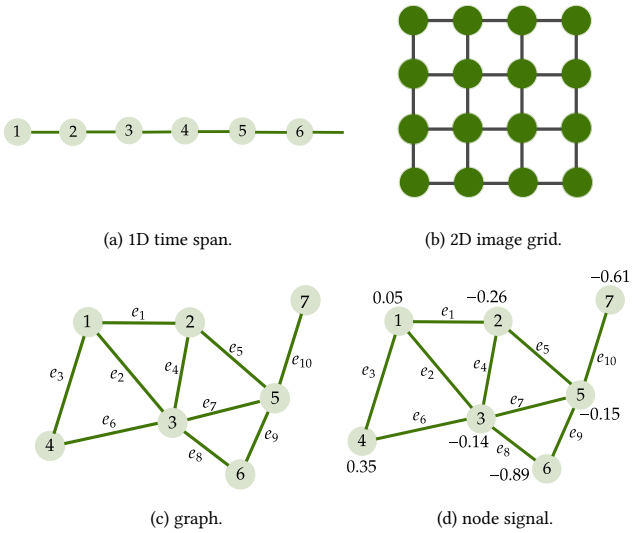
Introduction

In the past few years, we have witnessed a surge of interest in developing machine learning models for *various types of data*. By *various types*, we mean that the data can be of different nature. For example, audios, images, videos, etc., are often referred to as *regular data*, as in data defined over regular domains, because they are often mathematically treated as *signals* over a 1-dimensional (1D) time span, a 2-dimensional (2D) image grid, and so on, and they are thus known as *Euclidean data*. In contrast, *irregular data*, or *non-Euclidean data*, refers to the data defined over, or associated with, irregular domains, such as graphs, groups, manifolds and other non-Euclidean domains. The success of machine learning models to extract patterns from these data is then to exploit their local relationships and the invariants of the underlying domain.

Among them, *graphs* are simple and general *topological* structures, composed of a set of nodes and edges connecting the nodes that can be used to capture the local relationships between data points. Note that 1D discretized timepoints and 2D image grids can be viewed as special graphs, as shown in Fig. 1.1. The graph in Fig. 1.1c has seven nodes ($\{1, 2, \ldots, 7\}$) and ten edges ($\{e_1, \ldots, e_{10}\}$), where, e.g., e_1 connects nodes 1 and 2. Graphs have been used to model many real-world applications associated to *networked systems* where nodes represent entities and edges represent relationships or interactions between these entities. For instance, a social network can be modeled as a graph where users are represented as nodes and their relationships are represented as edges; a sensor network is a graph composed of sensors and their closeness as edges; and a molecule can be represented as a graph where atoms are nodes and bonds are edges. This representation allows various graph-based methods to process related graph data and perform downstream tasks such as classifying malicious users in the social network, route planning in road networks, and classifying molecules.

In these applications, the networked graph data often refers to data (or signals) associated with the *nodes* of a graph, as shown in Fig. 1.1d. In a sensor network, the sensor measurements at a timestamp may be modeled as a node signal. The node signal could also be the user profiles in a social network, or the atom properties in a molecule. In addition to

1



(a) 1D time span. (b) 2D image grid.

(c) graph. (d) node signal.

Figure 1.1: Discretized time span and image grid can be viewed as special graphs.

node signals, networked data can also be associated with the edges in a network. Such *edge data* may represent dynamic processes with both magnitudes and orientations, such as energy flows in power grids [Jia et al., 2019] or synaptic signals between neurons in brain networks [Faskowitz et al., 2022]. In financial markets, currencies can be modeled as nodes and exchange paths as edges, with exchange rates viewed as signals on the edges [Jiang et al., 2011]. Thus, we also call them *edge flows*. Sometimes, we have both node data and edge flows. Consider a water supply network, as shown in Fig. 1.2, where the nodes (tanks) have the water pressure or head data, and the edges (pipes) support the water flow rates.

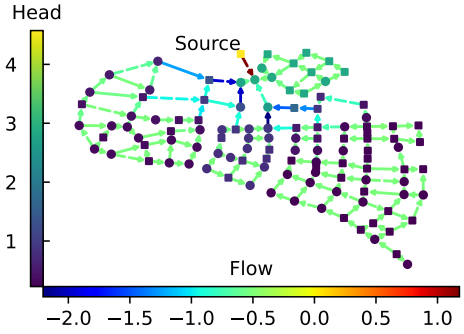


Figure 1.2: A water supply network with node data (water pressure/head) and edge data (water flow).

The recent development of machine learning for networked data revolves around the node data. *Graph neural networks* are among the most outstanding methods for graph machine learning. While there are many forms of graph neural networks, they are generally

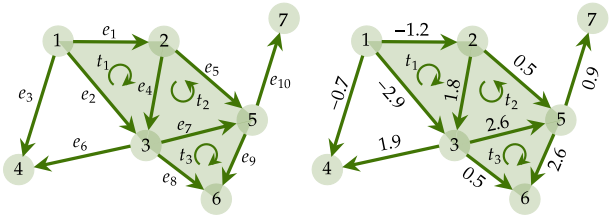
developed from the most modest form of graph convolutional networks (GCNs) [Bruna et al., 2013; Defferrard et al., 2016; Duvenaud et al., 2015; Kipf & Welling, 2017]. The fundamental idea of GCNs is to generalize the *convolution* operation from regular Euclidean domains to the graph domain. Essentially, the *graph convolution* is a *local, learnable* and *linear* transformation. It aggregates data from the *neighboring* nodes of each node, weighted by learnable parameters. This utilizes the topological structure encoded in the graph model to learn the node representations such that GCNs serve as a general framework for various graph-based tasks, such as node classification, link prediction, and graph classification.

The concept of *convolution* has long been fundamental in the field of classical *signal processing*. Similarly, in *graph signal processing* [Shuman et al., 2013], graph convolution based on the *graph Laplacian* has been better understood under the help of *spectral graph theory* [Chung, 1997]. This generalizes the notion of *frequency* to the graph domain, i.e., *graph frequency*, and allows for the construction of signal processing tools like graph filters [Sandryhaila & Moura, 2013; 2014]. Notably, the graph convolutions in the aforementioned GCNs are closely related to the graph convolutional filters [Gama et al., 2020b; Isufi et al., 2024]. In short, the graph convolutional filters are the building blocks of GCNs, and they are the key to the success of GCNs in various graph-based tasks. Here, we refer readers to the recent work of Isufi et al. [2024] which offers a comprehensive review of graph filters for *graph signal processing and graph machine learning*.

In this thesis, we go **beyond graphs and node signals**. First, we focus on a more general network model, namely, *simplicial complexes*. Simplices can be viewed as sets of vertices along with a relational structure. A 0-simplex is a node, a 1-simplex is an edge, a 2-simplex is a triangular face (triangle), and so on. To put it simply, a simplicial complex is a collection of *simplices*, where each subset of the simplices is also included in the complex. Graphs, also known as simplicial 1-complexes, are then a special case of simplicial complexes where only 0-simplices (nodes) and 1-simplices (edges) are present. The connectivity between nodes can be encoded in the *graph adjacency* matrix, indicating if two nodes are connected, and the relationships between nodes and edges are stored in the *node-edge incidence matrix*, showing which two nodes form the boundaries of an edge. A simplicial 2-complex is then a generalization of a graph where 2-simplices (triangles) are also present. To represent the richer topological structure, in addition to the graph representations, we also need the higher-order (*edge-triangle*) incidence matrices, showing which three edges form the boundaries of a triangle.

Secondly, instead of node signals, we focus on signals associated with the simplices of a simplicial complex, namely, *simplicial signals*. Node signals and edge flows are then also called 0- and 1-simplicial signals. To accommodate the directional information of simplicial signals, e.g., edge flows, we may additionally assign an *orientation* to each simplex in the complex, i.e., an ordering of the vertices. For an edge $e = \{i, j\}$, we choose the orientation as $e = [i, j]$ for $i < j$. For a triangle $t = \{i, j, k\}$, we choose the orientation as $t = [i, j, k]$ for $i < j < k$. This provides a reference for the signal directions on simplices with respect to their chosen orientation — if the signal value is positive, it is then aligned with the orientation; otherwise, it is against the orientation.

We refer to Fig. 1.3 for an *oriented* simplicial 2-complex and an edge flow. Compared to the



(a) simplicial 2-complex. (b) edge flow.

Figure 1.3: An *oriented* simplicial 2-complex and an edge flow signal.

graph in Fig. 1.1c and the node signal in Fig. 1.1d, the simplicial 2-complex includes also triangles (shaded in green) and the additional chosen orientations for edges (and triangles, denoted by arrows) to indicate the directions of each edge flow signal. Out ultimate goal is to exploit this more complex local and structual information between the signal elements to enable machine learning tasks.

1.1 Early Works

By the beginning of this PhD project, the research on *simplicial complexes* and *simplicial signals* was still in its infancy. Yet, a few works among others laid noteworthy foundations to enable our research. Grady & Polimeni [2010]; Lim [2020]; Lovász [2004] first introduced the notions of *gradient, divergence* and *curl* operators on simplicial complexes, which are closely related to the algebraic representations (specifically, the incidence matrices) of topological structures in the simplicial complexes. These *first-order discrete derivative* operators enable us to perform fundamental *calculus* on simplicial complexes and, more importantly, make the celebrated *Hodge decomposition* theory applicable to simplicial signals. When it comes to edge flows, this decouples the flow into *divergence-free* and *curl-free* components — the former has the property that the flow is conserved at each node, meaning that the total in-flow equals the total out-flow at each node; and the latter has the property of being *irrotational*, namely, the flow does not circulate around any loop (triangles). This decomposition plays an important role in real-world applications. For example, in circuit networks, the electric current respects the Kirchhoff's current law, being divergence-free, while the electric voltage respects the Kirchhoff's voltage law, being curl-free [Grady & Polimeni, 2010]. Similarly, water flows in water supply networks respect the conservation of mass, being divergence-free [Zhou et al., 2022]. Jiang et al. [2011] also introduced this decomposition to analyze the statistical ranking problem, and interpreted exchange rates respecting the arbitrage property in foreigen currency exchange markets as curl-free edge flows. The Hodge decomposition also finds its application in the analysis of game theory problems [Candogan et al., 2011], and in various other fields, e.g., brain networks [Vijay Anand et al., 2022], health care delivery networks [Gebhart et al., 2021] and finance networks [Fujiwara & Islam, 2020; Wand et al., 2024].

Moreover, the signal processing work on simplicial complexes was initiated by Barbarossa & Sardellitti [2020]; Barbarossa et al. [2018] where the notions of simplicial signals and

simplicial Fourier transform were introduced. Early works of Jia et al. [2019]; Schaub & Segarra [2018] investigated the signal processing tasks of flow smoothing, denoising and interpolations in the context of graphs. In analogy to how graph Laplacians underpin graph signal processing, these works are built upon the *Hodge Laplacian*—a generalization of the graph Laplacian to simplicial complexes, constructed from the incidence matrices. These works pioneer the signal processing from graphs to simplicial complexes, and drive the follow-up research on general signal processing and machine learning on simplicial complexes, including this thesis. Note that in parallel to our development, a branch of graph neural networks started looking into leveraging topological structures for graph machine learning tasks like graph or node classification [Bodnar et al., 2021b; Bunch et al., 2020; Ebli et al., 2020; Roddenberry & Segarra, 2019]. These works mainly generalize the convolution or message-passing scheme *directly* from graphs to simplicial complexes, without considering the Hodge decomposition of simplicial signals.

1.2 Research Goals and Contributions

In this thesis, with the goal of developing tools for *signal processing and machine learning on simplicial complexes*, we aim to provide researchers and practitioners with a principled framework for modeling, analyzing, processing and learning from the networked simplicial data—such as aforementioned edge flows arising from real-world applications. They should allow for downstream tasks like denoising, interpolation and predictions on simplicial complexes, going beyond node signals on graphs. However, while simplicial complexes offer a richer topological structure than graphs and simplicial signals enjoy more flexibility than node signals owing to the Hodge decomposition, they also pose new challenges compared to graph signal processing and machine learning that necessiate a different perspective. This in turn motivates the design of principled and flexible tools for simplicial signals that not only leverage the topological structure of simplicial complexes, but also respect the Hodge decomposition of simplicial signals. To this end, we address the following four research questions.

1 *How to process simplicial signals efficiently to achieve a desired output by leveraging the simplicial topology and the Hodge decomposition?*

This is addressed in Chapter 2, which is based on Yang et al. [2021; 2022b]:

- ***Maosheng Yang**, Elvin Isufi, Michael T. Schaub, and Geert Leus. Finite Impulse Response Filters for Simplicial Complexes. In 29th European Signal Processing Conference (EUSIPCO), 2021.*
- ***Maosheng Yang**, Elvin Isufi, Michael T. Schaub, and Geert Leus. Simplicial Convolutional Filters. IEEE Transactions on Signal Processing, 2022.*

We develop the *simplicial convolution* operation by leveraging both the simplicial topology and the Hodge decomposition. Specifically, we construct the *simplicial convolutional filters* to process simplicial signals in an efficient manner—by aggregating information within local

1

simplicial neighborhoods. More importantly, simplicial convolution has a clear spectral interpretation. Upon building the simplicial Fourier transform in terms of the Hodge decomposition, we analyze the frequency responses of the filters, and show that they are able to regulate the different Hodge components of the signals individually. This allows us to design filters that can preserve certain Hodge components for edge flows that are either divergence-free or curl-free. Simplicial convolution enjoys not only the locality and desired efficiency in the spatial domain, but also flexible and interpretable properties in the spectral domain, respecting the *convolution theorem* in classical signal processing. Similar to the role of *any* convolution, it lays the foundation for building more advanced signal processing tools, and allows for the design of *convolutional neural networks* for learning on simplicial complexes.

Simplicial convolutional filters perform however a limited *linear* operation. Having witnessed the success of *learning models* in various domains, e.g., neural networks, we are motivated to design more powerful learning models on simplicial complexes.

2 *How to perform efficient and interpretable learning on simplicial complexes?*

This is addressed in Chapter 3, which is based on Isufi & Yang [2022]; Yang et al. [2025]:

- ***Maosheng Yang**, Elvin Isufi, and Geert Leus. Simplicial Convolutional Neural Networks. In IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP), 2022.*
- ***Maosheng Yang**, Geert Leus, and Elvin Isufi. Hodge-Aware Convolutional Learning on Simplicial Complexes. Transactions on Machine Learning Research, 2025.*

We propose *convolutional neural networks* for learning on simplicial complexes, which are built upon composing multi-layer simplicial convolutions with nonlinear activation functions. Inheriting the properties of simplicial convolutions, the proposed models are *efficient* since the learning is performed within local simplicial neighborhoods, and *interpretable* since the learned is independent in different Hodge subspaces and able to capture the different Hodge components, namely, Hodge-aware. They are flexible to perform learning across simplices of different orders, e.g., nodes, edges and triangles, incorporating the inter-simplicial interactions. We also investigate the robustness of the models against perturbations in the simplicial topology, showing that they are stable to small perturbations. These models offer a principled design to convolutional learning on simplicial complexes, demonstrating strong performance across various applications, including financial markets, simplex prediction and trajectory prediction.

The first two research questions focus on the *deterministic* method for processing and learning on simpplicial complexes. However, often learning benefits from *probabilistic* methods because they account for the uncertainty in the data and model, offer more interpretability to model performance, and allow for uncertainty quantification. One fundamental and tractable approach that provides probabilistic insight into learning is to rely on Gaussian process modeling. Existing works however do not leverage the topology

and it remains unclear how to incorporate this topological prior into the modeling of simplicial signals. This leads to the following question.

3 *How to model simplicial signals using Gaussian processes and perform learning?*

This is addressed in Chapter 4, which is based on Yang et al. [2024]:

- ***Maosheng Yang**, Viacheslav Borovitskiy, and Elvin Isufi. Hodge-Compositional Edge Gaussian Processes. In International Conference on Artificial Intelligence and Statistics (AISTATS), 2024.*

We construct simplicial Gaussian processes as random functions defined over simplices, where any collection of these function values follows a multivariate Gaussian distribution. We can then view simplicial signals such as edge flows as samples from such a Gaussian process. We detail the formulation of Matérn Gaussian processes, where the *diffusion* instant is associated with the heat diffusion on simplicial complexes. To capture the practical properties of real-world edge flows being either divergence-free or curl-free, we further build the Gaussian processes as a composition of three, each only modeling a specific part given by the Hodge decomposition. This allows us to perform separate learning of different Hodge components. We demonstrate their performance in financial markets, ocean current analysis and water supply networks, bringing new approaches with both performance and interpretability to these applications. Our construction not only extends the framework of Gaussian processes for irregular domains, but also provides a principled way of probabilistic modeling of simplicial signals that respects the Hodge decomposition.

Given the previous probabilistic view of simplicial signals, we consider the more challenging task of matching two simplicial signal distributions. Distribution matching is the fundamental problem in generative modeling, which aims to learn a transformation from one data distribution to another for efficient sampling and inference. When one is a noise distribution, this promotes the generative modeling of simplicial signals and tackles the lack of lagre amount of labeled data in real-world applications. In the case of two data distributions, this allows us to learn a transport map between simplicial signals, providing insights to, for example, how divergence-free edge flows transform to curl-free flows. Therefore, we are interested in the following question.

4 *How to perform topology-aware generative learning of simplicial signals?*

This is addressed in Chapter 5, which is based on Yang [2025]:

- ***Maosheng Yang**. Topological Schrödinger Bridge Matching. In The Thirteenth International Conference on Learning Representations (ICLR), 2025.*

We consider the Schrödinger bridge problem, originated from Schrödinger's gas experiments [Schrödinger, 1931; 1932], which seeks the *most likely* random evolution between two boundary distributions with respect to a reference process. It is a *dynamic* version

1

of the optimal transport between two distributions [Villani, 2009]. In the calssical setting, the reference process is a Brownian motion. Schrödinger bridge based methods have been used in image generation and image-to-image translation, which unify other alternatives such as the popular *score matching* and *flow matching* based methods [Chen et al., 2022b; De Bortoli et al., 2021; Lipman et al., 2022; Song et al., 2020b].

We adapt the problem to graphs and simplicial complexes, referred to as *topological Schrödinger bridge problem*, where we set the reference to be driven by the simplicial convolution, which is *topology-aware*, *linear* and *tractable* e.g., the stochastic heat diffusions on graphs and simplicial complexes. The solution to this problem follows a forward-backward topological stochastic process with some *hard-to-solve* unknown terms. Upon this result, we propose the topological Schrödinger bridge matching models for topological signal generative modeling and maching. Specifically, we parameterize the unknowns by some (topology-aware) learnable models (e.g., graph/simplicial neural networks), and train them by maximizing the likelihood of the model based on the framework of Chen et al. [2022b]. Such models help us to build primitive dynamics on how brain signals evolve from one state to another, or how ocean currents to evolve from being curl-free to divergence-free. In single-cell dynamics, they excel in inferring the intermediate states of cells given the initial and final observations, which is a challenging task in single-cell RNA sequencing data analysis. They also help practitioners to generate graph and simplicial signals such as seismic events and traffic flows for data-hungry applications.

Overall, this thesis extends the field of *signal processing and machine learning on graphs*, and positions its contributions in the emerging fields of *topological signal processing* and *topological machine learning*. This thesis provides not only the fundamental and systematic theoretical analyses of the methods, but also justifies the methods in various real-world applications. Though some of the initial ideas are developed and extended from graph signal processing and machine learning, our proposed methods do not take routes away from the important principle of *Hodge-decomposition* of simplicial signals, as well as the *simplicial spectral theory*. For example, our convolutional filters are designed to process the different Hodge components of the signals *individually*, our neural network models are able to learn from the different components *separately*, and our Gaussian processes model the different components *independently*. Driven by real-world applications, as briefly discussed earlier and will also be detailed per chapter, this offers more flexibility and interpretability in the signal processing and machine learning tasks on simplicial complexes.

Recommended mathematical reading

In this thesis, beyond the standard signal processing and machine learning curriculum, we also make use of mathematical tools including (spectral) graph theory and fundamentals on simplicial complexes like simplicial signals, discrete calculus and Hodge Laplacians. Though they can be accessed by basic linear algebra, we refer readers to the pioneering works of Barbarossa & Sardellitti [2020] and Lim [2020] for the introduction of simplicial complexes where simplicial signals and properties of Hodge Laplacians are studied. However, note that in each chapter, we provide the necessary background from the above two materials for an accessible and self-contained reading.

For readers who are familar with *graph signal processing*, we refer to Isufi et al. [2024] for the most recent overview on the role of graph filters in graph signal processing and machine learning. This helps understand the importance of building simplicial convolutional filters for more advanced signal processing and machine learning methods on simplicial complexes. We also refer to Schaub et al. [2021] for a smooth overview on the signal processing from graphs to simplicial complexes. For readers who are familar with graph neural networks, we refer to Besta et al. [2024] for a recent overview of the neural networks on higher-order networks.

Out-takes*

In order to keep the thesis focused and concise, I have omitted some other works I collaborated on with my supervisors and colleagues during the PhD. Here, we give a brief overview of these:

1. *Topological Volterra Filters* [Leus et al., 2021] — *Volterra-type* filters on graphs taking into account the node-tuple interactions, led by my promotor Geert Leus, where I contributed to the experimental design and implementations.

2. *Online Filter Learning on Simplicial Complexes* [Yang et al., 2023] — Online learning of simplicial convolutional filters on expanding simplicial complexes where the edges join the complex, where I contributed to the method design and analysis, under the help of my colleague Bishwadeep Das, and the experimental implementation.

3. *State Estimation in Water Distribution System via Diffusion on the Edge Space* [Kerimov et al., 2025] — Estimating of the pressure and flow rates in water distribution systems based on the edge diffusion process, where I contributed to the method design and part of the paper writing.

2

Simplicial Convolutional Filters

In this chapter, we aim to build a convolution operator on simplicial complexes that can be used to process simplicial signals. Such an operator can be built upon the existing works (as reviewed in Isufi et al. [2024]) on graph convolution operators. These graph convolution operators are able to regulate graph signals in terms of the graph frequency, which measures the smoothness of graph signals. To construct such an operator for simplicial signals, it is necessary to have a clear understaning on the notion of frequency of simplicial signals. If the simplicial frequency also measures the smoothness of simplicial signals, how should this smoothness be defined? Moreover, as discussed in Chapter 1, simplicial signals admit the Hodge decomposition where different components have distinguishing properties — different parts of edge flows reflect the conservation and rotational properties, respectively. Thus, we intend to build the convolution operator that acts differently on the different Hodge components such that we have individual control on the Hodge components. This will allow us, for example, to preserve certain properties of the signals, such as the conservation of flows, while filtering out the rotational noisy parts. Finally, we aim to design the simplicial convolution operators such that they can achieve the desired filtering properties. This chapter is based on the work of Yang et al. [2021; 2022b].

2.1 Introduction

Methods to process signals supported on non-Euclidean domains modeled as graphs have attracted substantial research interest recently. Most of these graph signal processing (GSP) methods focus on signals supported on nodes, e.g., temperature measurements in weather stations network or EEG signals in a brain network [Ortega et al., 2018; Shuman et al., 2013]. Using linear shift operators that couple node signals to each other via the edges of a graph, e.g., in terms of an adjacency or a Laplacian matrix, we can design graph filters to process such node signals [Ortega et al., 2018; Sandryhaila & Moura, 2013; 2014; Shuman et al., 2013].

However, we often encounter signals that are naturally associated with edges or sets of nodes in real-world applications. For example, blood flow between different areas in the brain [Huang et al., 2018], water flow in a hydrological network, data flow in a communication network, or traffic flow in a road network [Leung et al., 1994; Schaub & Segarra, 2018]. We typically model these signals as a flow over the edges of a network. Edge flows have also been used in statistical ranking, to describe financial markets, to analyze games, etc. [Candogan et al., 2011; Gebhart et al., 2021; Jiang et al., 2011; Mock & Volic, 2021]. Similarly, we may even encounter signals supported on sets of nodes [Bick et al., 2023; Huang & Ribeiro, 2015]. For instance, in a co-authorship network, the number of publications with more than two authors can be seen as such a signal [Patania et al., 2017].

In these cases, rather than focusing on utilizing the relationships between the nodes to process node signals, it can be fruitful to study relations between the node-relationships (edges, higher-order edges) themselves. In the case of edge-signals, we want to understand couplings between edges, e.g., mediated through a common incident node (lower adjacency) or because these edges contribute to a triadic relation (upper adjacency). To account for such relationships, we can model a (network) system as a simplicial complex (SC). Using this representation we can analyze signals associated to subsets of nodes, i.e., simplicial signals, with algebraic tools via so-called Hodge Laplacians [Barbarossa & Sardellitti, 2020; Grady & Polimeni, 2010; Lim, 2020], which generalize the familiar graph Laplacians.

In addition, the Hodge Laplacian admits a Hodge decomposition, which allows for an intuitive physical interpretation of signals supported on SCs [Grady & Polimeni, 2010; Lim, 2020]. Namely, the Hodge decomposition states that any edge flow can be decomposed into gradient (curl-free), curl (divergence-free) and harmonic components, respectively. For instance, a water flow may contain a non-cyclic component which can be seen as the potential difference between water stations, a locally cyclic component with non-zero curl and a harmonic component being flow-conservative [Schaub & Segarra, 2018].

Previous works [Barbarossa & Sardellitti, 2020; Schaub et al., 2021] have established a framework to analyze simplicial signals and focused mostly on low-pass filtering applications. However, general linear filters for simplicial signals have not been considered in detail. In this chapter, we propose a *simplicial convolutional filter* via the shift-and-sum operation as a matrix polynomial of the Hodge Laplacians to enable a flexible simplicial signal filtering. Our filter accounts for lower and upper adjacencies in an SC, e.g., the relationships between edges via a common node or a common 2-simplex, and allows to separately filter the three signal components provided by the Hodge decomposition.

Contributions. Our three main contributions include:

1) Simplicial convolution. We study simplicial shifting via the Hodge Laplacians as a basic operation to propagate signals locally using both lower- and upper-connectivities in an SC. Leveraging this shifting and the shift-and-sum operation, we develop simplicial convolutional filters by aggregating multi-step shifted signals. Their local shifting operation allows a distributed implementation of the filter with a cost linear in the number of simplices. We show such filters are linear, shift-invariant, and equivariant to permutations of the labeling and the orientation of simplices.

2) Filtering in the spectral domain. Leveraging the simplicial Fourier transform (SFT) [Barbarossa & Sardellitti, 2020], we show that the principles of the convolutional theorem apply to the proposed filter, i.e., the filter output in the frequency domain operates as a point-wise multiplication between the filter frequency response and the SFT of the input signal. We further show how the simplicial frequencies act as measures of signal variations w.r.t. the lower and upper adjacencies and divide them into gradient, curl and harmonic frequencies. More precisely, the eigenmodes associated to these frequencies span the subspaces provided by the Hodge decomposition. Ultimately, this implies that the proposed filter can regulate signals independently in the three subspaces provided by the Hodge decomposition.

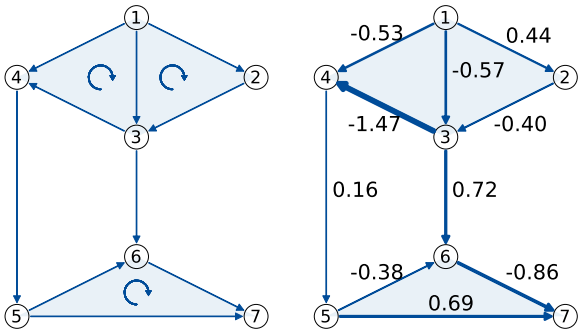
3) Filter Design. To implement a desired frequency response, we first consider a standard least-squares (LS) approach to design the filter. To avoid the eigenvalue computation, we then consider a grid-based universal design. As both strategies may suffer from numerical instability, we propose a numerically more stable Chebyshev polynomial design.

Related works. The idea of processing of signals defined on manifolds and topological spaces has been discussed in various areas, such as geometry processing [Botsch et al., 2010], and topological data analysis [Wasserman, 2018]. For an introduction that is geared more towards a signal processing perspective see Barbarossa & Sardellitti [2020]; Grady & Polimeni [2010].

Filtering of simplicial signals has been partly approached from a regularization perspective. The works in Schaub & Segarra [2018]; Schaub et al. [2021] proposed a regularized optimization framework based on (simplified variants of) the Hodge Laplacian, to promote flow conservation of the resulting estimated edge flows. The solution is a low-pass simplicial filter. The same regularizer was used in Jia et al. [2019]; Schaub et al. [2021] to perform edge flow interpolation by exploiting the divergence-free and curl-free behaviors of real-world flows. However, these assumptions do not always hold and the filters arising from the considered regularized optimization problems have limited degrees of freedom.

Filtering simplicial signals has also been analyzed in the Fourier domain, akin to how graph filters are analyzed via the graph Fourier transform [Shuman et al., 2013]. The analogous SFT, defined via the eigendecomposition of the Hodge Laplacian was described in Barbarossa & Sardellitti [2020]. The eigenvectors provide a simplicial Fourier basis and the eigenvalues carry a notion of frequency.

Outline. We begin by introducing some preliminaries in Section 2.2. Then we propose the simplicial convolutional filter in Section 2.3 and investigate its properties. In Section 2.4, we introduce the simplicial Fourier transform. We then analyze the simplicial filter in the spectral domain and study the notion of simplicial frequency. Different filter design methods are discussed in Section 2.5. Finally, we use simplicial filters for subcomponent extraction and edge flow denoising, and consider applications to financial market and transportation networks in Section 2.6.



(a) A simplicial complex. (b) A random edge flow.

Figure 2.1: Simplicial Complexes and Signals. (a): An SC of order 2 containing seven nodes, ten edges and three 2-simplex (the shaded filled triangles). Reference orientations of the simplices are indicate by corresponding arrows (the reference orientation of a node is trivial). (b): An arbitrary edge flow, where a negative flow indicates that the actual flow direction is opposite to the reference orientation and the magnitude is denoted by the edge width.

2.2 Background: Simplicial Complexes, Signals and Hodge Laplacians

In this section, we review SCs [Lim, 2020] and signals supported on simplices [Barbarossa & Sardellitti, 2020; Schaub et al., 2021]. We also introduce the Hodge Laplacians as an algebraic representation of a simplicial complex, that acts as a shift operator for simplicial signals.

Simplicial complexes. Given a finite set of vertices $\mathcal{V}$, a k-simplex s^k is a subset of $\mathcal{V}$ with cardinality $k+1$. A subset of a k-simplex s^k with cardinality k is called a *face* of s^k; hence, s^k has $k+1$ faces. A *coface* of a k-simplex is a simplex s^{k+1} that includes it. A simplicial complex $\mathcal{S}$ is a finite collection of simplices with an inclusion property: for any $s^k \in \mathcal{S}$ all its faces $s^{k-1} \subset s^k$ are also part of the simplicial complex, i.e., $s^{k-1} \in \mathcal{S}$. The order of an SC is the largest order of its simplices. For an SC of order K, we collect the k-simplices into a set $\mathcal{S}^k = \{s_1^k, \ldots, s_{N_k}^k\}$ where $N_k = |\mathcal{S}^k|$ is the number of k-simplices [Barbarossa & Sardellitti, 2020; Lim, 2020].

Based on its geometric realizations, we call a 0-simplex a node, a 1-simplex an edge and a 2-simplex a (filled) triangle. Note that an "empty triangle" formed by three vertices and three pairwise relations between them is not a 2-simplex. Henceforth, we refer to 2-simplices as triangles for simplicity. A graph is an SC of order 1 with N_0 nodes and N_1 edges. Fig. 2.1a shows an SC of order 2 including nodes, edges and triangles where edge $\{5,6\}$ has nodes $\{5\}$ and $\{6\}$ as its faces and triangle $\{5,6,7\}$ as its coface.

Two k-simplices in an SC are *lower adjacent* if they have a common face and are *upper adjacent* if they are both faces of a common $(k+1)$-simplex. Thus, for the simplex s_i^k, we define its *lower neighborhood* $\mathcal{N}_{i,\mathrm{d}}^k$ as the set of its lower adjacent k-simplices and its *upper*

neighborhood $\mathcal{N}^k_{i,\mathrm{u}}$ as the set of its upper adjacent k-simplices. For the ith edge $\{5,6\}$ in Fig. 2.1a, we have that $\mathcal{N}^1_{i,\mathrm{d}} = \{\{4,5\},\{3,6\},\{5,7\},\{6,7\}\}$, and the upper neighborhood is $\mathcal{N}^1_{i,\mathrm{u}} = \{\{5,7\},\{6,7\}\}$.

Simplicial signals. For computational purposes, we fix an (arbitrary) reference orientation (see Lim [2020] and Schaub et al. [2020, p. 5] for more details) for each simplex according to the lexicographical ordering of its vertices. Given this reference orientation for k-simplices, we define a *k-simplicial signal* $\mathbf{x}^k = [x^k_1,\dots,x^k_{N_k}]^\top \in \mathbb{R}^{N_k}$ by attributing the value x^k_i to the ith k-simplex s^k_i. If the signal value x^k_i is positive, then the corresponding signal is aligned with the reference orientation; opposite otherwise. Fig. 2.1b illustrates an arbitrary edge flow on an SC, in which some flows are aligned with and other are opposite the reference orientation (negative). For convenience, henceforth, we denote a node signal by $\mathbf{v} = [v_1,\dots,v_{N_0}]^\top \in \mathbb{R}^{N_0}$ and an edge flow by $\mathbf{f} = [f_1,\dots,f_{N_1}]^\top \in \mathbb{R}^{N_1}$.

Hodge Laplacians. We can describe the relationships between $(k-1)$-simplices and k-simplices by the kth incidence matrix $\mathbf{B}_k \in \mathbb{R}^{N_{k-1}\times N_k}$, which maps each k-simplex to the $(k-1)$-simplices that are its faces; cf. [Barbarossa & Sardellitti, 2020; Lim, 2020] for more details. Specifically, $\mathbf{B}_1$ is the node-edge incidence matrix, and $\mathbf{B}_2$ is the edge-triangle incidence matrix. The rows of $\mathbf{B}_k$ are indexed by $(k-1)$-simplices and the columns by k-simplices, with entries defined as:

$$[\mathbf{B}_k]_{ij} = \begin{cases} 1, \text{ if } s^{k-1}_i \subset s^k_j \text{ and } s^{k-1}_i \sim s^k_j \\ -1, \text{ if } s^{k-1}_i \subset s^k_j \text{ and } s^{k-1}_i \nsim s^k_j \\ 0, \text{ otherwise,} \end{cases} \tag{2.1}$$

where $s^{k-1}_i \sim s^k_j$ indicates the orientation of s^{k-1}_i is aligned with that of s^k_j and $\nsim$ denotes their orientations are opposite [Barbarossa & Sardellitti, 2020]. For example, in Fig. 2.1a, edge $\{5,6\}$ and $\{6,7\}$ are aligned with triangle $\{5,6,7\}$, while $\{5,7\}$ is not. As the ordering of a node is trivial, we consider the node-to-edge incidence matrix $\mathbf{B}_1$ as in graph theory. $\mathbf{B}_2$ is the edge-to-triangle incidence matrix. By definition, incidence matrices have the property [Barbarossa & Sardellitti, 2020; Lim, 2020]

$$\mathbf{B}_k\mathbf{B}_{k+1} = \mathbf{0}. \tag{2.2}$$

Upon defining the incidence matrices, we can describe an SC $\mathcal{S}$ of order K via the Hodge Laplacians

$$\mathbf{L}_k = \mathbf{B}_k^\top\mathbf{B}_k + \mathbf{B}_{k+1}\mathbf{B}_{k+1}^\top, \quad k = 1,\dots,K-1, \tag{2.3}$$

with the graph Laplacian $\mathbf{L}_0 = \mathbf{B}_1\mathbf{B}_1^\top$ and $\mathbf{L}_K = \mathbf{B}_K^\top\mathbf{B}_K$. The kth-*Hodge Laplacian* $\mathbf{L}_k$ contains the *lower Laplacian* $\mathbf{L}_{k,\mathrm{d}} := \mathbf{B}_k^\top\mathbf{B}_k$ and the *upper Laplacian* $\mathbf{L}_{k,\mathrm{u}} := \mathbf{B}_{k+1}\mathbf{B}_{k+1}^\top$. The lower Laplacian encodes the lower adjacency relationships between simplices through faces while the upper one encodes the upper adjacency relationships through cofaces. In particular, $\mathbf{L}_{1,\mathrm{d}}$ encodes the edge adjacencies through their incident nodes and $\mathbf{L}_{1,\mathrm{u}}$ through the common triangles that they form.

2.3 Simplicial Convolutional Filters

In this section, we propose a simplicial convolutional filter based on the Hodge Laplacian. We study its basic building block, the simplicial shifting, and show how its local characteristics make it amenable to a distributed implementation. We then look into the properties of shift-invariance and permutation and orientation equivariance in the simplicial domain.

2

Given the kth-Hodge Laplacian $\mathbf{L}_k$, we define a *simplicial convolutional filter* to process a k-simplicial signal $\mathbf{x}^k$ as

$$\mathbf{H}_k = h_0\mathbf{I} + \sum_{l_1=1}^{L_1} \alpha_{l_1}(\mathbf{B}_k^\top \mathbf{B}_k)^{l_1} + \sum_{l_2=1}^{L_2} \beta_{l_2}(\mathbf{B}_{k+1}\mathbf{B}_{k+1}^\top)^{l_2}, \tag{2.4}$$

where $\mathbf{H}_k := \mathbf{H}(\mathbf{L}_{k,\mathrm{d}}, \mathbf{L}_{k,\mathrm{u}})$ is a matrix polynomial of the lower and upper Hodge Laplacians with filter coefficients h_0, $\boldsymbol{\alpha} = [\alpha_1, \ldots, \alpha_{L_1}]^\top$, $\boldsymbol{\beta} = [\beta_1, \ldots, \beta_{L_2}]^\top$ and filter orders L_1, L_2. When $k = 0$, we obtain the graph convolutional filter $\mathbf{H}_0 := \mathbf{H}(\mathbf{L}_0)$ built upon the graph Laplacian $\mathbf{L}_0$ [Sandryhaila & Moura, 2013; 2014; Shuman et al., 2013].

In the following, we will see that assigning two different sets of coefficients to the lower and upper Laplacian parts in $\mathbf{H}_k$ enables the filter to treat lower and upper adjacencies differently, which results in a more flexible control of the frequency response. Instead, the filter $\mathbf{H}_k = \sum_{l=0}^{L} h_l \mathbf{L}_k^l$, which is equivalent to setting $L_1 = L_2 = L$ and $\boldsymbol{\alpha} = \boldsymbol{\beta}$ in (2.4), cannot differentiate between the two types of adjacencies and loses some expressive power.

When applying $\mathbf{H}_k$ to a k-simplicial signal $\mathbf{x}^k$, it generates an output $\mathbf{x}_{\mathrm{o}}^k = \mathbf{H}_k\mathbf{x}^k$ which is a *shift-and-sum* operation where the filter $\mathbf{H}_k$ first shifts the signal L_1 times over the lower neighborhoods and L_2 times over the upper neighborhoods, and then sums the shifted results according the corresponding coefficients. This is analogous to the convolutions of graph signals, images and time series [Shuman et al., 2013]. For ease of exposition, we study the filtering process of an edge flow $\mathbf{f}$ via an edge filter $\mathbf{H}_1$ hereafter.

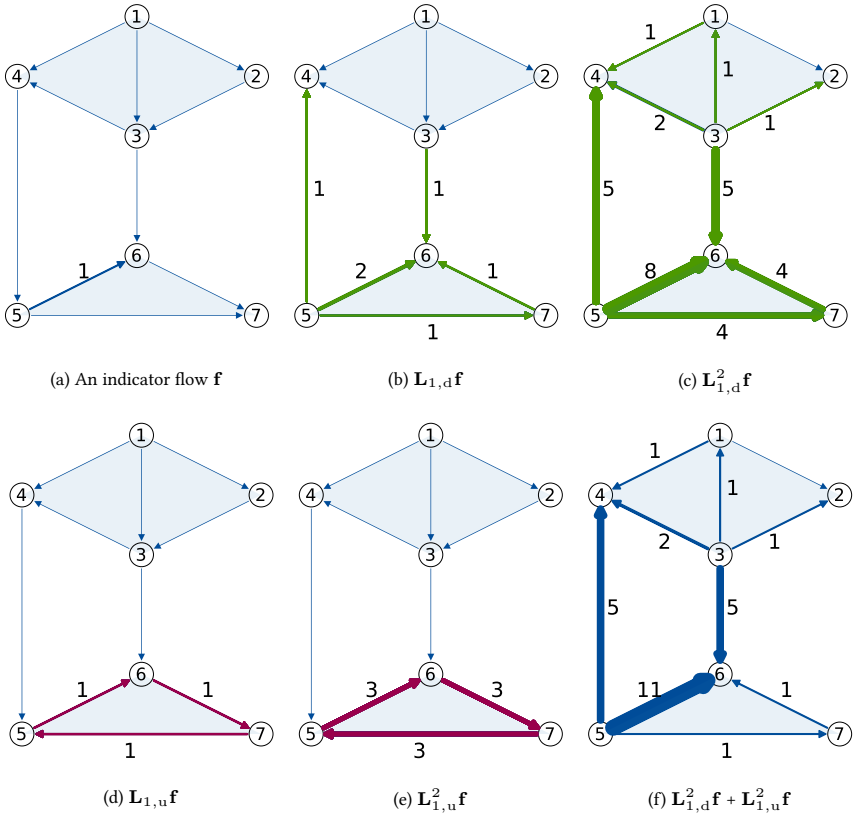
Simplicial shifting and local implementation. Consider an edge filter $\mathbf{H}_1$ applied to an edge flow $\mathbf{f}$ with an output

$$\mathbf{f}_{\mathrm{o}} = \mathbf{H}_1\mathbf{f} = h_0\mathbf{f} + \sum_{l_1=1}^{L_1} \alpha_{l_1}\mathbf{L}_{1,\mathrm{d}}^{l_1}\mathbf{f} + \sum_{l_2=1}^{L_2} \beta_{l_2}\mathbf{L}_{1,\mathrm{u}}^{l_2}\mathbf{f}, \tag{2.5}$$

where the basic operation consists of applying different powers of the lower/upper Hodge Laplacian to the edge flow. This basic operation is denoted *simplicial shifting*. Let us first consider the one-step lower shifting $\mathbf{f}_{\mathrm{d}}^{(1)} := \mathbf{L}_{1,\mathrm{d}}\mathbf{f}$ and one-step upper shifting $\mathbf{f}_{\mathrm{u}}^{(1)} := \mathbf{L}_{1,\mathrm{u}}\mathbf{f}$. We can express the one-step shifted results on the ith edge, $[\mathbf{f}_{\mathrm{d}}^{(1)}]_i$ and $[\mathbf{f}_{\mathrm{u}}^{(1)}]_i$, as

$$[\mathbf{f}_{\mathrm{d}}^{(1)}]_i = \textstyle\sum_{j\in\{\mathcal{N}_{\mathrm{d},i}^1 \cup\, i\}} [\mathbf{L}_{1,\mathrm{d}}]_{ij}[\mathbf{f}]_j, \quad [\mathbf{f}_{\mathrm{u}}^{(1)}]_i = \textstyle\sum_{j\in\{\mathcal{N}_{\mathrm{u},i}^1 \cup\, i\}} [\mathbf{L}_{1,\mathrm{u}}]_{ij}[\mathbf{f}]_j, \tag{2.6}$$

which are the weighted linear combinations of the edge flows on the lower and upper neighborhoods, $\mathcal{N}_{\mathrm{d},i}^1$ and $\mathcal{N}_{\mathrm{u},i}^1$, of edge i. This implies that one-step shifting is a *local operation* in the edge space within the direct lower/upper neighborhoods.



(a) An indicator flow $\mathbf{f}$ (b) $\mathbf{L}_{1,\mathrm{d}}\mathbf{f}$ (c) $\mathbf{L}^2_{1,\mathrm{d}}\mathbf{f}$

(d) $\mathbf{L}_{1,\mathrm{u}}\mathbf{f}$ (e) $\mathbf{L}^2_{1,\mathrm{u}}\mathbf{f}$ (f) $\mathbf{L}^2_{1,\mathrm{d}}\mathbf{f} + \mathbf{L}^2_{1,\mathrm{u}}\mathbf{f}$

Figure 2.2: Simplicial shifting. (a): An edge flow indicator $\mathbf{f}$ of edge $\{5,6\}$. (b): One-step lower shifting $\mathbf{L}_{1,\mathrm{d}}\mathbf{f}$. Edge $\{5,6\}$ and its direct lower neighboring edges (green) update their flows by aggregating information from their lower neighbors and themselves. (c): Two-step lower shifting $\mathbf{L}^2_{1,\mathrm{d}}\mathbf{f}$. lower neighboring edges (green) update their flows through faces within two hops away from edge $\{5,6\}$, which can be obtained by one-step shifting $\mathbf{L}_{1,\mathrm{d}}\mathbf{f}_{\mathrm{d}}$. (d): One-step upper shifting $\mathbf{L}_{1,\mathrm{u}}\mathbf{f}$. Edge $\{5,6\}$ and its upper neighbors (red) update their flows through local information aggregation. (e): Two-step upper shifting $\mathbf{L}^2_{1,\mathrm{u}}\mathbf{f}$. The output is localized within the one-hop upper neighborhood, as there is no upper neighboring edge two hops away from $\{5,6\}$. (f): Two-step shifting result $\mathbf{L}^2_1\mathbf{f}$, as the sum of (c) and (e).

Consider now the l-step lower shifting of an edge flow $\mathbf{f}$, $\mathbf{f}_{\mathrm{d}}^{(l)} := \mathbf{L}^l_{1,\mathrm{d}}\mathbf{f} = \mathbf{L}_{1,\mathrm{d}}\mathbf{f}_{\mathrm{d}}^{(l-1)}$, where the second equality indicates that the l-step lower shifting can be computed as a one-step shifting of the previously shifted result, $\mathbf{f}_{\mathrm{d}}^{(l-1)}$. Accordingly, the l-step upper shifting follows $\mathbf{f}_{\mathrm{u}}^{(l)} := \mathbf{L}^l_{1,\mathrm{u}}\mathbf{f} = \mathbf{L}_{1,\mathrm{u}}\mathbf{f}_{\mathrm{u}}^{(l-1)}$. Thus, the simplicial shifting allows a recursive implementation. For example, the two-step shifted results, $\mathbf{f}_{\mathrm{d}}^{(2)}$ and $\mathbf{f}_{\mathrm{u}}^{(2)}$, can be computed from $\mathbf{f}_{\mathrm{d}}^{(1)}$ and $\mathbf{f}_{\mathrm{u}}^{(1)}$ via another shifting. Each edge thus collects the flows from its lower and upper neighbors two hops away. Fig. 2.2 illustrates such shifting operations. Likewise, the l-step shifted results $\mathbf{f}_{\mathrm{d}}^{(l)}$ and $\mathbf{f}_{\mathrm{u}}^{(l)}$ contain the information up to the l-hop lower and

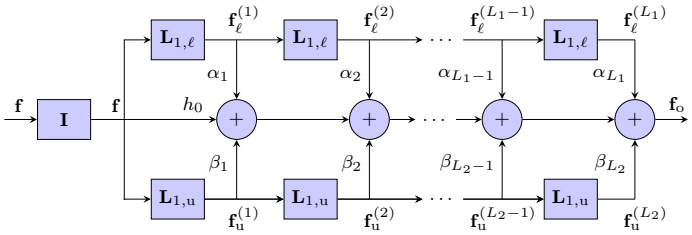


Figure 2.3: Simplicial convolutional filtering is a shift-and-sum operation.

upper neighborhoods. Finally, we can express the output (2.5) as

$$\mathbf{f}_\mathrm{o} = h_0\mathbf{f}^{(0)} + \textstyle\sum_{l_1=1}^{L_1} \alpha_{l_1}\mathbf{f}_\mathrm{d}^{(l_1)} + \sum_{l_2=1}^{L_2} \beta_{l_2}\mathbf{f}_\mathrm{u}^{(l_2)}, \tag{2.7}$$

which is a weighted linear combination of lower and upper shifted simplicial signals after different steps. Fig. 2.3 illustrates such a shift-and-sum operation.

Thus, the simplicial convolutional filter admits a *simplicial locality*, i.e., the output $\mathbf{f}_\mathrm{o}$ is localized in the L_1-hop lower neighborhood and L_2-hop upper neighborhood of an edge. If two edges are lower adjacent more than L_1 hops away or upper adjacent more than L_2 hops away, $\mathbf{H}_1$ does not mix signals defined on such edges. Note, however, if two edges are lower or upper adjacent at distance d, a d-step lower or upper shifting via $\mathbf{L}_{1,\mathrm{d}}^d$ or $\mathbf{L}_{1,\mathrm{u}}^d$ does not necessarily cause an interaction between them: the aggregation might be cancelled out by the combination of the filter coefficients and the topology of the SC which leads to positive and negative entries in $\mathbf{L}_{1,\mathrm{d}}$ or $\mathbf{L}_{1,\mathrm{u}}$.

As a local operation within the simplicial neighborhood, the simplicial shifting allows for a distributed filter implementation, in which each edge updates its information only by a direct communication with its lower and upper neighbors. The communication complexity mainly comes from operation (2.6), which is of order $\mathcal{O}(D_\mathrm{d})$ with $D_\mathrm{d} := \max\{|\mathcal{N}_{\mathrm{d},i}^1|_{i=1}^{N_1}\}$ for the lower simplicial shifting at each edge and $\mathcal{O}(D_\mathrm{u})$ with $D_\mathrm{u} := \max\{|\mathcal{N}_{\mathrm{u},i}^1|_{i=1}^{N_1}\}$ for the upper simplicial shifting, i.e., the maximum number of lower and upper neighbors among all edges, respectively. Then, the total communication cost of the distributed implementation for each edge is $\mathcal{O}(D_\mathrm{d}L_1 + D_\mathrm{u}L_2)$ due to the L_1 lower shifting steps and L_2 upper ones.

Linearity and shift invariance. Simplicial convolutional filters are linear operators that are invariant to shifts.

Proposition 2.1. *The simplicial filter $\mathbf{H}_k$ [cf. (2.4)] is linear and shift-invariant. Specifically, in the edge space, given two edge flows $\mathbf{f}_1$ and $\mathbf{f}_2$ and a simplicial filter $\mathbf{H}_1$, we have*

$$\begin{aligned} \textit{Linearity}: \mathbf{H}_1(a\mathbf{f}_1 + b\mathbf{f}_2) &= a\mathbf{H}_1\mathbf{f}_1 + b\mathbf{H}_1\mathbf{f}_2, \\ \textit{Shift-invariance}: \mathbf{L}_{1,\mathrm{d}}\big(\mathbf{H}_1\mathbf{f}_1\big) &= \mathbf{H}_1\big(\mathbf{L}_{1,\mathrm{d}}\mathbf{f}_1\big), \\ \mathbf{L}_{1,\mathrm{u}}\big(\mathbf{H}_1\mathbf{f}_1\big) &= \mathbf{H}_1\big(\mathbf{L}_{1,\mathrm{u}}\mathbf{f}_1\big). \end{aligned} \tag{2.8}$$

Proof. See Appendix 2.A. □

The shift-invariance implies that applying a lower or upper Hodge Laplacian to the simplicial output is equivalent to applying them to the input signal prior to filtering. Consequently, it holds that $\mathbf{H}_1\mathbf{H}_1'\mathbf{f} = \mathbf{H}_1'\mathbf{H}_1\mathbf{f}$ for any two filters $\mathbf{H}_1$ and $\mathbf{H}_1'$.

Equivariance. In an SC, the labeling and the reference orientation of the simplices should not affect the filter output. We show that this is indeed the case. We can model the simplex relabeling of simplices by a set of permutation matrices [Roddenberry et al., 2021]

$$\mathcal{P} = \{\mathbf{P}_k \in \{0,1\}^{N_k \times N_k} : \mathbf{P}_k\mathbf{1} = \mathbf{1}, \mathbf{P}_k^\top\mathbf{1} = \mathbf{1}, k \geq 0\}. \tag{2.9}$$

Let $\bar{\mathcal{P}} = (\mathbf{P}_0, \mathbf{P}_1, \dots) \subset \mathcal{P}$ denote a sequence of label permutations in an SC. After relabeling the simplices by $\bar{\mathcal{P}}$, signal $\mathbf{x}^k$ becomes $\mathbf{P}_k\mathbf{x}^k$, i.e., a reordering of the entries of $\mathbf{x}^k$. Similarly, the incidence matrix $\mathbf{B}_k$ becomes $\bar{\mathbf{B}}_k = \mathbf{P}_{k-1}\mathbf{B}_k\mathbf{P}_k$, i.e., a reordering of the rows and columns of $\mathbf{B}_k$. Likewise, for $\mathbf{L}_k$, we get $\bar{\mathbf{L}}_k = \mathbf{P}_k\mathbf{L}_k\mathbf{P}_k^\top$.

Proposition 2.2 (Permutation equivariance)**.** *Consider the Hodge Laplacians $\mathbf{L}_k$ and $\bar{\mathbf{L}}_k = \mathbf{P}_k\mathbf{L}_k\mathbf{P}^\top$ for a permutation sequence $\bar{\mathcal{P}}$. For the simplicial signals $\mathbf{x}^k$ and $\bar{\mathbf{x}}^k = \mathbf{P}_k\mathbf{x}^k$, with $\bar{\mathbf{H}}_k := \mathbf{H}(\bar{\mathbf{L}}_k)$ [cf. (2.4)], the simplicial filter outputs $\mathbf{x}_{\text{o}}^k := \mathbf{H}_k\mathbf{x}^k$ and $\bar{\mathbf{x}}_{\text{o}}^k := \bar{\mathbf{H}}_k\bar{\mathbf{x}}^k$ satisfy*

$$\bar{\mathbf{x}}_{\text{o}}^k := \bar{\mathbf{H}}_k\bar{\mathbf{x}}^k = \bar{\mathbf{H}}_k(\mathbf{P}_k\mathbf{x}^k) = \mathbf{P}_k\mathbf{H}_k\mathbf{x}^k := \mathbf{P}_k\mathbf{s}_{\text{o}}^k. \tag{2.10}$$

Proof. See Appendix 2.B. □

A new reference orientation of a k-simplex leads to a multiplication by -1 of the columns (or rows) of the incidence matrices $\mathbf{B}_k$ and $\mathbf{B}_{k+1}$ where the k-simplex appears and it also flips the sign of the corresponding simplicial signal. This can be modeled by a diagonal matrix $\mathbf{D}_k$ from the set

$$\mathcal{D} = \{\mathbf{D}_k = \text{diag}(\mathbf{d}_k) : \mathbf{d}_k \in \{\pm 1\}^{N_k}, k \geq 1, \mathbf{d}_0 = \mathbf{1}\}, \tag{2.11}$$

where $\mathbf{d}_0 = \mathbf{1}$, as the orientation of the nodes is trivial [Roddenberry et al., 2021]. Denote a sequence of orientation changes by $\bar{\mathcal{D}} = (\mathbf{D}_0, \mathbf{D}_1, \dots) \subset \mathcal{D}$. A k-simplicial signal $\mathbf{x}^k$ becomes $\mathbf{D}_k\mathbf{x}^k$ after an orientation change by $\bar{\mathcal{D}}$. Accordingly, the incidence matrix $\mathbf{B}_k$ becomes $\bar{\mathbf{B}}_k = \mathbf{D}_{k-1}\mathbf{B}_k\mathbf{D}_k$ and the Hodge Laplacian $\mathbf{L}_k$ becomes $\bar{\mathbf{L}}_k = \mathbf{D}_k\mathbf{L}_k\mathbf{D}_k$.

Proposition 2.3 (Orientation equivariance)**.** *Consider the Hodge Laplacians $\mathbf{L}_k$ and $\bar{\mathbf{L}}_k = \mathbf{D}_k\mathbf{L}_k\mathbf{D}_k$ for a sequence of orientation changes $\bar{\mathcal{D}}$. For the simplicial signals $\mathbf{x}^k$ and $\bar{\mathbf{x}}^k = \mathbf{D}_k\mathbf{x}^k$, with $\bar{\mathbf{H}}_k = \mathbf{H}(\bar{\mathbf{L}}_k)$, the simplicial filter outputs $\mathbf{x}_{\text{o}}^k := \mathbf{H}_k\mathbf{x}^k$ and $\bar{\mathbf{x}}_{\text{o}}^k := \bar{\mathbf{H}}_k\bar{\mathbf{x}}^k$ satisfy*

$$\bar{\mathbf{x}}_{\text{o}}^k := \bar{\mathbf{H}}_k\bar{\mathbf{x}}^k = \bar{\mathbf{H}}_k(\mathbf{D}_k\mathbf{x}^k) = \mathbf{D}_k\mathbf{H}_k\mathbf{x}^k := \mathbf{D}_k\mathbf{s}_{\text{o}}^k. \tag{2.12}$$

Proof. See Appendix 2.C. □

Intuitively, the two previous propositions state that for the simplicial filter $\mathbf{H}_k$ the labeling and reference orientation of simplices are inconsequential for the filter output. These two properties have been previously reported in the context of neural network on SCs [Bodnar et al., 2021b; Roddenberry et al., 2021; Yang et al., 2022a]. These equivariances imply that we can learn a filter to process a given simplex by seeing only permuted and reoriented versions of it: if two parts of an SC are topologically equivalent and the simplices support corresponding flows, a simplicial convolutional filter yields equivalent outputs.

2.4 Spectral Analysis of Simplicial Filters

We now analyze the spectral properties of the simplicial convolutional filter. First, we introduce the Hodge decomposition and review the simplicial Fourier transform (SFT) [Barbarossa & Sardellitti, 2020]. Then, we investigate the simplicial frequency in terms of the Hodge decomposition. By defining three frequency types, we characterize the frequency response of the filter.

2.4.1 Hodge Decomposition

Theorem 2.4. *The Hodge decomposition in the edge space states that:*

$$\mathbb{R}^{N_1} = \mathrm{im}(\mathbf{B}_1^\top) \oplus \mathrm{im}(\mathbf{B}_2) \oplus \ker(\mathbf{L}_1) \tag{2.13}$$

where $\mathrm{im}(\cdot)$ *and* $\ker(\cdot)$ *are the image and kernel of a matrix.*

This implies that the edge space is composed of three orthogonal subspaces, namely, the gradient space $\mathrm{im}(\mathbf{B}_1^\top)$, the curl space $\mathrm{im}(\mathbf{B}_2)$, and the harmonic space $\ker(\mathbf{L}_1)$ [Lim, 2020; Schaub et al., 2021]. Thus, any edge flow $\mathbf{f} \in \mathbb{R}^{N_1}$ can be decomposed into three orthogonal components

$$\mathbf{f} = \mathbf{f}_\mathrm{G} + \mathbf{f}_\mathrm{C} + \mathbf{f}_\mathrm{H}, \tag{2.14}$$

which are the *gradient component* $\mathbf{f}_\mathrm{G} \in \mathrm{im}(\mathbf{B}_1^\top)$, the *curl component* $\mathbf{f}_\mathrm{C} \in \mathrm{im}(\mathbf{B}_2)$, and the *harmonic component* $\mathbf{f}_\mathrm{H} \in \ker(\mathbf{L}_1)$, respectively. Furthermore, the incidence matrices $\mathbf{B}_1$, $\mathbf{B}_2$ and their adjoints can be interpreted as follows [Barbarossa & Sardellitti, 2020; Schaub et al., 2021].

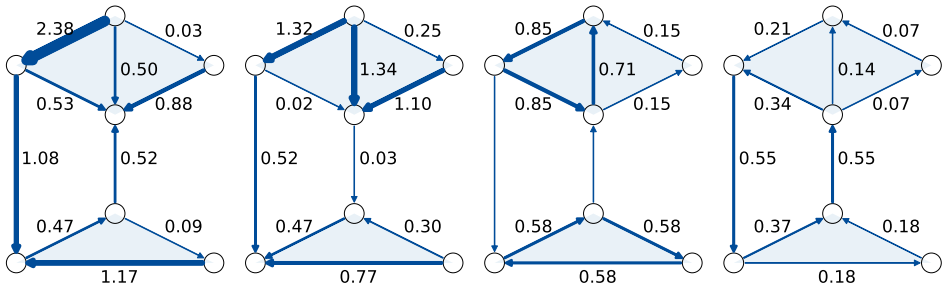
Divergence operator $\mathbf{B}_1$. The incidence matrix $\mathbf{B}_1$ acts as a *divergence operator*. By applying it to an edge flow $\mathbf{f}$, we compute the divergence of the flow, $\mathrm{div}(\mathbf{f}) = \mathbf{B}_1\mathbf{f}$. The ith entry of $\mathrm{div}(\mathbf{f})$ is the netflow passing through the ith vertex, i.e., the difference between the total inflow and outflow at vertex i. This is a nodal variation measure of the edge flow. A vertex is a source or sink if it has a nonzero netflow.

Gradient operator $\mathbf{B}_1^\top$. The adjoint operator $\mathbf{B}_1^\top$ is called the *gradient operator*, which takes the difference between node signals along the oriented edges to induce an edge flow, $\mathbf{f}_\mathrm{G} = \mathbf{B}_1^\top\mathbf{v}$. We call $\mathbf{f}_\mathrm{G} \in \mathrm{im}(\mathbf{B}_1^\top)$ a *gradient flow* and subspace $\mathrm{im}(\mathbf{B}_1^\top)$ the *gradient space*, as any gradient flow can be induced from a node signal via the gradient operator.

Curl adjoint $\mathbf{B}_2$. By applying matrix $\mathbf{B}_2$ to a triangle signal $\mathbf{t} \in \mathbb{R}^{N_2}$, we can induce a *curl flow*, $\mathbf{f}_\mathrm{C} = \mathbf{B}_2\mathbf{t}$, corresponding to a flow locally circling along the edges of triangles. The space $\mathrm{im}(\mathbf{B}_2)$ is the *curl space* as any flow in it can be induced from a triangle signal.

Curl operator $\mathbf{B}_2^\top$. By applying the operator $\mathbf{B}_2^\top$ to an edge flow $\mathbf{f}$, we compute its curl as $\mathrm{curl}(\mathbf{f}) = \mathbf{B}_2^\top\mathbf{f}$, where the ith entry is the netflow circulating along the ith triangle, i.e., the sum of the edge flows forming the triangle. This can be seen as a rotational variation measure of the edge flow.

The two incidence matrices and their adjoints provide insights into the three orthogonal subspaces and signal components given by the Hodge decomposition.



(a) $\mathbf{f}$ (b) $\mathbf{f}_{\mathrm{G}}$ (c) $\mathbf{f}_{\mathrm{C}}$ (d) $\mathbf{f}_{\mathrm{H}}$

Figure 2.4: Flow decomposition illustration (all numbers rounded to two decimal places). (a): A synthetic edge flow $\mathbf{f}$. (b): The gradient component $\mathbf{f}_{\mathrm{G}}$ has a nonzero netflow at each node, but a zero flow around each triangle. (c): The curl component $\mathbf{f}_{\mathrm{C}}$ has a zero netflow at each node, i.e., is divergence-free, but a nonzero flow around each triangle. (d): The harmonic component $\mathbf{f}_{\mathrm{H}}$ has a zero netflow at each node and zero circulation around each triangle, i.e., is divergence- and curl-free (see Section 2.6.1).

i) If edge flow $\mathbf{f}$ has a zero divergence at each vertex, i.e., $\mathbf{B}_1\mathbf{f} = \mathbf{0} \Leftrightarrow \mathbf{f} \in \ker(\mathbf{B}_1)$, then it is cyclic or *divergence-free*. The space $\ker(\mathbf{B}_1)$ is called the *cycle space*, orthogonal to the gradient space, i.e., $\mathbb{R}^{N_1} = \mathrm{im}(\mathbf{B}_1^\top) \oplus \ker(\mathbf{B}_1)$. A gradient flow always has nonzero divergence, while a curl flow is divergence-free due to (2.2).

ii) If edge flow $\mathbf{f}$ has a zero curl on each triangle, it is *curl-free* and $\mathbf{f} \in \ker(\mathbf{B}_2^\top)$. The curl space $\mathrm{im}(\mathbf{B}_2)$ is orthogonal to the space $\ker(\mathbf{B}_2^\top)$ and we have $\mathbb{R}^{N_1} = \mathrm{im}(\mathbf{B}_2) \oplus \ker(\mathbf{B}_2^\top)$. A gradient flow $\mathbf{f}_{\mathrm{G}}$ is curl-free due to (2.2).

iii) The space $\ker(\mathbf{L}_1)$ is called the harmonic space. Any flow $\mathbf{f}_{\mathrm{H}} \in \ker(\mathbf{L}_1)$ satisfies $\mathbf{L}_1\mathbf{f}_{\mathrm{H}} = \mathbf{0}$, which is harmonic, i.e., both divergence- and curl-free.

The decomposition of an edge flow into its three components reveals different properties of the flow, as shown in Fig. 2.4. For instance, we can study the effect of an external source or sink by extracting the gradient component of the edge flow [Barbarossa & Sardellitti, 2020]. In Section 2.6.1, we will discuss this subcomponent extraction problem and solve it with simplicial filters.

2.4.2 Simplicial Fourier Transform

The Hodge Laplacians are positive semidefinite matrices and admit an eigendecomposition

$$\mathbf{L}_k = \mathbf{U}_k \mathbf{\Lambda}_k \mathbf{U}_k^\top, \tag{2.15}$$

where the orthonormal matrix $\mathbf{U}_k = [\mathbf{u}_{k,1}, \ldots, \mathbf{u}_{k,N_k}]$ collects the eigenvectors, and the diagonal matrix $\mathbf{\Lambda}_k = \mathrm{diag}(\lambda_{k,1}, \ldots, \lambda_{k,N_k})$ the associated eigenvalues. There exists a correspondence between $\mathbf{U}_1$ and the three orthogonal subspaces given by the Hodge decomposition (2.13), detailed in the following proposition.

Proposition 2.5. *Given the* 1*-Hodge Laplacian of an SC* $\mathbf{L}_1 = \mathbf{L}_{1,\mathrm{d}} + \mathbf{L}_{1,\mathrm{u}}$*, the following holds.*

1. *Gradient eigenvectors* $\mathbf{U}_{\mathrm{G}} = [\mathbf{u}_{\mathrm{G},1}, \ldots, \mathbf{u}_{\mathrm{G},N_{\mathrm{G}}}] \in \mathbb{R}^{N_1 \times N_{\mathrm{G}}}$ *of* $\mathbf{L}_{1,\mathrm{d}}$ *associated with nonzero eigenvalues span the gradient space* $\mathrm{im}(\mathbf{B}_1^\top)$ *with dimension* N_{G}*, i.e.,* $\mathrm{im}(\mathbf{B}_1^\top) = \mathrm{im}(\mathbf{U}_{\mathrm{G}})$.

2. *Curl eigenvectors* $\mathbf{U}_{\mathrm{C}} = [\mathbf{u}_{\mathrm{C},1}, \ldots, \mathbf{u}_{\mathrm{C},N_{\mathrm{C}}}] \in \mathbb{R}^{N_1 \times N_{\mathrm{C}}}$ *of* $\mathbf{L}_{1,\mathrm{u}}$ *associated with nonzero eigenvalues span the curl space* $\mathrm{im}(\mathbf{B}_2)$ *with dimension* N_{C}*, i.e.,* $\mathrm{im}(\mathbf{B}_2) = \mathrm{im}(\mathbf{U}_{\mathrm{C}})$.

3. *Gradient eigenvectors* $\mathbf{U}_{\mathrm{G}}$ *are orthogonal to curl ones* $\mathbf{U}_{\mathrm{C}}$*. Matrix* $[\mathbf{U}_{\mathrm{G}}\ \mathbf{U}_{\mathrm{C}}]$ *forms the eigenvectors of* $\mathbf{L}_1$ *associated with nonzero eigenvalues, which span the space* $\mathrm{im}(\mathbf{L}_1)$ *with dimension* $N_{\mathrm{G}} + N_{\mathrm{C}}$.

4. *Harmonic eigenvectors* $\mathbf{U}_{\mathrm{H}} = [\mathbf{u}_{\mathrm{H},1}, \ldots, \mathbf{u}_{\mathrm{H},N_{\mathrm{H}}}] \in \mathbb{R}^{N_1 \times N_{\mathrm{H}}}$ *of* $\mathbf{L}_1$ *associated with zero eigenvalues span the harmonic space* $\ker(\mathbf{L}_1)$ *with dimension* N_{H}*, i.e.,* $\ker(\mathbf{L}_1) = \mathrm{im}(\mathbf{U}_{\mathrm{H}})$*. Matrices* $[\mathbf{U}_{\mathrm{H}}\ \mathbf{U}_{\mathrm{C}}]$ *and* $[\mathbf{U}_{\mathrm{H}}\ \mathbf{U}_{\mathrm{G}}]$ *provide the eigenvectors of* $\mathbf{L}_{1,\mathrm{d}}$ *and* $\mathbf{L}_{1,\mathrm{u}}$ *associated with zero eigenvalues, respectively.*

5. *The columns of* $\mathbf{U}_1$ *can be ordered such that* $\mathbf{U}_1 = [\mathbf{U}_{\mathrm{H}}\ \mathbf{U}_{\mathrm{G}}\ \mathbf{U}_{\mathrm{C}}]$*. Matrix* $\mathbf{U}_1$ *forms an eigenvector basis for* $\mathbf{L}_1$*,* $\mathbf{L}_{1,\mathrm{d}}$ *and* $\mathbf{L}_{1,\mathrm{u}}$*, and* $N_1 = N_{\mathrm{H}} + N_{\mathrm{G}} + N_{\mathrm{C}}$.

Proof. See Appendix 2.D. □

Proposition 2.5 shows that:

1. the eigenvectors in $\mathbf{U}_1$ can fully span the three orthogonal subspaces given by the Hodge decomposition;

2. the Hodge Laplacian $\mathbf{L}_1$ and its lower and upper counterparts, $\mathbf{L}_{1,\mathrm{d}}$ and $\mathbf{L}_{1,\mathrm{u}}$, can be simultaneously diagonalized by $\mathbf{U}_1$; and,

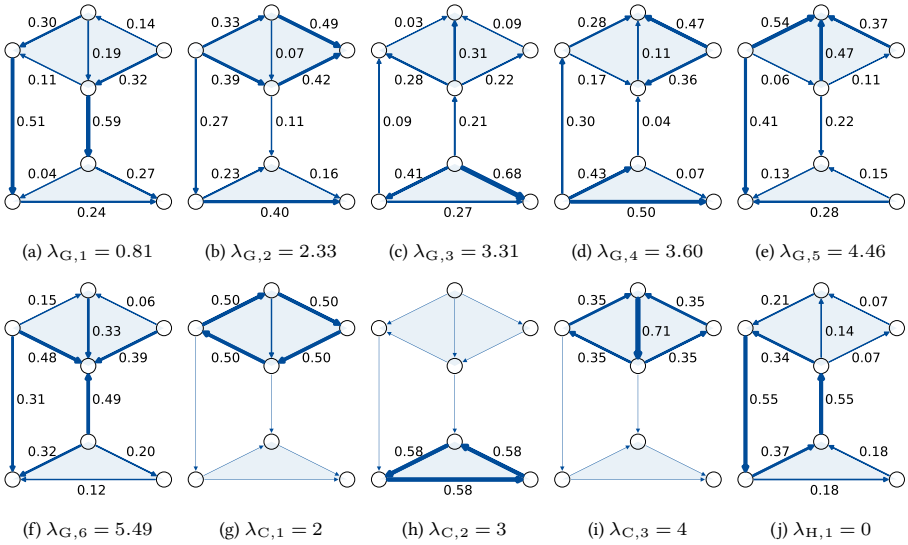
3. from $\mathrm{im}(\mathbf{B}_1^\top) = \mathrm{im}(\mathbf{L}_{1,\mathrm{d}})$, we have that the image of the lower Hodge Laplacian $\mathbf{L}_{1,\mathrm{d}}$ coincides with the gradient space, and from $\mathrm{im}(\mathbf{B}_2) = \mathrm{im}(\mathbf{L}_{1,\mathrm{u}})$, the image of $\mathbf{L}_{1,\mathrm{u}}$ coincides with the curl space.

These results are applicable to the k-Hodge Laplacian accordingly. See Schaub et al. [2021, Thm. 1] and Barbarossa & Sardellitti [2020, Prop. 1] for related discussions.

Thus, the lower shifting of a curl or harmonic flow leads to zero as the harmonic and curl space correspond to the null space of $\mathbf{L}_{1,\mathrm{d}}$. Likewise, the upper shifting of a gradient or harmonic flow leads to zero, i.e.,

$$\mathbf{L}_{1,\mathrm{d}}\mathbf{f}_{\mathrm{C}} = \mathbf{0},\ \mathbf{L}_{1,\mathrm{u}}\mathbf{f}_{\mathrm{G}} = \mathbf{0},\ \mathbf{L}_{1,\mathrm{d}}\mathbf{f}_{\mathrm{H}} = \mathbf{L}_{1,\mathrm{u}}\mathbf{f}_{\mathrm{H}} = \mathbf{0}. \tag{2.16}$$

Given a k-simplicial signal $\mathbf{s}^k$, the *simplicial Fourier transform (SFT)* is given by its projection onto the eigenvectors $\mathbf{U}_k$, i.e., $\tilde{\mathbf{s}}^k := \mathbf{U}_k^\top \mathbf{s}^k$. Entry $[\mathbf{s}^k]_i$ represents the weight eigenvector $\mathbf{u}_{k,i}$ has on expressing $\mathbf{s}^k$. The inverse SFT is given by $\mathbf{s}^k = \mathbf{U}_k \tilde{\mathbf{s}}^k$. For $k = 0$, the SFT coincides with the GFT [Barbarossa & Sardellitti, 2020]. As for the GFT, the eigenvalues of $\mathbf{L}_k$ carry the notion of simplicial frequencies. But in the simplex domain, this frequency



(a) $\lambda_{\mathrm{G},1} = 0.81$ (b) $\lambda_{\mathrm{G},2} = 2.33$ (c) $\lambda_{\mathrm{G},3} = 3.31$ (d) $\lambda_{\mathrm{G},4} = 3.60$ (e) $\lambda_{\mathrm{G},5} = 4.46$

(f) $\lambda_{\mathrm{G},6} = 5.49$ (g) $\lambda_{\mathrm{C},1} = 2$ (h) $\lambda_{\mathrm{C},2} = 3$ (i) $\lambda_{\mathrm{C},3} = 4$ (j) $\lambda_{\mathrm{H},1} = 0$

Figure 2.5: Spectral analysis of the edge space in Fig. 2.1a. The flow value (all numbers rounded to two decimal places) is indicated by the edge width and annotated next to the edge. It is zero if the edge is not annotated. If a flow orientation is opposite to the reference orientation, the corresponding flow value is negative. (a)-(c): The 1st, 3rd, and 6th eigenvectors in the gradient space $\mathbf{U}_{\mathrm{G}}$ with the corresponding gradient frequencies λ_{G}. The total divergence of the eigenvector increases with the eigenvalue. (d)-(e): The 1st and 3rd eigenvectors in the curl space $\mathbf{U}_{\mathrm{C}}$ with the corresponding curl frequencies λ_{C}. The total curl of the eigenvectors increases with the eigenvalue. (f): The only eigenvector in the harmonic space $\mathbf{U}_{\mathrm{H}}$ has frequency 0 and zero divergence and curl.

notion is more involved. As we illustrate in the sequel for $k = 1$, the eigenvalues in the set $\mathcal{Q} = \{\lambda_{1,1}, \ldots, \lambda_{1,N_1}\}$ of $\mathbf{L}_1$ measure three types of simplicial frequencies.

Gradient frequency. For any unit norm gradient eigenvector $\mathbf{u}_{\mathrm{G}} \in \mathbf{U}_{\mathrm{G}}$, associated to the gradient space $\mathrm{im}(\mathbf{B}_1^\top)$, its corresponding eigenvalue follows

$$\lambda_{\mathrm{G}} = \mathbf{u}_{\mathrm{G}}^\top \mathbf{L}_1 \mathbf{u}_{\mathrm{G}} = \|\mathbf{B}_1 \mathbf{u}_{\mathrm{G}}\|_2^2 + \|\mathbf{B}_2^\top \mathbf{u}_{\mathrm{G}}\|_2^2 = \|\mathbf{B}_1 \mathbf{u}_{\mathrm{G}}\|_2^2, \tag{2.17}$$

where the last equality is due to the fact that $\mathbf{u}_{\mathrm{G}}$ is curl-free, i.e., $\mathbf{B}_2^\top \mathbf{u}_{\mathrm{G}} = \mathbf{0}$. Thus, eigenvalue λ_{G} is the squared ℓ_2-norm of the divergence $\mathbf{B}_1 \mathbf{u}_{\mathrm{G}}$ of the eigenvector edge flow $\mathbf{u}_{\mathrm{G}}$. The magnitude of λ_{G} measures the extent of total divergence, i.e., the nodal variation. The gradient eigenvectors associated with a large eigenvalue have a large total divergence. If the SFT $\tilde{\mathbf{f}} = \mathbf{U}_1^\top \mathbf{f}$ of an edge flow has a large weight on such an eigenvector, we say that it contains a high gradient frequency, corresponding to its large divergence. We call any eigenvalue λ_{G} associated to the gradient eigenvectors $\mathbf{U}_{\mathrm{G}}$ a *gradient frequency* and collect them in the set $\mathcal{Q}_{\mathrm{G}} = \{\lambda_{\mathrm{G},1}, \ldots, \lambda_{\mathrm{G},N_{\mathrm{G}}}\}$.

Curl frequency. For any unit norm curl eigenvector $\mathbf{u}_{\mathrm{C}} \in \mathbf{U}_{\mathrm{C}}$ associated to the curl space $\mathrm{im}(\mathbf{B}_2)$, its corresponding eigenvalue follows

$$\lambda_{\mathrm{C}} = \mathbf{u}_{\mathrm{C}}^\top \mathbf{L}_1 \mathbf{u}_{\mathrm{C}} = \|\mathbf{B}_1 \mathbf{u}_{\mathrm{C}}\|_2^2 + \|\mathbf{B}_2^\top \mathbf{u}_{\mathrm{C}}\|_2^2 = \|\mathbf{B}_2^\top \mathbf{u}_{\mathrm{C}}\|_2^2, \tag{2.18}$$

where the last equality is due to the fact that $\mathbf{u}_{\mathrm{C}}$ is divergence-free, i.e., $\mathbf{B}_1\mathbf{u}_{\mathrm{C}} = \mathbf{0}$. Eigenvalue λ_{C} is the squared ℓ_2-norm of the curl $\mathbf{B}_2^\top\mathbf{u}_{\mathrm{C}}$ of the eigenvector $\mathbf{u}_{\mathrm{C}}$. Thus, the magnitude of λ_{C} measures the extent of total curl, i.e., the rotational variation. Any eigenvector in the curl space corresponding to a large eigenvalue has a large total curl. If the SFT of an edge flow contains large weights on such eigenvectors, we say that it has a high curl frequency. We name any eigenvalue λ_{C} associated to the curl eigenvectors $\mathbf{U}_{\mathrm{C}}$ a *curl frequency* and collect them in the set $\mathcal{Q}_{\mathrm{C}} = \{\lambda_{\mathrm{C},1}, \ldots, \lambda_{\mathrm{C},N_{\mathrm{C}}}\}$.

Harmonic frequency. The remaining eigenvalues λ_{H} are associated to the eigenvectors $\mathbf{u}_{\mathrm{H}} \in \mathbf{U}_{\mathrm{H}}$ which span the harmonic space $\ker(\mathbf{L}_1)$. They can be expressed as

$$\lambda_{\mathrm{H}} = \mathbf{u}_{\mathrm{H}}^\top\mathbf{L}_1\mathbf{u}_{\mathrm{H}} = \|\mathbf{B}_1\mathbf{u}_{\mathrm{H}}\|_2^2 + \|\mathbf{B}_2^\top\mathbf{u}_{\mathrm{H}}\|_2^2 = 0, \tag{2.19}$$

because $\mathbf{u}_{\mathrm{H}}$ is harmonic, i.e., both divergence- and curl-free. If the SFT of an edge flow has only nonzeros at the harmonic frequencies (which are all zeros), then it is a harmonic flow. We collect the harmonic frequencies (all zeros) in the set $\mathcal{Q}_{\mathrm{H}} = \{\lambda_{\mathrm{H},1}, \ldots, \lambda_{\mathrm{H},N_{\mathrm{H}}}\}$.

With these three types of simplicial frequencies, low and high frequency notions in an SC are only meaningful with respect to a certain type. A higher gradient (curl) frequency indicates respectively a larger nodal (rotational) variability. This is different from the frequency notion in discrete and graph signal processing. A zero simplicial frequency does not correspond to a constant edge flow but a globally conservative flow, i.e., divergence- and curl-free. Fig. 2.5 shows examples of different eigenvectors and the associated eigenvalues, which would be the analogous of the complex exponentials in discrete-time signal processing for the edge space in Fig. 2.1a.

Simplicial embeddings. From Proposition 2.5 and three types of simplicial frequencies, we can interpret the SFT by the following diagonalization of $\mathbf{L}_1$

$$\mathbf{L}_1 = \mathbf{U}_1\mathrm{blkdiag}\big(\mathbf{\Lambda}_{\mathrm{H}}, \mathbf{\Lambda}_{\mathrm{G}}, \mathbf{\Lambda}_{\mathrm{C}}\big)\mathbf{U}_1^\top \tag{2.20}$$

with $\mathbf{U}_1 = \big[\mathbf{U}_{\mathrm{H}}\ \mathbf{U}_{\mathrm{G}}\ \mathbf{U}_{\mathrm{C}}\big]$ and $\mathrm{blkdiag}(\mathbf{A}, \mathbf{B}, \mathbf{C})$ a block-diagonal matrix containing the square matrices $\mathbf{A}, \mathbf{B}, \mathbf{C}$ as diagonal blocks. Similarly, we have

$$\mathbf{L}_{1,\mathrm{d}} = \mathbf{U}_1\mathrm{blkdiag}\big(\mathbf{0}, \mathbf{\Lambda}_{\mathrm{G}}, \mathbf{0}\big)\mathbf{U}_1^\top \tag{2.21a}$$

$$\mathbf{L}_{1,\mathrm{u}} = \mathbf{U}_1\mathrm{blkdiag}\big(\mathbf{0}, \mathbf{0}, \mathbf{\Lambda}_{\mathrm{C}}\big)\mathbf{U}_1^\top \tag{2.21b}$$

for the lower and upper Laplacians, where $\mathbf{0}$ is an all-zero matrix of appropriate dimensions. Such insightful eigendecompositions enable us to define the following three embeddings of an edge flow $\mathbf{f} \in \mathbb{R}^{N_1}$

$$\begin{cases} \tilde{\mathbf{f}}_{\mathrm{H}} = \mathbf{U}_{\mathrm{H}}^\top\mathbf{f} = \mathbf{U}_{\mathrm{H}}^\top\mathbf{f}_{\mathrm{H}} \in \mathbb{R}^{N_{\mathrm{H}}}, & \text{harmonic embedding} \\ \tilde{\mathbf{f}}_{\mathrm{G}} = \mathbf{U}_{\mathrm{G}}^\top\mathbf{f} = \mathbf{U}_{\mathrm{G}}^\top\mathbf{f}_{\mathrm{G}} \in \mathbb{R}^{N_{\mathrm{G}}}, & \text{gradient embedding} \\ \tilde{\mathbf{f}}_{\mathrm{C}} = \mathbf{U}_{\mathrm{C}}^\top\mathbf{f} = \mathbf{U}_{\mathrm{C}}^\top\mathbf{f}_{\mathrm{C}} \in \mathbb{R}^{N_{\mathrm{C}}}, & \text{curl embedding.} \end{cases} \tag{2.22}$$

They follow from the orthogonality of the three components given by the Hodge decomposition. Equivalently, we can write the SFT of $\mathbf{f}$ as $\tilde{\mathbf{f}} = [\tilde{\mathbf{f}}_{\mathrm{H}}^\top, \tilde{\mathbf{f}}_{\mathrm{G}}^\top, \tilde{\mathbf{f}}_{\mathrm{C}}^\top]^\top$. Each entry of an

embedding represents the weight the flow has on the corresponding eigenvector (simplicial Fourier basis vector), e.g., entry $[\tilde{\mathbf{f}}_{\mathrm{G}}]_i$ is the SFT of $\mathbf{f}$ at the ith gradient frequency $\lambda_{\mathrm{G},i}$. Such an embedding provides a compressed representation of the edge flow if they present a degree of sparsity [Barbarossa & Sardellitti, 2020] and allows us to differentiate different types of edge flow, e.g., to cluster trajectories and analyze ocean drift data [Roddenberry & Segarra, 2019; Schaub et al., 2020].

2.4.3 Filter Frequency Response

Upon defining the SFT, we can analyze the frequency response of the simplicial convolutional filter (2.4). From the diagonalizations (2.21a) and (2.22), we see that the lower simplicial shifting of an edge flow affects only the gradient component and the upper shifting affects only the curl component, i.e.,

$$\mathbf{L}_{1,\mathrm{d}}\mathbf{f} = \mathbf{U}_{\mathrm{G}}\mathbf{\Lambda}_{\mathrm{G}}\tilde{\mathbf{f}}_{\mathrm{G}}, \text{ and } \mathbf{L}_{1,\mathrm{u}}\mathbf{f} = \mathbf{U}_{\mathrm{C}}\mathbf{\Lambda}_{\mathrm{C}}\tilde{\mathbf{f}}_{\mathrm{C}}. \tag{2.23}$$

Through diagonalizing $\mathbf{H}_1$ by $\mathbf{U}_1 = \begin{bmatrix}\mathbf{U}_{\mathrm{H}} & \mathbf{U}_{\mathrm{G}} & \mathbf{U}_{\mathrm{C}}\end{bmatrix}$, we can find the frequency response of $\mathbf{H}_1$ as

$$\tilde{\mathbf{H}}_1 = \mathbf{U}_1^{\top}\mathbf{H}_1\mathbf{U}_1 = \mathrm{blkdiag}\big(\tilde{\mathbf{H}}_{\mathrm{H}}, \tilde{\mathbf{H}}_{\mathrm{G}}, \tilde{\mathbf{H}}_{\mathrm{C}}\big), \tag{2.24}$$

where $\tilde{\mathbf{H}}_{\mathrm{H}} = h_0\mathbf{I}$, $\tilde{\mathbf{H}}_{\mathrm{G}} = h_0\mathbf{I} + \sum_{l_1=1}^{L_1}\alpha_{l_1}\mathbf{\Lambda}_{\mathrm{G}}^{l_1}$ and $\tilde{\mathbf{H}}_{\mathrm{C}} = h_0\mathbf{I} + \sum_{l_2=1}^{L_2}\beta_{l_2}\mathbf{\Lambda}_{\mathrm{C}}^{l_2}$. At an arbitrary frequency λ, the frequency response $\tilde{H}_1(\lambda)$ is given by

$$\begin{cases} \tilde{H}_{\mathrm{H}}(\lambda) := h_0, & \text{for } \lambda \in \mathcal{Q}_{\mathrm{H}}, \\ \tilde{H}_{\mathrm{G}}(\lambda) := h_0 + \sum_{l_1=1}^{L_1}\alpha_{l_1}\lambda^{l_1}, & \text{for } \lambda \in \mathcal{Q}_{\mathrm{G}}, \\ \tilde{H}_{\mathrm{C}}(\lambda) := h_0 + \sum_{l_2=1}^{L_2}\beta_{l_2}\lambda^{l_2}, & \text{for } \lambda \in \mathcal{Q}_{\mathrm{C}}, \end{cases} \tag{2.25}$$

which is the filter frequency response at the harmonic, gradient and curl frequencies, respectively. *By the definition of spectral filtering, the filter $\mathbf{H}_1$ cannot distinguish the signal components belonging to the subspace spanned by the eigenvectors associated to an eigenvalue of multiplicity greater than one. For instance, the filter cannot respond differently to multiple harmonic components, but only scale them by a factor h_0.* In addition, we make the following three observations.

1. Filter $\mathbf{H}_1$ controls the different frequency types independently. The coefficient h_0 determines the harmonic frequency response and contributes to the whole simplicial spectrum. The coefficients $\boldsymbol{\alpha}$ and $\boldsymbol{\beta}$ contribute only to the gradient and curl frequency response, respectively. This independent control on different signal subspaces corresponds to the different parameters imposed on the lower and upper adjacencies in the simplicial domain. In contrast, if setting $L_1 = L_2$ and $\boldsymbol{\alpha} = \boldsymbol{\beta}$ in $\mathbf{H}_1$, the filter cannot regulate the gradient and curl spaces independently and has less flexibility.

2. The gradient frequency response is fully determined by the matrix polynomial in $\mathbf{L}_{1,\mathrm{d}}$ [cf. (2.21a) and (2.24)]. Thus, if $L_2 = 0$, $\mathbf{H}_1$ has as responses h_0 for $\lambda \in \mathcal{Q}_{\mathrm{H}} \cup \mathcal{Q}_{\mathrm{C}}$, and $\tilde{H}_{\mathrm{G}} = h_0 + \sum_{l_1=1}^{L_1}\alpha_{l_1}\lambda^{l_1}$, for $\lambda \in \mathcal{Q}_{\mathrm{G}}$. This controls the gradient and non-gradient

frequencies with a reduced design burden due to fewer parameters. But we give up the control on the curl frequencies. Likewise for the case of $L_1 = 0$, which is beneficial when only curl components need to be tuned.

3. At two *overlapping* frequencies $\lambda_1 = \lambda_2$ with $\lambda_1 \in \mathcal{Q}_{\mathrm{G}}$, and $\lambda_2 \in \mathcal{Q}_{\mathrm{C}}$, filter $\mathbf{H}_1$ responds differently as $\tilde{H}_{\mathrm{G}}(\lambda_1)$ and $\tilde{H}_{\mathrm{C}}(\lambda_2)$ [cf. (2.25)], respectively. This cannot be realized for the filter $\mathbf{H}_1 = \sum_{l=0}^{L} h_l \mathbf{L}_1^l$, i.e., setting $L_1 = L_2$ and $\boldsymbol{\alpha} = \boldsymbol{\beta}$, which follows $\tilde{H}_{\mathrm{G}}(\lambda_1) = \tilde{H}_{\mathrm{C}}(\lambda_2)$ instead. In the latter case, it will preserve the unwanted curl component at λ_2 when setting $\tilde{H}_{\mathrm{G}}(\lambda) = 1$, for $\lambda \in \mathcal{Q}_{\mathrm{G}}$, when the goal is to extract the gradient component.

2.5 Filter Design

Given a training set of input-output edge flow relations $\mathcal{T} = \{(\mathbf{f}_1, \mathbf{f}_{\mathrm{o},1}), \ldots, (\mathbf{f}_{|\mathcal{T}|}, \mathbf{f}_{\mathrm{o},|\mathcal{T}|})\}$, we can learn the filter coefficients in a data-driven fashion by fitting the filtered output $\mathbf{H}_1\mathbf{f}$ to the output $\mathbf{f}_{\mathrm{o}}$. Specifically, consider a mean squared error (MSE) cost function and a regularizer $r(h_0, \boldsymbol{\alpha}, \boldsymbol{\beta})$ to avoid overfitting, we formulate the problem as

$$\min_{h_0, \boldsymbol{\alpha}, \boldsymbol{\beta}} \frac{1}{|\mathcal{T}|} \sum_{(\mathbf{f}_i, \mathbf{f}_{\mathrm{o},i}) \in \mathcal{T}} \|\mathbf{H}_1\mathbf{f}_i - \mathbf{f}_{\mathrm{o},i}\|_2^2 + \gamma r(h_0, \boldsymbol{\alpha}, \boldsymbol{\beta}), \tag{2.26}$$

with $\gamma > 0$. A flow prediction based on (2.26) is detailed in Yang et al. [2021].

In this section, we focus in detail on designing the simplicial filter given a desired frequency response. Specifically, we assume a desired frequency response g_0 at the harmonic frequency $\lambda = 0$, a gradient frequency response $g_{\mathrm{G}}(\lambda)$ for $\lambda \in \mathcal{Q}_{\mathrm{G}}$, and a curl frequency response $g_{\mathrm{C}}(\lambda)$ for $\lambda \in \mathcal{Q}_{\mathrm{C}}$. To design the coefficients $h_0, \boldsymbol{\alpha}, \boldsymbol{\beta}$, our goal is then to approximate the desired response by the filter frequency response $\tilde{H}_1(\lambda)$ [cf. (2.25)], which can be formulated as

$$\begin{cases} h_0 \approx g_0, & \text{for } \lambda_i = 0, \\ h_0 + \sum_{l_1=1}^{L_1} \alpha_{l_1} \lambda_i^{l_1} \approx g_{\mathrm{G}}(\lambda_i), & \text{for } \lambda_i \in \mathcal{Q}_{\mathrm{G}}, \\ h_0 + \sum_{l_2=1}^{L_2} \beta_{l_2} \lambda_i^{l_2} \approx g_{\mathrm{C}}(\lambda_i), & \text{for } \lambda_i \in \mathcal{Q}_{\mathrm{C}}. \end{cases} \tag{2.27}$$

In the following, we first use a standard least-squares (LS) approach to solve (2.27). Later, we consider a universal design to avoid the eigenvalue computation when a continuous desired frequency response is given. In particular, we consider a grid-based and a Chebyshev polynomial approach.

2.5.1 Least-Squares Filter Design

Denote the number of distinct gradient frequencies in $\mathcal{Q}_{\mathrm{G}}$ by D_{G}, and that of distinct curl frequencies in $\mathcal{Q}_{\mathrm{C}}$ by D_{C}. Then, the three sets of equations in (2.27) contain respectively one, D_{G} and D_{C} distinct linear equations. Let $\mathbf{g} = [g_0, \mathbf{g}_{\mathrm{G}}^\top, \mathbf{g}_{\mathrm{C}}^\top]^\top$ collect the desired responses at distinct frequencies where $[\mathbf{g}_{\mathrm{G}}]_i = g_{\mathrm{G}}(\lambda_{\mathrm{G},i})$, for $i = 1 \ldots, D_{\mathrm{G}}$, is the response at the ith distinct gradient frequency $\lambda_{\mathrm{G},i}$, and $[\mathbf{g}_{\mathrm{C}}]_i = g_{\mathrm{C}}(\lambda_{\mathrm{C},i})$, for $i = 1 \ldots, D_{\mathrm{C}}$, at the

ith distinct curl frequency. Then, we can obtain the filter coefficients by solving the LS problem

$$\min_{h_0,\boldsymbol{\alpha},\boldsymbol{\beta}} \left\| \left[\begin{array}{c|cc} \mathbf{1} & \multicolumn{2}{c}{\mathbf{0}} \\ \hline & \boldsymbol{\Phi}_{\mathrm{G}} & \mathbf{0} \\ & \mathbf{0} & \boldsymbol{\Phi}_{\mathrm{C}} \end{array}\right] \begin{bmatrix} h_0 \\ \boldsymbol{\alpha} \\ \boldsymbol{\beta} \end{bmatrix} - \mathbf{g} \right\|_2^2, \tag{2.28}$$

where $\mathbf{1}$ ($\mathbf{0}$) is an all-one (all-zero) matrix or vector of an appropriate dimension, $\boldsymbol{\Phi}_{\mathrm{G}} \in \mathbb{R}^{D_{\mathrm{G}} \times L_1}$ and $\boldsymbol{\Phi}_{\mathrm{C}} \in \mathbb{R}^{D_{\mathrm{C}} \times L_2}$ are Vandermonde matrices with respective entries $[\boldsymbol{\Phi}_{\mathrm{G}}]_{ij} = \lambda_{\mathrm{G},i}^j$ and $[\boldsymbol{\Phi}_{\mathrm{C}}]_{ij} = \lambda_{\mathrm{C},i}^j$. We refer to problem (2.28) as the *LS design*, which can be solved either with a direct solver or with a decoupled solver, studied as follows.

Direct LS design. From the Cayley-Hamilton theorem (see [Horn & Johnson, 2012] and [Sandryhaila & Moura, 2013, Thm. 3]), we know that any analytical function of a matrix can be expressed as a matrix polynomial of degree less than its minimal polynomial degree, which equals the number of distinct eigenvalues for a positive semi-definite matrix. Thus, we can assume the filter orders $L_1 \leq D_{\mathrm{G}}$ and $L_2 \leq D_{\mathrm{C}}$. Under this condition, problem (2.28) admits a unique solution which can be obtained via the pseudo-inverse of the system matrix. Furthermore, when $L_1 = D_{\mathrm{G}}$ and $L_2 = D_{\mathrm{C}}$, $\boldsymbol{\Phi}_{\mathrm{G}}$ and $\boldsymbol{\Phi}_{\mathrm{C}}$ are square and any two rows in them are linearly independent, the solution leads to a zero cost in (2.28). We refer to this pseudo-inverse solution of (2.28) as the *direct LS design* [1].

In addition, given a desired edge operator $\mathbf{G}$, the following proposition states that it can be implemented by filter $\mathbf{H}_1$.

Proposition 2.6. *A desired linear operator $\mathbf{G} \in \mathbb{R}^{N_1 \times N_1}$ can be perfectly implemented by a simplicial filter $\mathbf{H}_1$ if the following three conditions hold true:*

1. *Matrices $\mathbf{G}$ and $\mathbf{L}_1$ are simultaneously diagonalizable, i.e., $\mathbf{U}_1$ forms an eigenvector basis for $\mathbf{G}$. Let $\mathbf{g} = [\mathbf{g}_{\mathrm{H}}^\top, \mathbf{g}_{\mathrm{G}}^\top, \mathbf{g}_{\mathrm{C}}^\top]^\top$ collect the eigenvalues of $\mathbf{G}$.*
2. *If two eigenvalues of $\mathbf{L}_1$ are of the same frequency type and equal, the corresponding eigenvalues of $\mathbf{G}$ are also equal. For $\lambda_{\mathrm{H},i} = \lambda_{\mathrm{H},j} = 0 \in \mathcal{Q}_{\mathrm{H}}$, it holds that $[\mathbf{g}_{\mathrm{H}}]_i = [\mathbf{g}_{\mathrm{H}}]_j$, for $\lambda_{\mathrm{G},i} = \lambda_{\mathrm{G},j} \in \mathcal{Q}_{\mathrm{G}}$, $[\mathbf{g}_{\mathrm{G}}]_i = [\mathbf{g}_{\mathrm{G}}]_j$, and for $\lambda_{\mathrm{C},i} = \lambda_{\mathrm{C},j} \in \mathcal{Q}_{\mathrm{C}}$, $[\mathbf{g}_{\mathrm{C}}]_i = [\mathbf{g}_{\mathrm{C}}]_j$.*
3. *The filter orders of $\mathbf{H}_1$ fulfill $L_1 \geq D_{\mathrm{G}}$, $L_2 \geq D_{\mathrm{C}}$.*

Proof. See Appendix 2.E. □

Decoupled LS design. We can reduce the complexity of solving (2.28) by decoupling the cost function for different frequency types. First, we rewrite problem (2.28) as

$$\min_{h_0,\boldsymbol{\alpha},\boldsymbol{\beta}} \left\| \begin{bmatrix} \mathbf{1} & \boldsymbol{\Phi}_{\mathrm{G}} \end{bmatrix} \begin{bmatrix} h_0 \\ \boldsymbol{\alpha} \end{bmatrix} - \mathbf{g}_{\mathrm{G}} \right\|_2^2 + \left\| \begin{bmatrix} \mathbf{1} & \boldsymbol{\Phi}_{\mathrm{C}} \end{bmatrix} \begin{bmatrix} h_0 \boldsymbol{\beta} \end{bmatrix} - \mathbf{g}_{\mathrm{C}} \right\|_2^2 + \| h_0 - g_0 \|_2^2 . \tag{2.29}$$

To approximate a solution, we ignore the dependence of the first two terms in (2.29) on coefficient h_0 and solve the last term separately to estimate h_0. We then substitute the

[1] Note that in the special case of $L_1 = L_2$ with $\boldsymbol{\alpha} = \boldsymbol{\beta}$ or when $L_1 = 0$ or $L_2 = 0$, the spectral design is equivalent to that of a graph filter [Ortega et al., 2018; Sandryhaila & Moura, 2014].

estimate $\hat{h}_0$ in the first two terms to obtain $\hat{\boldsymbol{\alpha}}$ and $\hat{\boldsymbol{\beta}}$, given by

$$\begin{cases} \hat{h}_0 = g_0, \\ \hat{\boldsymbol{\alpha}} = \boldsymbol{\Phi}_{\mathrm{G}}^{\dagger}(\mathbf{g}_{\mathrm{G}} - g_0\mathbf{1}), \\ \hat{\boldsymbol{\beta}} = \boldsymbol{\Phi}_{\mathrm{C}}^{\dagger}(\mathbf{g}_{\mathrm{C}} - g_0\mathbf{1}), \end{cases} \tag{2.30}$$

which, referred to as the *decoupled LS design*, is suboptimal compared to the direct LS design. The following proposition discusses this suboptimality.

Proposition 2.7. *The decoupled LS design (2.30) converges to the direct solution of (2.28) as* $\|\boldsymbol{\Phi}_{\mathrm{G}}\boldsymbol{\Phi}_{\mathrm{G}}^{\dagger} - \mathbf{I}\|_F \to 0$ *and* $\|\boldsymbol{\Phi}_{\mathrm{C}}\boldsymbol{\Phi}_{\mathrm{C}}^{\dagger} - \mathbf{I}\|_F \to 0$ *where* $\|\cdot\|_F$ *is the Frobenius norm.*

Proof. See Appendix 2.F. □

Proposition 2.7 states that as the pseudo-inverses of $\boldsymbol{\Phi}_{\mathrm{G}}$ and $\boldsymbol{\Phi}_{\mathrm{C}}$ become closer to the true inverses, the decoupled solution converges to the direct solution. Moreover, the decoupled LS design reduces the computational cost to $\mathcal{O}(D_{\mathrm{G}}^3 + D_{\mathrm{C}}^3)$ from $\mathcal{O}((1 + D_{\mathrm{G}} + D_{\mathrm{C}})^3)$ for the direct one. However, both designs require the computation of the eigenvalues of the Hodge Laplacians, which takes in general a computational complexity of $\mathcal{O}(N_1^3)$ [Watkins, 2007]. In the sequel, we consider a universal design strategy to avoid the eigenvalue computation, specifically, a grid-based design and a Chebyshev polynomial design [Shuman et al., 2018].

2.5.2 Grid-Based Filter Design

The grid-based filter design aims to match the desired frequency response in a continuous interval where the exact frequencies lie such that the eigenvalue computation of $\mathbf{L}_1$ can be avoided. Given a harmonic frequency response g_0, a continuous gradient frequency response $g_{\mathrm{G}}(\lambda), \lambda \in [\lambda_{\mathrm{G,min}}, \lambda_{\mathrm{G,max}}]$ and a continuous curl frequency response $g_{\mathrm{C}}(\lambda), \lambda \in [\lambda_{\mathrm{C,min}}, \lambda_{\mathrm{C,max}}]$, we want that

$$\begin{cases} h_0 - g_0 \approx 0 \\ \int_{\lambda_{\mathrm{G,min}}}^{\lambda_{\mathrm{G,max}}} \left| h_0 + \sum_{l_1=1}^{L_1} \alpha_{l_1} \lambda^{l_1} - g_{\mathrm{G}}(\lambda) \right|^2 \mathrm{d}\lambda \approx 0 \\ \int_{\lambda_{\mathrm{C,min}}}^{\lambda_{\mathrm{C,max}}} \left| h_0 + \sum_{l_2=1}^{L_2} \beta_{l_2} \lambda^{l_2} - g_{\mathrm{C}}(\lambda) \right|^2 \mathrm{d}\lambda \approx 0, \end{cases} \tag{2.31}$$

which is a continuous version of (2.27). An example of the continuous frequency response is given in Fig. 2.6 (top).

By sampling M_1 and M_2 (grid-)points uniformly from the intervals $[\lambda_{\mathrm{G,min}}, \lambda_{\mathrm{G,max}}]$ and $[\lambda_{\mathrm{C,min}}, \lambda_{\mathrm{C,max}}]$, the problem (2.31) can then be formulated as an LS problem of form (2.28) but with the sampled frequencies as the entries of the system matrix instead of the true eigenvalues. We can again solve this LS problem either via a direct pseudo-inverse of the system matrix or via the decoupled solution method [cf.(2.30)]. Notice that the largest true eigenvalue can be approximated by efficient algorithms, e.g., power iteration, [Sleijpen & Van der Vorst, 2000; Watkins, 2007]. For the smallest, we can set a small value greater than 0 as the lower bound.

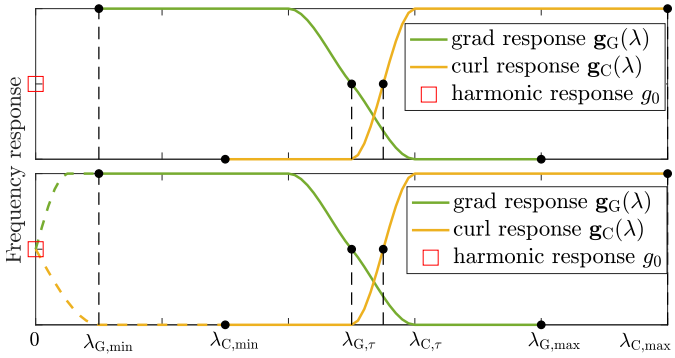


Figure 2.6: An example of a continuous frequency response for the grid-based (top) and Chebyshev (bottom) filter design, which promotes low gradient frequencies and the high curl frequencies. The cut-off frequencies are subscripted by τ. The harmonic frequency response is given at frequency 0. For the Chebyshev polynomial design, we require that the gradient and curl are continuous functions starting from frequency 0 and $g_{\mathrm{G}}(0) = g_{\mathrm{C}}(0) = g_0$.

2.5.3 Chebyshev Polynomial Filter Design

Both discussed filter designs rely on solving an LS problem, which has a Vandermonde matrix as the system matrix, and suffers from numerical instability. To tackle this issue, we consider a Chebyshev polynomial based filter design [Druskin & Knizhnerman, 1989; Shuman et al., 2018]. As illustrated in Fig. 2.6(Bottom), consider a continuous gradient frequency response $g_{\mathrm{G}}(\lambda), \lambda \in [0, \lambda_{\mathrm{G,max}}]$ and a continuous curl frequency response $g_{\mathrm{C}}(\lambda), \lambda \in [0, \lambda_{\mathrm{C,max}}]$. We further require that $g_{\mathrm{G}}(0) = g_{\mathrm{C}}(0) = g_0$. That is, the continuous functions $g_{\mathrm{G}}(\lambda)$ and $g_{\mathrm{C}}(\lambda)$ are defined starting from frequency 0, at which they are equal to the harmonic frequency response g_0.

As the filter $\mathbf{H}_1$ is a sum of matrix polynomials of $\mathbf{L}_{1,\mathrm{d}}$ and $\mathbf{L}_{1,\mathrm{u}}$, our strategy is to first consider the Chebyshev polynomial design for each of them so to separately obtain the gradient and curl frequency responses, then sum these two polynomials to obtain the final filter. However, one type of frequency response could be affected unwantedly by the identity matrix term in the Chebyshev polynomial designed for the other frequency response, as detailed later. The requirement that $g_{\mathrm{G}}(0) = g_{\mathrm{C}}(0) = g_0$ allows a possible correction.

First, we approximate the gradient frequency response via a truncated series of shifted Chebyshev polynomials $\mathbf{H}_{\mathrm{d}} := \mathbf{H}_{\mathrm{d}}(\mathbf{L}_{1,\mathrm{d}})$. Let $\bar{P}_l(\lambda), \lambda \in [-1, 1]$ be the lth Chebyshev polynomial of the first kind [Mason & Handscomb, 2002]. We perform a transformation $P_l(\lambda) := \bar{P}_l(\frac{\lambda-\omega}{\omega})$ with $\omega := \frac{\lambda_{\mathrm{G,max}}}{2}$ to shift the domain to $[0, \lambda_{\mathrm{G,max}}]$. We then approximate the operator $g_{\mathrm{G}}(\mathbf{L}_{1,\mathrm{d}})$ that has the gradient frequency response $g_{\mathrm{G}}(\lambda)$ by $\mathbf{H}_{\mathrm{d}}$ of order L_1

$$\mathbf{H}_{\mathrm{d}} = \tfrac{1}{2} c_{\mathrm{d},0} \mathbf{I} + \textstyle\sum_{l_1=1}^{L_1} c_{\mathrm{d},l_1} P_{l_1}(\mathbf{L}_{1,\mathrm{d}}) \tag{2.32}$$

where we have $P_0(\mathbf{L}_{1,\mathrm{d}}) = \mathbf{I}$, $P_1(\mathbf{L}_{1,\mathrm{d}}) = \frac{2}{\lambda_{\mathrm{G,max}}} \mathbf{L}_{1,\mathrm{d}} - \mathbf{I}$, the l_1th Chebyshev term, for

$l_1 \geq 2$, is

$$P_{l_1}(\mathbf{L}_{1,\mathrm{d}}) = 2P_1(\mathbf{L}_{1,\mathrm{d}})P_{l_1-1}(\mathbf{L}_{1,\mathrm{d}}) - P_{l_1-2}(\mathbf{L}_{1,\mathrm{d}}), \tag{2.33}$$

and the Chebyshev coefficients c_{d,l_1} can be computed as

$$c_{\mathrm{d},l_1} = \frac{2}{\pi}\int_0^{\pi} \cos(l_1\phi) g_{\mathrm{G}}\big(\omega(\cos\phi+1)\big)\mathrm{d}\phi. \tag{2.34}$$

The frequency response $\tilde{H}_{\mathrm{d}}(\lambda)$ of $\mathbf{H}_{\mathrm{d}}$ can be found as

$$\begin{cases} p_{\mathrm{d},0} := \frac{1}{2}c_{\mathrm{d},0} + \sum_{l_1=1}^{\lfloor L_1/2 \rfloor}(c_{\mathrm{d},2l_1} - c_{\mathrm{d},2l_1-1}), \text{for } \lambda \in \mathcal{Q}_{\mathrm{H}} \cup \mathcal{Q}_{\mathrm{C}} \\ \tilde{H}_{\mathrm{d,G}}(\lambda) := \frac{1}{2}c_{\mathrm{d},0} + \sum_{l_1=1}^{L_1} c_{\mathrm{d},l_1} P_{l_1}(\lambda), \text{for } \lambda \in \mathcal{Q}_{\mathrm{G}}, \end{cases} \tag{2.35}$$

with coefficient $p_{\mathrm{d},0}$ on the identity term of $\mathbf{H}_{\mathrm{d}}$, which is the frequency response at the harmonic and curl frequencies, associated to the kernel of $\mathbf{L}_{1,\mathrm{d}}$ [cf. (2.21a)]. For a reasonably large L_1 we have $p_{\mathrm{d},0} \approx g_0$ and $\tilde{H}_{\mathrm{d,G}}(\lambda) \approx g_{\mathrm{G}}(\lambda), \lambda \in \mathcal{Q}_{\mathrm{G}}$.

Second, to approximate the curl frequency response $g_{\mathrm{C}}(\lambda)$, we follow the same procedure [cf. (2.32)-(2.34)] to obtain the Chebyshev polynomial $\mathbf{H}_{\mathrm{u}} := \mathbf{H}_{\mathrm{u}}(\mathbf{L}_{1,\mathrm{u}})$ of order L_2

$$\mathbf{H}_{\mathrm{u}} = \tfrac{1}{2}c_{\mathrm{u},0}\mathbf{I} + \sum_{l_2=1}^{L_2} c_{\mathrm{u},l_2} P_{l_2}(\mathbf{L}_{1,\mathrm{u}}). \tag{2.36}$$

It has a frequency response $\tilde{H}_{\mathrm{u}}(\lambda)$

$$\begin{cases} p_{\mathrm{u},0} := \frac{1}{2}c_{\mathrm{u},0} + \sum_{l_2=1}^{\lfloor L_2/2 \rfloor}(c_{\mathrm{u},2l_2} - c_{\mathrm{u},2l_2-1}), \text{for } \lambda \in \mathcal{Q}_{\mathrm{H}} \cup \mathcal{Q}_{\mathrm{G}} \\ \tilde{H}_{\mathrm{u,C}}(\lambda) := \frac{1}{2}c_{\mathrm{u},0} + \sum_{l_2=1}^{L_2} c_{\mathrm{u},l_2} P_{l_2}(\lambda), \text{for } \lambda \in \mathcal{Q}_{\mathrm{C}}. \end{cases} \tag{2.37}$$

with coefficient $p_{\mathrm{u},0}$ on the identity term of $\mathbf{H}_{\mathrm{u}}$, which is the frequency response at the harmonic and gradient frequencies, associated to the kernel of $\mathbf{L}_{1,\mathrm{u}}$ [cf. (2.21b)]. For a reasonably large L_2 we have $p_{\mathrm{u},0} \approx g_0$ and $\tilde{H}_{\mathrm{u,C}}(\lambda) \approx g_{\mathrm{C}}(\lambda), \lambda \in \mathcal{Q}_{\mathrm{C}}$.

Lastly, by summing $\mathbf{H}_{\mathrm{d}}$ and $\mathbf{H}_{\mathrm{u}}$, we obtain a filter that approximates the gradient and curl frequency responses. However, from (2.35), we see that $\mathbf{H}_{\mathrm{d}}$ generates a response $p_{\mathrm{d},0}$ at both harmonic and curl frequencies. This will lift up the curl frequency response unwantedly by $p_{\mathrm{d},0}$. Similarly, $\mathbf{H}_{\mathrm{u}}$ has the effect of lifting the gradient frequency response by $p_{\mathrm{u},0}$ [cf. (2.37)]. By requiring that $g_{\mathrm{G}}(0) = g_{\mathrm{C}}(0) = g_0$, we can remove this unwanted influence by subtracting a term $g_0\mathbf{I}$ from the summation. Hence, the final Chebyshev polynomial design $\mathbf{H}_1$ of orders L_1 and L_2 is given by

$$\mathbf{H}_1 = \mathbf{H}_{\mathrm{d}} + \mathbf{H}_{\mathrm{u}} - g_0\mathbf{I}, \tag{2.38}$$

which has a frequency response $\tilde{H}_1(\lambda)$

$$\begin{cases} p_{\mathrm{d},0} + p_{\mathrm{u},0} - g_0, & \text{for } \lambda \in \mathcal{Q}_{\mathrm{H}} \\ \tilde{H}_{\mathrm{l,G}}(\lambda) + p_{\mathrm{u},0} - g_0, & \text{for } \lambda \in \mathcal{Q}_{\mathrm{G}} \\ \tilde{H}_{\mathrm{u,C}}(\lambda) + p_{\mathrm{d},0} - g_0, & \text{for } \lambda \in \mathcal{Q}_{\mathrm{C}}. \end{cases} \tag{2.39}$$

The following proposition states that the approximation error of the Chebyshev polynomial design (2.38) is bounded.

Proposition 2.8. *Let* $\mathbf{G}$ *be the desired operator corresponding to the continuous gradient and curl frequency responses* $g_{\mathrm{G}}(\lambda)$ *with* $\lambda \in [0, \lambda_{\mathrm{G,max}}]$ *and* $g_{\mathrm{C}}(\lambda)$ *with* $\lambda \in [0, \lambda_{\mathrm{C,max}}]$*, as well as the harmonic one* $g_{\mathrm{G}}(0) = g_{\mathrm{C}}(0) = g_0$*. Let* $\mathbf{H}_1$ *[cf. (2.38)] be a truncated series of Chebyshev polynomials of orders* L_1 *and* L_2 *with frequency response* $\tilde{H}_1(\lambda)$ *[cf. (2.39)]. Define*

$$\begin{aligned} B_1(L_1) &:= \sup_{\lambda \in [0, \lambda_{\mathrm{G,max}}]} \left\{ \left| \tilde{H}_{\mathrm{d}}(\lambda) - g_0 - g_{\mathrm{G}}(\lambda) \right| \right\}, \\ B_2(L_2) &:= \sup_{\lambda \in [0, \lambda_{\mathrm{C,max}}]} \left\{ \left| \tilde{H}_{\mathrm{u}}(\lambda) - g_0 - g_{\mathrm{C}}(\lambda) \right| \right\}, \end{aligned} \tag{2.40}$$

and $B := \max \left\{ B_1(L_1), B_2(L_2) \right\}$*. Then, we have that*

$$\|\mathbf{G} - \mathbf{H}_1\|_2 := \max_{\mathbf{f} \neq \mathbf{0}} \frac{\|(\mathbf{G} - \mathbf{H}_1)\mathbf{f}\|_2}{\|\mathbf{f}\|_2} \leq B \tag{2.41}$$

Proof. See Appendix 2.G. □

When $g_{\mathrm{G}}(\cdot)$ and $g_{\mathrm{C}}(\cdot)$ are real analytic, a stronger bound can be found [Hammond et al., 2011; Shuman et al., 2018]. In addition, we make the following three comments.

1. When only the gradient frequency response $g_{\mathrm{G}}(\lambda)$ is of interest, we can directly consider the Chebyshev polynomial $\mathbf{H}_{\mathrm{d}}$ [cf. (2.32)] with $L_2 = 0$. Likewise we consider $\mathbf{H}_{\mathrm{u}}$ when only $g_{\mathrm{C}}(\lambda)$ is of interest.

2. If a frequency response $g(\lambda)$ is given on the whole spectrum, we can build a filter $\mathbf{H} = \sum_{l=0}^{L} h_l \mathbf{L}_1^l$ based on a Chebyshev polynomial design analogous to the graph filter case [Shuman et al., 2018].

3. The Chebyshev polynomial design requires no eigenvalue computation of $\mathbf{L}_1$. Thus, it does not suffer from numerical instability, allows to build simplicial filters with large L_1 and L_2 for an accurate design, and admits a recursive and distributed implementation due to the Chebyshev polynomial property (2.33).

2.6 Applications

In this section, we first discuss how to use a simplicial filter for subcomponent extraction and edge flow denoising. We then consider analyses of financial markets, street and traffic networks. These are similar to previous works [Barbarossa & Sardellitti, 2020; Jiang et al., 2011; Schaub & Segarra, 2018; Schaub et al., 2014; Youn et al., 2008], but we directly use our developed filters instead of employing a regularized optimization problem (which implicitly defines a low-pass filter).

To gauge the performance in estimation tasks, we use the normalized root mean square error (NRMSE), $e = \|\hat{\mathbf{f}} - \mathbf{f}_0\|_2 / \|\mathbf{f}_0\|_2$ with the flow estimate $\hat{\mathbf{f}}$ and the true flow $\mathbf{f}_0$. For filter design problems, we evaluate the spectral norm $\|\mathbf{H}_1 - \mathbf{G}\|_2$ with the designed filter $\mathbf{H}_1$ and the true operator $\mathbf{G}$. The Chebfun toolbox was used for the Chebyshev polynomial filter design [Driscoll et al., 2014].

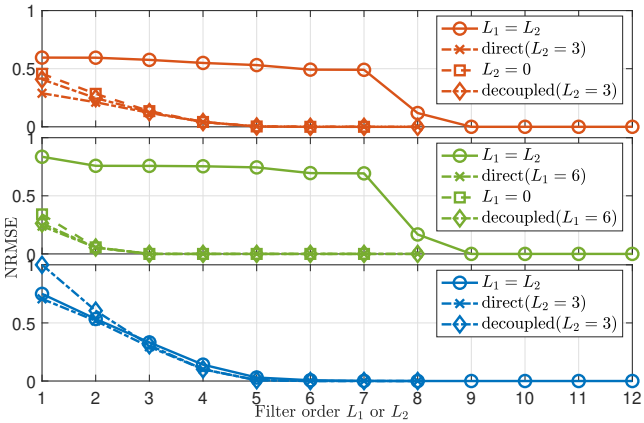


Figure 2.7: Subcomponent extraction performance by filters $\mathbf{H}_1$ with different parameters based on an LS design. The extraction becomes better as the filter order increases. For the gradient and curl components, setting $L_1 = L_2$ and $\boldsymbol{\alpha} = \boldsymbol{\beta}$ worsens the performance because of the limited expressive power. For the general filter form, the direct and decoupled LS designs have smaller performance difference as the filter order grows (see Proposition 2.7).

2.6.1 Subcomponent Extraction

In pairwise ranking problems, we aim to rank alternatives by comparing their scores. The work [Jiang et al., 2011] modeled the score differences between alternatives as edge flows on a pairwise comparison graph. The gradient component of these flows gives a global ranking and the curl components measures the inconsistency of the ranking. For further examples, see Candogan et al. [2011]; Gebhart et al. [2021]; Mock & Volic [2021]. A common approach to obtain the three components of the flow is to compute [Barbarossa & Sardellitti, 2020; Jiang et al., 2011; Lim, 2020]:

$$\hat{\mathbf{f}}_{\mathrm{G}} = \mathbf{P}_{\mathrm{G}}\mathbf{f}, \quad \hat{\mathbf{f}}_{\mathrm{C}} = \mathbf{P}_{\mathrm{C}}\mathbf{f}, \quad \hat{\mathbf{f}}_{\mathrm{H}} = \mathbf{P}_{\mathrm{H}}\mathbf{f} = \mathbf{f} - \hat{\mathbf{f}}_{\mathrm{G}} - \hat{\mathbf{f}}_{\mathrm{C}}. \tag{2.42}$$

where $\mathbf{P}_{\mathrm{G}} = \mathbf{B}_1^\top(\mathbf{B}_1\mathbf{B}_1^\top)^\dagger\mathbf{B}_1$ is the projection onto the gradient space, the curl projector is $\mathbf{P}_{\mathrm{C}} = \mathbf{B}_2(\mathbf{B}_2^\top\mathbf{B}_2)^\dagger\mathbf{B}_2^\top$ and the harmonic projector $\mathbf{P}_{\mathrm{H}} = \mathbf{I} - \mathbf{L}_1\mathbf{L}_1^\dagger$. Notably, we can use a (polynomial) simplicial filter $\mathbf{H}_1$ to implement these operators, too.

Lemma 2.9. *The projection operators (2.42) are equivalent to* $\mathbf{P}_{\mathrm{G}} = \mathbf{U}_{\mathrm{G}}\mathbf{U}_{\mathrm{G}}^\top$, $\mathbf{P}_{\mathrm{G}} = \mathbf{U}_{\mathrm{C}}\mathbf{U}_{\mathrm{C}}^\top$ *and* $\mathbf{P}_{\mathrm{H}} = \mathbf{U}_{\mathrm{H}}\mathbf{U}_{\mathrm{H}}^\top$. *As* $\mathbf{U}$ *requires the knowledge of global properties in this form the projections cannot be computed in a distributed way. However, for* $L_1 = D_{\mathrm{G}}$, $L_2 = D_{\mathrm{C}}$, *there exists a unique* $\{h_0, \boldsymbol{\alpha}, \boldsymbol{\beta}\}$ *such that* $\mathbf{H}_1 = \mathbf{P}_{\mathrm{G}}$. *These coefficients can be found by solving the system (2.28) with* $g_0 = 0, \mathbf{g}_{\mathrm{C}} = \mathbf{0}$ *and* $\mathbf{g}_{\mathrm{G}} = \mathbf{1}$; *analogous arguments lead to distributed implementations of* $\mathbf{H}_1 = \mathbf{P}_{\mathrm{C}}$ *and* $\mathbf{H}_1 = \mathbf{P}_{\mathrm{H}}$.

Proof. See Appendix 2.H. □

For the gradient and curl components, this can be simplified.

Corollary 2.10. *For the gradient projector* $\mathbf{P}_{\mathrm{G}}$*, there exist a filter with* $L_1 = D_{\mathrm{G}}, L_2 = 0$ *and a unique* $\{h_0, \boldsymbol{\alpha}\}$ *such that* $\mathbf{H}_1 = \mathbf{P}_{\mathrm{G}}$*. The solution is* $h_0 = 0$ *and* $\boldsymbol{\alpha} = \boldsymbol{\Phi}_{\mathrm{G}}^{-1}\mathbf{1}$*. For the curl projector* $\mathbf{P}_{\mathrm{C}}$*, there exist a filter with* $L_1 = 0, L_2 = D_{\mathrm{C}}$ *and a unique* $\{h_0, \boldsymbol{\beta}\}$ *such that* $\mathbf{H}_1 = \mathbf{P}_{\mathrm{C}}$*. The solution is* $h_0 = 0$ *and* $\boldsymbol{\beta} = \boldsymbol{\Phi}_{\mathrm{C}}^{-1}\mathbf{1}$

Proof. See Appendix 2.I. □

Corollary 2.10 shows a gradient projector can be build solely upon $\mathbf{L}_{1,\mathrm{d}}$, since $\mathbf{L}_{1,\mathrm{u}}$ has no effect in the gradient space [cf. (2.16) and (2.23)]. Similar arguments hold for the curl projector.

Fig. 2.7 reports the performance of the subcomponent extraction based on $\mathbf{H}_1$ via an LS design. We generated a synthetic edge flow $\mathbf{f} = \mathbf{U}_1\tilde{\mathbf{f}}$ with a flat spectrum $\tilde{\mathbf{f}} = \mathbf{1}$ on the SC in Fig. 2.1a. We observe that as the filter order increases, the filter performs better as its expressive power increases. Setting $L_1 = L_2$ with $\boldsymbol{\alpha} = \boldsymbol{\beta}$ reduces the expressive power as expected. If filter orders obey $L_1 \geq 6$ and $L_3 \geq 3$, then the gradient (or curl) component can be perfectly extracted, as shown in Fig. 2.4 and indicated by Lemma 2.9 and Corollary 2.10. In the general parameter setting, the decoupled LS design performs closer to the direct LS design as the filter order grows as shown in Proposition 2.7.

2.6.2 Edge Flow Denoising

Consider a noisy edge flow $\mathbf{f} = \mathbf{f}^0 + \boldsymbol{\epsilon}$ with $\mathbf{f}^0$ the true edge flow and $\boldsymbol{\epsilon}$ a zero-mean white Gaussian noise. To obtain an estimate $\hat{\mathbf{f}}$, we can solve the regularized optimization problem [Schaub & Segarra, 2018; Schaub et al., 2021]:

$$\min_{\hat{\mathbf{f}}} \|\hat{\mathbf{f}} - \mathbf{f}\|_2^2 + \mu\, \hat{\mathbf{f}}^\top \mathbf{P}\hat{\mathbf{f}}, \tag{2.43}$$

with an optimal solution $\hat{\mathbf{f}} = \mathbf{H}_P\hat{\mathbf{f}} := (\mathbf{I} + \mu\mathbf{P})^{-1}\mathbf{f}$. For matrix $\mathbf{P}$ we have two choices:

1. the edge Laplacian $\mathbf{L}_{1,\mathrm{d}} = \mathbf{B}_1^\top\mathbf{B}_1$, leading to a regularizer $\|\mathbf{B}_1\hat{\mathbf{f}}\|_2^2$ to penalize the flows with a nonzero divergence [Schaub & Segarra, 2018];
2. the Hodge Laplacian $\mathbf{L}_1$, leading to a regularizer $\hat{\mathbf{f}}^\top\mathbf{L}_1\hat{\mathbf{f}} = \|\mathbf{B}_1\hat{\mathbf{f}}\|_2^2 + \|\mathbf{B}_2^\top\hat{\mathbf{f}}\|_2^2$ to penalize the flows with a nonzero divergence or curl [Schaub et al., 2021].

Operator $\mathbf{H}_P$ has the frequency response $\tilde{H}_P(\lambda_i) = 1/(1+\mu\lambda_i)$ with (i) $\lambda_i = 0$ or $\lambda_i \in \mathcal{Q}_{\mathrm{G}}$ for $\mathbf{P} = \mathbf{L}_{1,\mathrm{d}}$, and (ii), $\lambda_i = 0$ or $\lambda_i \in \mathcal{Q}_{\mathrm{G}} \cup \mathcal{Q}_{\mathrm{C}}$ for $\mathbf{P} = \mathbf{L}_1$. Thus, $\mathbf{H}_P$ is a low pass filter which suppresses either the gradient frequencies or the non-harmonic frequencies. We can implement a simplicial filter to approximate $\mathbf{H}_P$. Fig. 2.8 shows the frequency responses of the filter $\mathbf{H}_P$ with $\mathbf{P} = \mathbf{L}_1$, $\mu = 0.5$ based on the grid-based design, for which we considered 10 samples in the frequency interval $[0, 5.488]$ and the maximal eigenvalue is estimated with 50 steps of power iterations. The grid-based filter design errors compared to using the true eigenvalues are negligible, 0.023 and 0.004 for $L = 2$ and $L = 4$, respectively.

Note that the above optimization framework relies on the assumption that the true edge flow is either divergence-free or harmonic, which is not always true for real-world flows

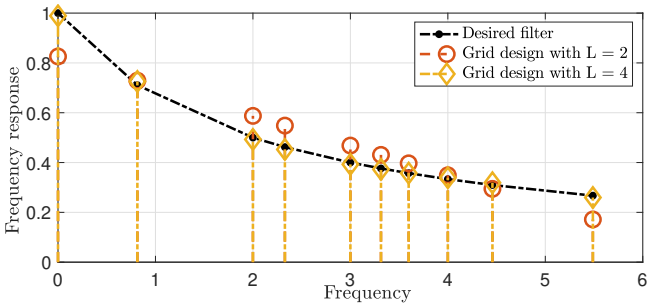


Figure 2.8: Frequency responses of the denoising filter $\mathbf{H}_P = (\mathbf{I} + 0.5\mathbf{L}_1)^{-1}$ based on the grid design with different filter orders.

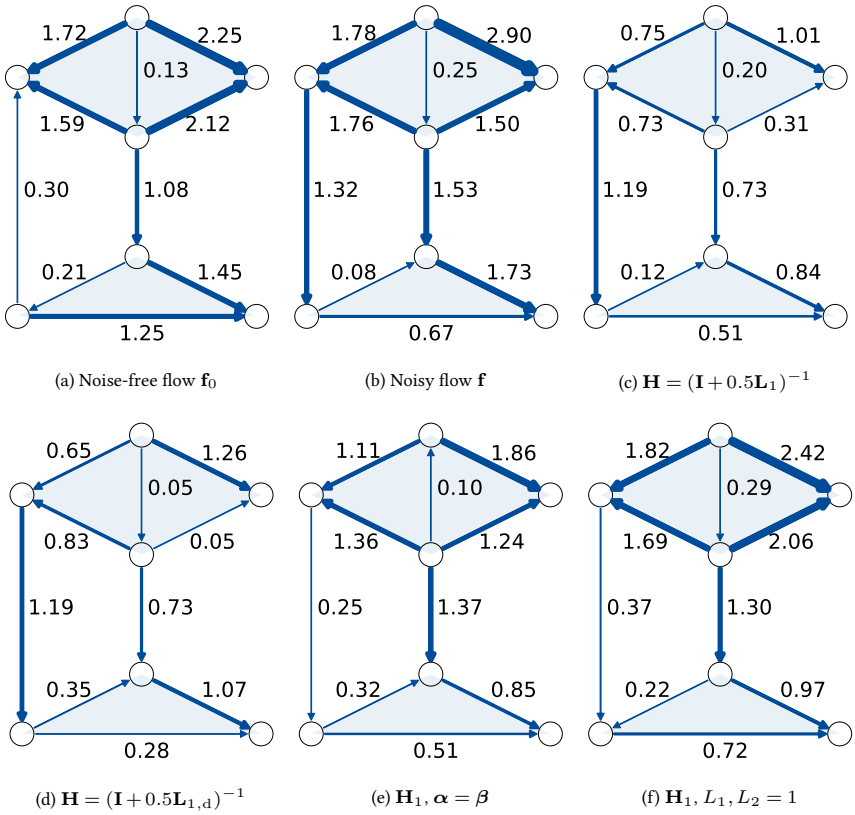
[Jia et al., 2019; Jiang et al., 2011]. When different spectral properties of the underlying edge flow are known, or certain spectral properties of noise are known, we are able to deal with various situations for denoising by properly designing the simplicial filters. To illustrate this we induced a gradient flow from a node signal with a flat spectrum and added Gaussian noise with an error of 0.46, shown in Figs. Fig. 2.9a and Fig. 2.9b. Fig. 2.9 contrasts the denoising results of the regularized optimization methods with our filters that preserve the gradient component based on an LS design (Corollary 2.10). As Fig. 2.9c to Fig. 2.9f, shown in this context the low-pass filters which are implicitly defined via the optimization procedures lead to large errors. In contrast the simplicial filters that preserve the gradient can denoise properly. Setting $\boldsymbol{\alpha} = \boldsymbol{\beta}$ again reduces the performance.

2.6.3 Currency Exchange Market

Table 2.1: Currency exchange rates captured from Yahoo!Finance, not arbitrage-free.

	USD	EUR	CNY	HKD	GBP	JPY	AUD
1 USD	1	0.8422	6.3739	7.7666	0.7207	110.1020	1.3377
1 EUR	1.1873	1	7.5681	9.2218	0.8557	130.7314	1.5883
1 CNY	0.1539	0.1321	1	1.2185	0.1131	17.2683	0.2099
1 HKD	0.1288	0.1085	0.8207	1	0.0928	14.1718	0.1723
1 GBP	1.3871	1.1685	8.8414	10.7732	1	152.6758	1.8557
1 JPY	0.0091	0.0077	0.0579	0.0706	0.0066	1	0.0122
1 AUD	0.7475	0.6299	4.7602	5.8001	0.5385	82.1837	1

A currency exchange market can be described as a network where the vertices represent currencies, and the edges indicate the pairwise exchange rates. If all pairs of currencies are exchangeable, the vertex set $\mathcal{V}$ and edge set $\mathcal{E}$ make up a complete graph. For any currencies $i, j, k \in \mathcal{V}$, we expect an arbitrary-free condition, $r^{i/j} r^{j/k} = r^{i/k}$ with the exchange rate $r^{i/j}$ between i and j, i.e., the exchange path $i \to j \to k$ provides no gain or loss over a direct exchange $i \to k$. If we represent the exchange rates as edge flows $f_{ij} = \log(r^{i/j})$, this can be translated into the fact that $\mathbf{f}$ is curl-free, i.e., $f_{ij} + f_{jk} + f_{ki} = 0$. Therefore, an



(a) Noise-free flow $\mathbf{f}_0$ (b) Noisy flow $\mathbf{f}$ (c) $\mathbf{H} = (\mathbf{I} + 0.5\mathbf{L}_1)^{-1}$

(d) $\mathbf{H} = (\mathbf{I} + 0.5\mathbf{L}_{1,\mathrm{d}})^{-1}$ (e) $\mathbf{H}_1, \boldsymbol{\alpha} = \boldsymbol{\beta}$ (f) $\mathbf{H}_1, L_1, L_2 = 1$

Figure 2.9: Gradient flow denoising. (a) Edge flow $\mathbf{f}_0$ induced by a node signal with a flat spectrum. (b) The noisy observation $\mathbf{f}$ with error $e = 0.46$. (c-d) Denoising with the low-pass filter $\mathbf{H}_P$ with (c) $\mathbf{P} = \mathbf{L}_1$ Schaub et al. [2021], or (d) $\mathbf{P} = \mathbf{L}_{1,\mathrm{d}}$ Schaub & Segarra [2018] leads to an even larger error of $e = 0.70$ or $e = 0.73$, respectively. (e) Denoising by a gradient based simplicial filter $\mathbf{H}_1$ with an order $L_1 = L_2 = 4$ and $\boldsymbol{\alpha} = \boldsymbol{\beta}$, yields a much better result with error $e = 0.39$. (f) Denoising by a general filter $\mathbf{H}_1$ with $\boldsymbol{\alpha} \neq \boldsymbol{\beta}$ provides an even smaller error with $e = 0.23$, even for a lower filter order $L_1 = L_2 = 1$.

ideal exchange edge flow is a gradient flow. This idea was exploited in Jia et al. [2019]; Jiang et al. [2011] to assess arbitrage possibilities in exchange markets, and provide arbitrage free exchange rates, respectively.

Here, we illustrate how we can analogously remove arbitrage opportunities via a simplicial filter that preserves only the gradient component of a given exchange rate flow. For a complete graph, there are two distinct eigenvalues, zero and N_0, for the lower or upper Hodge Laplacian. Then, based on Corollary 2.10 we can extract the gradient component via $\mathbf{H}_1 = \frac{1}{N_0}\mathbf{L}_{1,\mathrm{d}}$. Similarly, filter $\mathbf{H}_1 = \frac{1}{N_0}\mathbf{L}_{1,\mathrm{u}}$ can extract the curl component, which indicate possible arbitrage opportunities.

In Table 2.1, we show a real-world exchange market of seven currencies at 2021/07/12 10:30 UTC from the Currency Converter Yahoo!Finance. We built an SC formed by the

Table 2.2: The gradient component, arbitrage-free, provides a fair market.

	USD	EUR	CNY	HKD	GBP	JPY	AUD
1 USD	1	0.8422	6.3738	7.7665	0.7208	110.0171	1.3385
1 EUR	1.1874	1	7.5680	9.2216	0.8559	130.6292	1.5893
1 CNY	0.1569	0.1321	1	1.2185	0.1131	17.2608	0.2100
1 HKD	0.1288	0.1084	0.8207	1	0.0928	14.1656	0.1723
1 GBP	1.3873	1.1684	8.8425	10.7746	1	152.6286	1.8557
1 JPY	0.0091	0.0077	0.0579	0.0706	0.0066	1	0.0122
1 AUD	0.7471	0.6292	4.7618	5.8022	0.5385	82.1919	1

seven currencies and all the 2-, 3-cliques, where the edge flow $\mathbf{f}$ is the logarithm of the exchange rates in the upper triangular part without the diagonal in Table 2.1. If one unit of currency yields more than 0.3% benefit or loss after two successive exchanges, we say the corresponding exchange rates are non-arbitrage-free. By computing the curl $\mathbf{B}_2^\top \mathbf{f}$, we can identify six such triangles (up to machine precision) that are not arbitrage-free., e.g., USD-JPY-AUD (1 USD would yield 1.0041 USD), EUR-JPY-AUD, and HKD-GBP-JPY. By applying a filter $\mathbf{H}_1 = \frac{1}{N_0}\mathbf{L}_{1,\mathrm{d}}$ on the exchange rate flows, we can extract its gradient flow, leading to the arbitrage-free exchange rate flow in Table 2.2. This yields essentially the same results as solving the LS optimization problem considered in Jiang et al. [2011]. This simple example demonstrates the use of simplicial filters to generate an efficient financial market. Especially in a complete market, the form of the filters is trivial and the computational cost is much smaller compared to solving the LS problems.

2.6.4 London Street Network: Fast PageRank of Edges

PageRank, as a ranking scheme for web pages, can be studied in terms of a random walk on a graph, which can be used to measure the centrality of a node. PageRank was extended to the edge space to assess the topological importance of an edge in Schaub et al. [2020]. The input edge flow $\mathbf{f}$ is an indicator vector which has value one on the edge of interest and zeros on the rest, then a PageRank vector $\boldsymbol{\pi}$ follows the linear system [Schaub et al., 2020, Def. 6.2], $(\gamma\mathbf{I}+\mathbf{L}_{1,\mathrm{n}})\boldsymbol{\pi} = \mathbf{f}$, with $\mathbf{L}_{1,\mathrm{n}}$ being the normalized 1-Hodge Laplacian[2] [Schaub et al., 2020, Def. 3.3] and $\gamma > 0$. The solution is $\boldsymbol{\pi} = (\gamma\mathbf{I}+\mathbf{L}_{1,\mathrm{n}})^{-1}\mathbf{f}$ with the PageRank operator $\mathbf{H}_{\mathrm{PR}} := (\gamma\mathbf{I}+\mathbf{L}_{1,\mathrm{n}})^{-1}$, which does not require to construct the edge space random walk matrix [Schaub et al., 2020, Thm. 3.4] compared to a power iteration implementation. For an indicator edge flow $\mathbf{f}$ of edge i, the absolute values of the entries of $\boldsymbol{\pi}$ are the influence measures edge i has on the edges and the signs the influence orientations w.r.t. the reference orientations [Schaub et al., 2020]. Furthermore, the overall importance of an edge can be assessed with the d_2-norm $\|\boldsymbol{\pi}\|_2$ of its PageRank vector $\boldsymbol{\pi}$. By extracting the gradient component $\boldsymbol{\pi}_{\mathrm{G}}$,we can study the importance of this edge w.r.t. the gradient space via its d_2-norm $\|\boldsymbol{\pi}_{\mathrm{G}}\|_2$ or relative norm $\frac{\|\boldsymbol{\pi}_{\mathrm{G}}\|_2}{\|\boldsymbol{\pi}\|_2}$; likewise for the curl and harmonic components.

[2] $\mathbf{L}_{1,\mathrm{n}}$ admits a similarity transformation to its symmetric version, so its eigenvalues are real and carry the simplicial frequency notion.

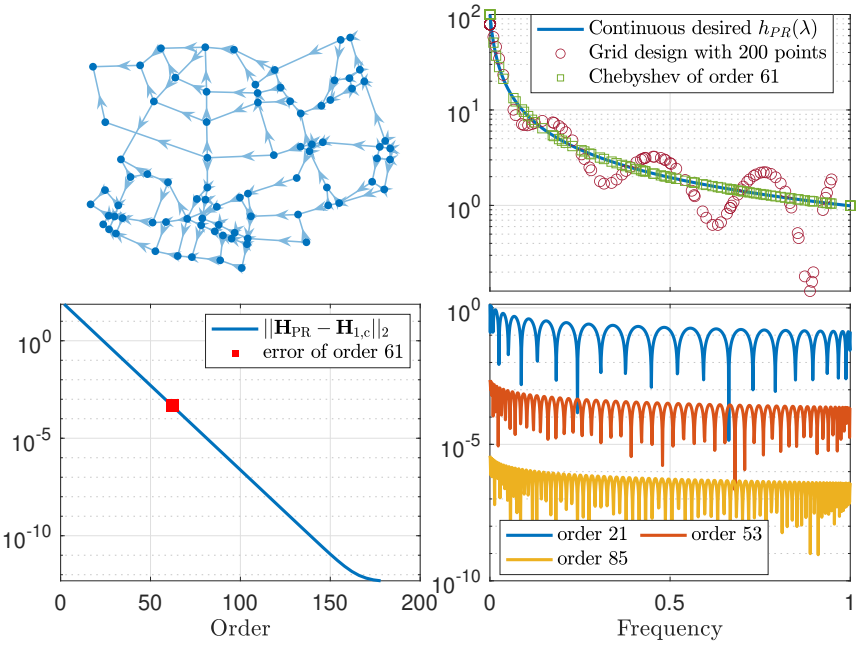


Figure 2.10: PageRank analysis of the street network of London. Top left: network illustration. Top right: Frequency responses of the grid-based designed filter of order 9 and the Chebyshev filter of order 61 w.r.t. the desired continuous frequency response. Bottom left: Spectral norm error of the Chebyshev polynomial filter of different orders. Bottom right: Continuous frequency response errors of the Chebyshev filter of different orders.

Experiment setup. The operator $\mathbf{H}_{\mathrm{PR}}$ can be interpreted as a low pass filter which attenuates the high gradient and curl frequencies. But to implement it we need to invert a matrix. Instead, we propose here a faster variant via a simplicial filter $\mathbf{H}_1 := \mathbf{H}_1(\mathbf{L}_{1,\mathrm{n}})$ built on the normalized Hodge Laplacian. To find the filter coefficients, we considered a grid-based design and a Chebyshev polynomial with a desired response, $g(\lambda) = \frac{1}{\gamma+\lambda}$ with $\lambda \in [0,1]$ and $\gamma = 0.01$. We implemented the PageRank operator in the street network of London with 82 crossings (nodes), 130 streets (edges) and 12 triangles, as shown in the top left of Fig. 2.10 [Schaub et al., 2014; Youn et al., 2008]. We considered a grid-based design with 200 samples within the eigenvalue interval and a filter order of 9 and implemented the Chebyshev polynomials $\mathbf{H}_1$ of different orders.

Results. From Fig. 2.10, we see that the performance of the grid-based design decreases heavily when the filter order is larger than 9 due to the numerical instability, while a Chebyshev design allows a more accurate design as shown in Fig. 2.10.

We then computed the PageRank results of all edges with a Chebyshev filter of order 61 and obtained the norms of their three components in the absolute and relative senses. From Fig. 2.11, we can identify the most influential streets (dark grey) in the network, as the indicator flow on these streets induce a PageRank vector with the largest total norm. The

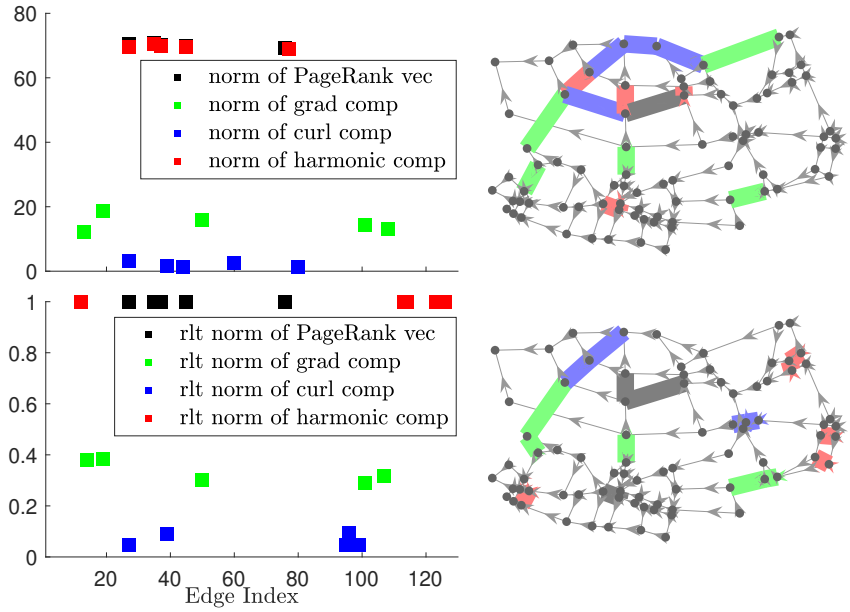


Figure 2.11: PageRank vectors analysis. The scattered squares are the edges whose PageRank vectors contain the top-five largest absolute (Top) and relative (Bottom) total norm, gradient, curl and harmonic norms. The left figures show the PageRank values w.r.t. edge indices. The rights figures show the highlighted edges in the network with shaded grey indicating the total norm, green the gradient norm, blue the curl norm and red the harmonic norm.

streets (green) that have the biggest influence on the gradient space are the ones on which a traffic change leads to congestion on the intersections as the traffic flows on them have a large divergence. The red streets induce the most impact on the harmonic space and the blues ones on the curl space, where the traffic flows tend to induce a global or local cyclic flow, thus a small chance of congestion. The influences are measured in a relative sense in the bottom figures, and we notice that most streets would not cause a large influence on the curl space.

Finally, we show the simplicial PageRank vector of four edges to assess where their influences are concentrated in Fig. 2.12. Edge 19 (top left), sitting in the gradient space, has a large influence in terms of gradient flow components on the surrounding edges to which congestion on edge 19 would spread, as shown in Fig. 2.11. Edge 27 (top right) has a large influence on edges that form 1-dimensional "hole" [Lim, 2020], containing mostly harmonic components. This may imply that a traffic change on edge 27 would less likely cause congestion. Edge 45 (bottom left), whose PageRank result has a large total norm, as seen in Fig. 2.11, acts similar as edge 27. Edge 96 (bottom right) induces smaller influences than the other three, but they reach further in the network. The most influenced edges are its direct upper neighbors, as also seen in the bottom (blue) of Fig. 2.11, where congestion would rather not happen.

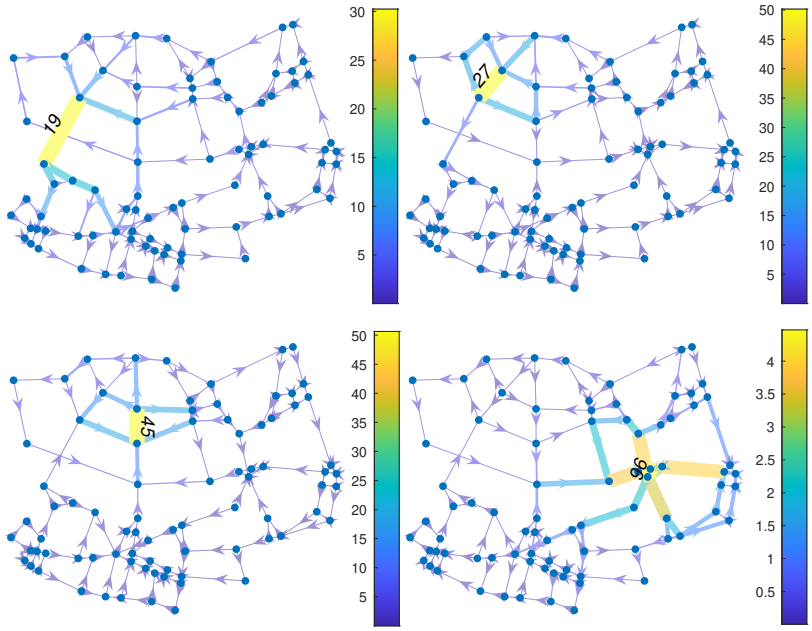


Figure 2.12: Examples of the PageRank vectors of four edges. The edge width and color indicates the magnitude of the PageRank result on that edge. The labeled edges are the chosen edges, also with the largest PageRank results.

2.6.5 Chicago Road Network: Gradient Component Extraction

We now conduct subcomponent extraction on a real-world larger network. On the Chicago road network with 546 nodes, 1088 edges, and 112 triangles [Stabler et al., 2018], we perform the gradient component extraction of the measured traffic flow, which is not divergence-free, via filter $\mathbf{H}_1$ built on the lower Hodge Laplacian. It is challenging to perform the filter design in this setting because some simplicial frequencies are close to each other, leading to the ill-conditioning of the LS design. This can be avoided by the Chebyshev polynomial design. This requires a continuous desired frequency response to perform the gradient component extraction, which ideally is an indicator function $\mathbf{1}_{\lambda>0}$ with $\lambda \in [0, \lambda_{\mathrm{G,max}}]$. Here we use the logistic function $g_{\mathrm{G}}(\lambda) = \frac{1}{1+\exp^{-k(\lambda-\lambda_0)}}$ with the growth rate $k > 0$ and the midpoint λ_0. If the smallest gradient frequency is close to 0, a large k and a small λ_0 are required to achieve a good approximation of the ideal indicator function.

We applied different filter design methods. For the LS-based methods, we set a filter order of 9 to avoid ill-conditioning and considered the decoupled solver. Moreover, we treated the eigenvalues with a difference smaller than 0.3 as the same for the LS design, leading to 30 "different" eigenvalues. For the grid-based design, we uniformly sampled 100 points in the interval $[0, \lambda_{\mathrm{G,max}}]$ with $\lambda_{\mathrm{G,max}} = 10.8$ approximated by a 50-step power iteration. Lastly, we set $k = 100$ and $\lambda_0 = 0.01$ for the logistic function in the Chebyshev polynomial design.

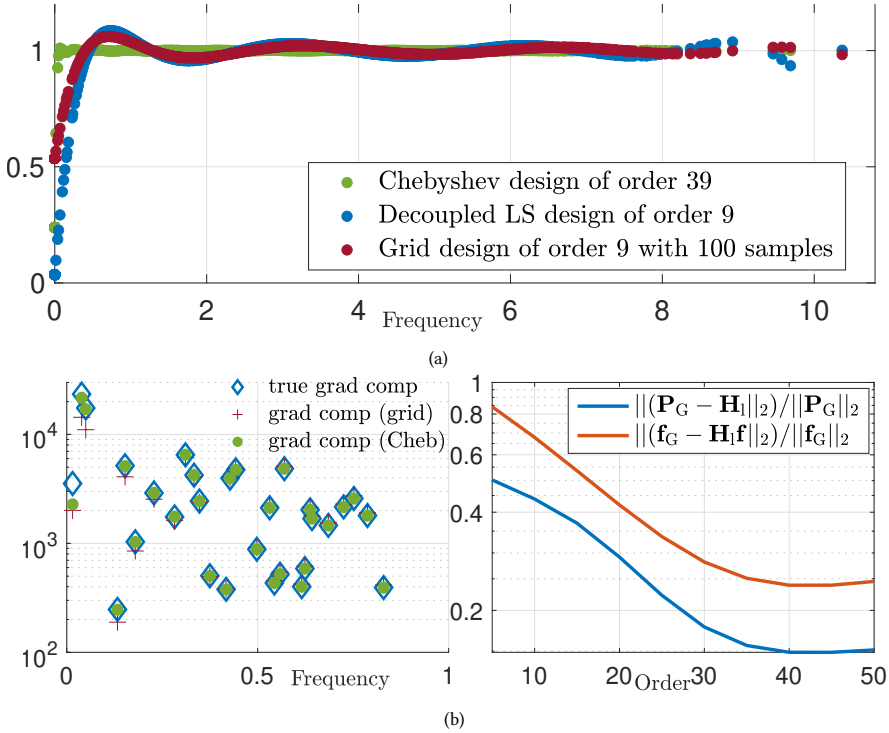


Figure 2.13: Chicago road network gradient component extraction. (a): Filter frequency responses with different designs. (b): Left: SFT of the extracted gradient component in frequency range $[0, 1]$. Right: Approximation errors of Chebyshev filters of different orders and the extracted gradient component.

As seen in Fig. 2.13a, the Chebyshev polynomial of order 39 only has one frequency response smaller than 0.9 at the smallest gradient frequency, while at the remaining frequencies, it is able to well preserve the gradient component. The other methods have a poorer performance especially at small gradient frequencies. We then compared the gradient component extracted by above grid-based and Chebyshev polynomial filters. Fig. 2.13b (left) reports the SFT of the extracted flows at frequencies smaller than 1. The Chebyshev polynomial has a good extraction ability as it performs well even at the very small frequencies where the grid-based design fails. Fig. 2.13b (right) shows the filter design and the extraction errors of the Chebyshev polynomial of different orders. The extraction error cannot be further reduced because the traffic flow contains large components at the small frequencies and the logistic function, after all, is an approximate of the indicator function.

2.7 Conclusion

We proposed a simplicial convolutional filter as a matrix polynomial of the Hodge Laplacians for simplicial signal processing. It generates an output as a linear combination of the shifted

simplicial signals. This shift-and-sum operation is analogous to the convolutions of time series, images and graph signals and allows for a distributed filter implementation. In the frequency domain, the filter acts as a pointwise multiplication respecting the convolution theorem. Furthermore, the lower and upper Hodge Laplacians encode lower and upper adjacencies, respectively. For an edge flow, its lower shifting propagates the flow to its neighbors via the common incident nodes, while the upper one via the common triangles. By assigning two different sets of coefficients on the lower and upper parts in the filter, we can differentiate the lower and upper adjacencies. Via the Hodge decomposition, we see that this corresponds to an independent control of the filter on the gradient and curl spaces in the frequency domain. To achieve a desired frequency response, different filter design approaches are considered with pros and cons. The filter provides a faster and distributed solution for subcomponent extraction, simplicial signal denoising and other tasks in exploiting the higher-order connectivities of the network.

Appendix

2

2.A Proof of Proposition 2.1

Due to the distributivity of matrix-vector multiplication, we have $\mathbf{L}_{1,\mathrm{d}}(a\mathbf{f}_1 + b\mathbf{f}_2) = a\mathbf{L}_{1,\mathrm{d}}\mathbf{f}_1 + b\mathbf{L}_{1,\mathrm{d}}\mathbf{f}_2$, and $\mathbf{L}_{1,\mathrm{u}}(a\mathbf{f}_1 + b\mathbf{f}_2) = a\mathbf{L}_{1,\mathrm{u}}\mathbf{f}_1 + b\mathbf{L}_{1,\mathrm{u}}\mathbf{f}_2$. Then, by working out the expression $\mathbf{H}_1(a\mathbf{f}_1 + b\mathbf{f}_2)$ and using the distributivity w.r.t the addition, the linearity proof completes. Since we have $\mathbf{L}_{1,\mathrm{d}}^{l}\mathbf{L}_{1,\mathrm{d}} = \mathbf{L}_{1,\mathrm{d}}\mathbf{L}_{1,\mathrm{d}}^{l}$, and similarly for for $\mathbf{L}_{1,\mathrm{u}}$, and $\mathbf{L}_{1,\mathrm{d}}\mathbf{L}_{1,\mathrm{u}} = \mathbf{0}$, the shift-invariance holds. The same proof applies to the general case with $k \neq 1$.

2.B Proof of Proposition 2.2

Since the permutation matrix $\mathbf{P}_k$ is orthogonal, i.e., $\mathbf{P}_k\mathbf{P}_k^\top = \mathbf{P}_k^\top\mathbf{P}_k = \mathbf{I}$, we have that $(\mathbf{P}_k\mathbf{L}_{k,\mathrm{d}}\mathbf{P}_k^\top)^{l_1} = \mathbf{P}_k\mathbf{L}_{k,\mathrm{d}}^{l_1}\mathbf{P}_k^\top$, and similarly $(\mathbf{P}_k\mathbf{L}_{k,\mathrm{u}}\mathbf{P}_k^\top)^{l_2} = \mathbf{P}_k\mathbf{L}_{k,\mathrm{u}}^{l_2}\mathbf{P}_k^\top$. Thus, we can express the permuted simplicial filter as

$$\begin{aligned}
\bar{\mathbf{H}}_k &= h_0\mathbf{I} + \sum_{l_1=1}^{L_1}\alpha_{l_1}(\mathbf{P}_k\mathbf{L}_{k,\mathrm{d}}\mathbf{P}_k^\top)^{l_1} + \sum_{l_2=1}^{L_2}\beta_{l_2}(\mathbf{P}_k\mathbf{L}_{k,\mathrm{u}}\mathbf{P}_k^\top)^{l_2} \\
&= h_0\mathbf{I} + \sum_{l_1=1}^{L_1}\alpha_{l_1}\mathbf{P}_k\mathbf{L}_{k,\mathrm{d}}^{l_1}\mathbf{P}_k^\top + \sum_{l_2=1}^{L_2}\beta_{l_2}\mathbf{P}_k\mathbf{L}_{k,\mathrm{u}}^{l_2}\mathbf{P}_k^\top \\
&= \mathbf{P}_k\left(h_0\mathbf{I} + \sum_{l_1=1}^{L_1}\alpha_{l_1}\mathbf{L}_{k,\mathrm{d}}^{l_1} + \sum_{l_2=1}^{L_2}\beta_{l_2}\mathbf{L}_{k,\mathrm{u}}^{l_2}\right)\mathbf{P}_k^\top \\
&= \mathbf{P}_k\mathbf{H}_k\mathbf{P}_k^\top.
\end{aligned} \tag{2.44}$$

The output on the permuted SC can be written as $\bar{\mathbf{x}}_{\mathrm{o}}^k := \bar{\mathbf{H}}_k\bar{\mathbf{x}}^k = \bar{\mathbf{H}}_k(\mathbf{P}_k\mathbf{x}^k) = \mathbf{P}_k\mathbf{H}_k\mathbf{P}_k^\top\mathbf{P}_k\mathbf{x}^k = \mathbf{P}_k\mathbf{H}_k\mathbf{x}^k := \mathbf{P}_k\mathbf{x}_{\mathrm{o}}^k$. The proof completes.

2.C Proof of Proposition 2.3

The diagonal matrix $\mathbf{D}_k$ satisfies that $\mathbf{D}_k\mathbf{D}_k^\top = \mathbf{D}_k^\top\mathbf{D}_k = \mathbf{I}$. Following from that, we have that $(\mathbf{D}_k\mathbf{L}_{k,\mathrm{d}}\mathbf{D}_k^\top)^{l_1} = \mathbf{D}_k\mathbf{L}_{k,\mathrm{d}}^{l_1}\mathbf{D}_k^\top$, and similarly $(\mathbf{D}_k\mathbf{L}_{k,\mathrm{u}}\mathbf{D}_k^\top)^{l_2} = \mathbf{D}_k\mathbf{L}_{k,\mathrm{u}}^{l_2}\mathbf{D}_k^\top$. Following the same procedure in (2.44), we have

$$\bar{\mathbf{H}}_k = h_0\mathbf{I} + \sum_{l_1=1}^{L_1}\alpha_{l_1}(\mathbf{D}_k\mathbf{L}_{k,\mathrm{d}}\mathbf{D}_k^\top)^{l_1} + \sum_{l_2=1}^{L_2}\beta_{l_2}(\mathbf{D}_k\mathbf{L}_{k,\mathrm{u}}\mathbf{D}_k^\top)^{l_2} = \mathbf{D}_k\mathbf{H}_k\mathbf{D}_k^\top. \tag{2.45}$$

Thus, the filter output on the reoriented simplices can be expressed as $\bar{\mathbf{x}}_{\mathrm{o}}^{k} := \bar{\mathbf{H}}_k \bar{\mathbf{x}}^k = \bar{\mathbf{H}}_k(\mathbf{D}_k \mathbf{x}^k) = \mathbf{D}_k \mathbf{H}_k \mathbf{x}^k := \mathbf{D}_k \mathbf{s}_{\mathrm{o}}^{k}$. The proof completes.

2.D Proof of Proposition 2.5

We show the proof for each item.

1) First, we show that the image of $\mathbf{L}_{1,\mathrm{d}}$ is equivalent to the gradient space $\mathrm{im}(\mathbf{B}_1^\top)$.

1. To show $\mathrm{im}(\mathbf{L}_{1,\mathrm{d}}) \subseteq \mathrm{im}(\mathbf{B}_1^\top)$: From $\mathbf{L}_{1,\mathrm{d}} = \mathbf{B}_1^\top \mathbf{B}_1$, for any non-zero $\mathbf{x} \in \mathrm{im}(\mathbf{L}_{1,\mathrm{d}})$, we have $\mathbf{x} = \mathbf{L}_{1,\mathrm{d}}\mathbf{y}$, and we can always find a vector $\mathbf{z} = \mathbf{B}_1\mathbf{y} \in \mathbb{R}^{N_0}$ such that $\mathbf{x} = \mathbf{B}_1^\top \mathbf{z}$;
2. To show $\mathrm{im}(\mathbf{B}_1^\top) \subseteq \mathrm{im}(\mathbf{L}_{1,\mathrm{d}})$: for every non-zero $\mathbf{x} \in \mathrm{im}(\mathbf{B}_1^\top)$, we can find some $\mathbf{y} \perp \ker(\mathbf{B}_1^\top)$ such that $\mathbf{x} = \mathbf{B}_1^\top \mathbf{y} \neq \mathbf{0}$. This implies $\mathbf{y} \in \mathrm{im}(\mathbf{B}_1)$, so there exists some $\mathbf{z} \in \mathbb{R}^{N_1}$ such that $\mathbf{y} = \mathbf{B}_1\mathbf{z}$ and $\mathbf{x} = \mathbf{B}_1^\top \mathbf{B}_1 \mathbf{z} = \mathbf{L}_{1,\mathrm{d}}\mathbf{z}$, and hence $\mathrm{im}(\mathbf{B}_1^\top) \subseteq \mathrm{im}(\mathbf{L}_{1,\mathrm{d}})$.

Combining (i) and (ii), we have that $\mathrm{im}(\mathbf{L}_{1,\mathrm{d}}) = \mathrm{im}(\mathbf{B}_1^\top)$.

Second, we show that the eigenvectors $\mathbf{U}_{\mathrm{G}}$ of $\mathbf{L}_{1,\mathrm{d}}$ associated with nonzero eigenvalues span the image of $\mathbf{L}_{1,\mathrm{d}}$. As $\mathbf{L}_{1,\mathrm{d}}$ is positive semidefinite (PSD), thus, diagonalizable, the geometric multiplicity of every eigenvalue equals to the algebraic multiplicity. That is, all the eigenvectors are linearly independent and form an eigenbasis. Then, matrix $\mathbf{U}_{\mathrm{G}}$ has a full column rank. Furthermore, for any $\mathbf{x} \in \mathrm{im}(\mathbf{L}_{1,\mathrm{d}})$, we have $\mathbf{x} = \mathbf{L}_{1,\mathrm{d}}\mathbf{y} = \mathbf{U}_{\mathrm{G}}\mathbf{\Lambda}_{\mathrm{G}}\mathbf{U}_{\mathrm{G}}^\top\mathbf{y}$, i.e., $\mathbf{x} \in \mathrm{im}(\mathbf{U}_{\mathrm{G}})$ and $\mathrm{im}(\mathbf{L}_{1,\mathrm{d}}) \subseteq \mathrm{im}(\mathbf{U}_{\mathrm{G}})$. Due to $\dim \mathrm{im}(\mathbf{L}_{1,\mathrm{d}}) = \mathrm{rank}(\mathbf{L}_{1,\mathrm{d}}) = N_{\mathrm{G}}$, matrix $\mathbf{U}_{\mathrm{G}}$ spans the image of $\mathbf{L}_{1,\mathrm{d}}$ and the gradient space $\mathrm{im}(\mathbf{B}_1^\top)$.

2) The proof of 2) follows similarly as the proof of 1).

3) For arbitrary eigenvectors $\mathbf{u}_{\mathrm{G}}$ in $\mathbf{U}_{\mathrm{G}}$ and $\mathbf{u}_{\mathrm{C}}$ in $\mathbf{U}_{\mathrm{C}}$, we have $\mathbf{u}_{\mathrm{G}} = \frac{1}{\lambda_{\mathrm{G}}}\mathbf{L}_{1,\mathrm{d}}\mathbf{u}_{\mathrm{G}}$ and $\mathbf{u}_{\mathrm{C}} = \frac{1}{\lambda_{\mathrm{C}}}\mathbf{L}_{1,\mathrm{u}}\mathbf{u}_{\mathrm{C}}$. Thus, from (2.2), their inner product follows $\mathbf{u}_{\mathrm{G}}^\top\mathbf{u}_{\mathrm{C}} = \frac{1}{\lambda_{\mathrm{G}}\lambda_{\mathrm{C}}}\mathbf{u}_{\mathrm{G}}^\top\mathbf{L}_{1,\mathrm{d}}\mathbf{L}_{1,\mathrm{u}}\mathbf{u}_{\mathrm{C}} = 0$, i.e., $\mathbf{U}_{\mathrm{G}} \perp \mathbf{U}_{\mathrm{C}}$. Moreover, from the definition of $\mathbf{L}_1$, matrix $[\mathbf{U}_{\mathrm{G}}\ \mathbf{U}_{\mathrm{C}}]$ contains the eigenvectors of $\mathbf{L}_1$ associated with nonzero eigenvalues. The Hodge decomposition indicates that $\mathrm{im}(\mathbf{B}_1^\top) \oplus \mathrm{im}(\mathbf{B}_2) = \mathrm{im}(\mathbf{L}_1)$. By combining with 1) and 2), we have that $\mathrm{im}(\mathbf{B}_1^\top) = \mathrm{im}(\mathbf{U}_{\mathrm{G}})$ and $\mathrm{im}(\mathbf{B}_2) = \mathrm{im}(\mathbf{U}_{\mathrm{C}})$.

4) As $\mathbf{L}_1$ is PSD, the eigenvectors associated to zero eigenvalues are linearly independent. Any vector $\mathbf{x} \in \ker(\mathbf{L}_1)$ follows $\mathbf{L}_1\mathbf{x} = \mathbf{0}$, which means $\mathbf{x}$ is an eigenvector associated with an eigenvalue 0, i.e., $\ker(\mathbf{L}_1) = \mathrm{im}(\mathbf{U}_{\mathrm{H}})$ with dimension N_{H}. Moreover, we have $\ker(\mathbf{L}_1) = \ker(\mathbf{L}_{1,\mathrm{d}}) \cap \ker(\mathbf{L}_{1,\mathrm{u}})$ from the definition of $\mathbf{L}_1$, then the columns of $\mathbf{U}_{\mathrm{H}}$ can be used as eigenvectors of $\mathbf{L}_{1,\mathrm{d}}$ or $\mathbf{L}_{1,\mathrm{u}}$ associated with zero eigenvalues. From 3) and $\ker(\mathbf{L}_1) = \mathrm{im}(\mathbf{U}_{\mathrm{H}})$, we have $\mathbf{U}_{\mathrm{H}} \perp \mathbf{U}_{\mathrm{G}}$ and $\mathbf{U}_{\mathrm{H}} \perp \mathbf{U}_{\mathrm{C}}$. Thus, matrix $[\mathbf{U}_{\mathrm{H}}\ \mathbf{U}_{\mathrm{C}}]$ (or $[\mathbf{U}_{\mathrm{H}}\ \mathbf{U}_{\mathrm{G}}]$) provides the eigenvectors of $\mathbf{L}_{1,\mathrm{d}}$ (or $\mathbf{L}_{1,\mathrm{u}}$) associated with zero eigenvalues, and the proof completes.

5) From 3) and 4), we have that matrix $\mathbf{U}_1$ collects all eigenvectors of $\mathbf{L}_1$. From 1), 2), and 4), we have that $\mathbf{U}_1$ provides all eigenvectors for $\mathbf{L}_{1,\mathrm{d}}$ and $\mathbf{L}_{1,\mathrm{u}}$.

2.E Proof of Proposition 2.6

With condition i), we can eigendecompose the operator $\mathbf{G}$ as $\mathrm{diag}(\mathbf{g}) = \mathbf{U}_1^\top \mathbf{G} \mathbf{U}_1$, then the equivalence between $\mathbf{G}$ and $\mathbf{H}_1$ can be achieved through a set of linear equations in the spectral domain, i.e., $H_1(\lambda_i) = g_i$, for all $i = 1, \ldots, N_1$. Based on condition ii), this set of linear equations is equivalent to linear system (2.28) and (2.29) with $1 + D_\mathrm{G} + D_\mathrm{C}$ equations. With the filter order requirement in condition iii), the Vandermonde matrices $\mathbf{\Phi}_\mathrm{G}$ and $\mathbf{\Phi}_\mathrm{C}$ have full row rank. Thus, there exist at least one solution to problem (2.28) and the proof completes.

2.F Proof of Proposition 2.7

The cost function J in problem (2.29) is convex w.r.t. variables, h_0, $\boldsymbol{\alpha}$ and $\boldsymbol{\beta}$. Thus, we could find the optimality condition by setting the gradients of the cost function w.r.t. the three variables to zeros, given by

$$\begin{cases} \nabla_{h_0} J = h_0 - g_0 + (h_0 \mathbf{1} + \mathbf{\Phi}_\mathrm{G} \boldsymbol{\alpha} - \mathbf{g}_\mathrm{G})^\top \mathbf{1} + (h_0 \mathbf{1} + \mathbf{\Phi}_\mathrm{C} \boldsymbol{\beta} - \mathbf{g}_\mathrm{C})^\top \mathbf{1} = 0, \\ \nabla_{\boldsymbol{\alpha}} J = \mathbf{\Phi}_\mathrm{G}^\top (h_0 \mathbf{1} + \mathbf{\Phi}_\mathrm{G} \boldsymbol{\alpha} - \mathbf{g}_\mathrm{G}) = \mathbf{0}, \\ \nabla_{\boldsymbol{\beta}} J = \mathbf{\Phi}_\mathrm{C}^\top (h_0 \mathbf{1} + \mathbf{\Phi}_\mathrm{C} \boldsymbol{\beta} - \mathbf{g}_\mathrm{C}) = \mathbf{0}. \end{cases} \tag{2.46}$$

First, consider the case where we have that $\|\mathbf{\Phi}_\mathrm{G} \mathbf{\Phi}_\mathrm{G}^\dagger - \mathbf{I}\|_F = 0$ and $\|\mathbf{\Phi}_\mathrm{C} \mathbf{\Phi}_\mathrm{C}^\dagger - \mathbf{I}\|_F = 0$, i.e., $\mathbf{\Phi}_\mathrm{G} \mathbf{\Phi}_\mathrm{G}^\dagger = \mathbf{I}$ and $\mathbf{\Phi}_\mathrm{C} \mathbf{\Phi}_\mathrm{C}^\dagger = \mathbf{I}$, then the solution (2.30) results in that $(\hat{h}_0 \mathbf{1} + \mathbf{\Phi}_\mathrm{G} \hat{\boldsymbol{\alpha}} - \mathbf{g}_\mathrm{G}) = \mathbf{0}$ and $(\hat{h}_0 \mathbf{1} + \mathbf{\Phi}_\mathrm{C} \hat{\boldsymbol{\beta}} - \mathbf{g}_\mathrm{C}) = \mathbf{0}$, which satisfies the optimality condition (2.46).

Second, consider the general case that $\|\mathbf{\Phi}_\mathrm{G} \mathbf{\Phi}_\mathrm{G}^\dagger - \mathbf{I}\|_F \neq 0$ and $\|\mathbf{\Phi}_\mathrm{C} \mathbf{\Phi}_\mathrm{C}^\dagger - \mathbf{I}\|_F \neq 0$. By substituting the solution (2.30) into the optimality condition (2.46), we have

$$\begin{cases} \nabla_{h_0} J = \big((\mathbf{\Phi}_\mathrm{G} \mathbf{\Phi}_\mathrm{G}^\dagger - \mathbf{I})(\mathbf{g}_\mathrm{G} - g_0 \mathbf{1})\big)^\top \mathbf{1} + \big((\mathbf{\Phi}_\mathrm{C} \mathbf{\Phi}_\mathrm{C}^\dagger - \mathbf{I})(\mathbf{g}_\mathrm{C} - g_0 \mathbf{1}\big)^\top \mathbf{1}, \\ \nabla_{\boldsymbol{\alpha}} J = \mathbf{\Phi}_\mathrm{G}^\top (\mathbf{\Phi}_\mathrm{G} \mathbf{\Phi}_\mathrm{G}^\dagger - \mathbf{I})(\mathbf{g}_\mathrm{G} - g_0 \mathbf{1}), \\ \nabla_{\boldsymbol{\beta}} J = \mathbf{\Phi}_\mathrm{C}^\top (\mathbf{\Phi}_\mathrm{C} \mathbf{\Phi}_\mathrm{C}^\dagger - \mathbf{I})(\mathbf{g}_\mathrm{C} - g_0 \mathbf{1}). \end{cases} \tag{2.47}$$

For any matrix $\mathbf{A} \in \mathbb{R}^{m \times n}$ with singular values σ_i, $i = 1, \ldots, \min\{m, n\}$, we have that $\|\mathbf{A}\|_F = \big(\sum_{i=1}^{\min\{m,n\}} \sigma_i^2\big)^{\frac{1}{2}}$. If it holds $\|\mathbf{\Phi}_\mathrm{G} \mathbf{\Phi}_\mathrm{G}^\dagger - \mathbf{I}\|_F \to 0$ and $\|\mathbf{\Phi}_\mathrm{C} \mathbf{\Phi}_\mathrm{C}^\dagger - \mathbf{I}\|_F \to 0$, i.e., the Frobenius norms approach to zero, the number of trivial (zero) singular values of $\mathbf{\Phi}_\mathrm{G} \mathbf{\Phi}_\mathrm{G}^\dagger - \mathbf{I}$ and $\mathbf{\Phi}_\mathrm{C} \mathbf{\Phi}_\mathrm{C}^\dagger - \mathbf{I}$ increases. Accordingly, the number of trivial entries in $(\mathbf{\Phi}_\mathrm{G} \mathbf{\Phi}_\mathrm{G}^\dagger - \mathbf{I})(\mathbf{g}_\mathrm{G} - g_0 \mathbf{1})$ and $(\mathbf{\Phi}_\mathrm{C} \mathbf{\Phi}_\mathrm{C}^\dagger - \mathbf{I})(\mathbf{g}_\mathrm{C} - g_0 \mathbf{1})$ increases, which corresponds to a suboptimal condition of (2.46), i.e., the gradients $\nabla_{h_0} J(\hat{h}_0, \hat{\boldsymbol{\alpha}}, \hat{\boldsymbol{\beta}}) \to 0$, $\nabla_{\boldsymbol{\alpha}} J(\hat{h}_0, \hat{\boldsymbol{\alpha}}) \to \mathbf{0}$ and $\nabla_{\boldsymbol{\beta}} J(\hat{h}_0, \hat{\boldsymbol{\beta}}) \to \mathbf{0}$. The proof completes.

2.G Proof of Proposition 2.8

Since the operator $\mathbf{G}$ corresponds to the desired continuous harmonic, gradient and curl frequency responses, it can be diagonalized by $\mathbf{U}_1$. Therefore, we have that

$$\begin{aligned}\|\mathbf{G}-\mathbf{H}_{\mathrm{c}}\|_2 &= \|\mathbf{U}_1(g(\mathbf{\Lambda})-\tilde{H}_1(\mathbf{\Lambda}))\mathbf{U}_1^{\top}\|_2 \\ &= \|g(\mathbf{\Lambda})-\tilde{H}_1(\mathbf{\Lambda})\|_2 = \max_{i=1,\ldots,N_1}|g(\lambda_i)-\tilde{H}_1(\lambda_i)|\end{aligned} \tag{2.48}$$

where the diagonal matrix $g(\mathbf{\Lambda})$ has entries $g(\lambda_i)=g_0$ for $\lambda_i\in\mathcal{Q}_{\mathrm{H}}$, $g(\lambda_i)=g_{\mathrm{G}}(\lambda_i)$ for $\lambda_i\in\mathcal{Q}_{\mathrm{G}}$ and $g(\lambda_i)=g_{\mathrm{C}}(\lambda_i)$ for $\lambda_i\in\mathcal{Q}_{\mathrm{C}}$. The frequency response $\tilde{H}_1(\lambda_i)$ for $\lambda_i\in\mathcal{Q}$ is given in (2.39). Moreover, based on the definition of $B_1(L_1)$ and $B_2(L_2)$ we have that

$$\max_{i=1,\ldots,N_1}|g(\lambda_i)-\tilde{H}_1(\lambda_i)| \leq \max\left\{B_1(L_1),B_2(L_2)\right\}=B. \tag{2.49}$$

The proof completes.

2.H Proof of Lemma 2.9

We first show the equivalence between the two projection operator forms. From the Hodge decomposition, the gradient, the curl and the harmonic components are in the subspaces $\mathrm{im}(\mathbf{B}_1^{\top})$, $\mathrm{im}(\mathbf{B}_2)$ and $\ker(\mathbf{L}_1)$, respectively. Furthermore, from Proposition 2.5, we have that $\mathrm{im}(\mathbf{B}_1^{\top})=\mathrm{im}(\mathbf{U}_{\mathrm{G}})$, $\mathrm{im}(\mathbf{B}_2)=\mathrm{im}(\mathbf{U}_{\mathrm{C}})$ and $\ker(\mathbf{L}_1)=\mathrm{im}(\mathbf{U}_{\mathrm{H}})$. Thus, each subcomponent can be obtained as the orthogonal projection of $\mathbf{f}$ onto the subspace spanned by the eigenbasis, i.e., the LS estimate. The gradient projection is $\mathbf{P}_{\mathrm{G}}:=\mathbf{U}_{\mathrm{G}}\mathbf{U}_{\mathrm{G}}^{\top}$, the curl one $\mathbf{P}_{\mathrm{C}}:=\mathbf{U}_{\mathrm{C}}\mathbf{U}_{\mathrm{C}}^{\top}$ and the harmonic one $\mathbf{P}_{\mathrm{H}}:=\mathbf{U}_{\mathrm{H}}\mathbf{U}_{\mathrm{H}}^{\top}$. This can be shown via the SFT as well, i.e., $\mathbf{f}_{\mathrm{G}}=\mathbf{U}_{\mathrm{G}}\tilde{\mathbf{f}}_{\mathrm{G}}=\mathbf{U}_{\mathrm{G}}\mathbf{U}_{\mathrm{G}}^{\top}\mathbf{f}$, likewise for the other two.

Second, the simplicial filter $\mathbf{H}_1$ can implement the gradient projector $\mathbf{P}_{\mathrm{G}}=\mathbf{U}_{\mathrm{G}}\mathbf{U}_{\mathrm{G}}^{\top}$, if and only if we have that

$$\mathbf{H}_1=\mathbf{U}_{\mathrm{G}}\mathbf{U}_{\mathrm{G}}^{\top}=[\mathbf{U}_{\mathrm{G}}^{\perp}\ \mathbf{U}_{\mathrm{G}}]\begin{bmatrix}\mathbf{0} & \\ & \mathbf{I}_{N_{\mathrm{G}}}\end{bmatrix}\begin{bmatrix}(\mathbf{U}_{\mathrm{G}}^{\perp})^{\top}\\ \mathbf{U}_{\mathrm{G}}^{\top}\end{bmatrix}, \tag{2.50}$$

with $\mathbf{U}_{\mathrm{G}}^{\perp}=[\mathbf{U}_{\mathrm{H}}\ \mathbf{U}_{\mathrm{C}}]$. We have that $\mathbf{H}_1=\mathbf{U}_1\tilde{\mathbf{H}}_1\mathbf{U}_1^{\top}$ from (2.24). Thus, (2.50) is equivalent to problem (2.28) with $g_0=0$, $\mathbf{g}_{\mathrm{C}}=\mathbf{0}$ and $\mathbf{g}_{\mathrm{G}}=\mathbf{1}$. With $L_1=D_{\mathrm{G}}$ and $L_2=D_{\mathrm{C}}$, there admits a unique solution $\{h_0,\boldsymbol{\alpha},\boldsymbol{\beta}\}$ to system (2.28). Similar procedure can be followed for the implementation of the curl and harmonic projectors. The proof completes.

2.I Proof of Corollary 2.10

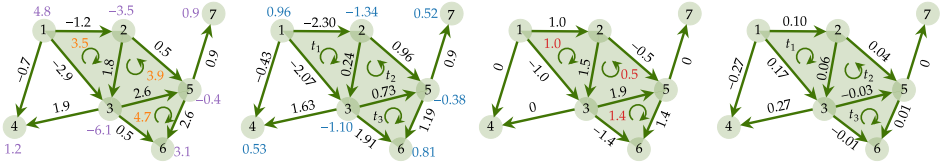
To implement the projector $\mathbf{P}_{\mathrm{G}}$ via $\mathbf{H}_1$ with $L_2=0$ ($\boldsymbol{\beta}=\mathbf{0}$), from (2.50), it is equivalent to set $g_0=0$ for $\lambda_i\in\mathcal{Q}_{\mathrm{H}}\cup\mathcal{Q}_{\mathrm{C}}$ and $g(\lambda_i)=1$ for $\lambda_i\in\mathcal{Q}_{\mathrm{G}}$ in (2.25), i.e., $\mathbf{g}=[0\ \mathbf{1}_{D_{\mathrm{G}}}^{\top}]^{\top}$. This returns problem (2.28) without $\mathbf{\Phi}_{\mathrm{C}}$ and $\boldsymbol{\beta}$, i.e.,

$$\min_{h_0,\boldsymbol{\alpha}}\left\|\left[\mathbf{1}\ \middle|\ \begin{matrix}\mathbf{0}\\ \mathbf{\Phi}_{\mathrm{G}}\end{matrix}\right]\begin{bmatrix}h_0\\ \boldsymbol{\alpha}\end{bmatrix}-\mathbf{g}\right\|_2^2. \tag{2.51}$$

If $L_1 = D_{\mathrm{G}}$, the system matrix is square and any two rows are linearly independent, it admits a unique solution of $h_0 = 0$ and $\boldsymbol{\alpha} = \boldsymbol{\Phi}_{\mathrm{G}}^{-1}\mathbf{1}$. Similar procedure can be followed for the curl projector $\mathbf{P}_{\mathrm{C}}$. The proof completes.

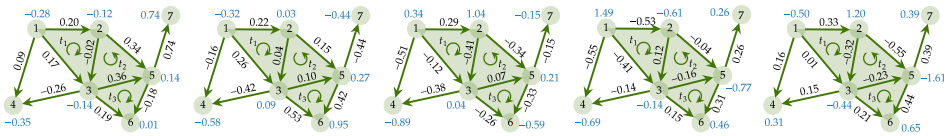
2.J Illustrations of Hodge decomposition and SFT on aother SC

In addition to the illustrations previously, we here demonstrate the Hodge decomposition in Appendix 2.J and the simplicial Fourier basis and frequency in Fig. 2.15 on the SC in Fig. 1.3.

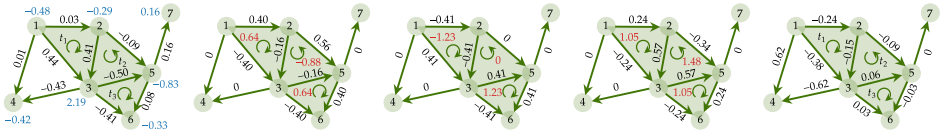


(a) Edge flow (b) Gradient flow (c) Curl flow (d) Harmonic flow

Figure 2.14: (a) An edge flow where we denote its divergence and curl in purple and orange, respectively. (b)-(d) The Hodge decomposition of the edge flow in (a). The gradient flow is the gradient of some node signal (in blue) and is curl-free. The curl flow can be obtained from some triangle flow (in red), and is divergence-free. The harmonic flow has zero divergence and zero curl, and is circulating around the hole $\{1,3,4\}$. Note that in this figure, the flow numbers are rounded upper to two decimal places. Thus, at some nodes or triangles with zero-divergence or zero-curl, the divergence or curl might not be exactly zero.



(a) $\mathbf{u}_{\mathrm{G},1}, \lambda_{\mathrm{G},1}(0.80)$ (b) $\mathbf{u}_{\mathrm{G},2}, \lambda_{\mathrm{G},2}(1.61)$ (c) $\mathbf{u}_{\mathrm{G},3}, \lambda_{\mathrm{G},3}(2.43)$ (d) $\mathbf{u}_{\mathrm{G},4}, \lambda_{\mathrm{G},4}(3.96)$ (e) $\mathbf{u}_{\mathrm{G},5}, \lambda_{\mathrm{G},5}(5.12)$



(f) $\mathbf{u}_{\mathrm{G},6}, \lambda_{\mathrm{G},6}(6.08)$ (g) $\mathbf{u}_{\mathrm{C},1}, \lambda_{\mathrm{C},1}(1.59)$ (h) $\mathbf{u}_{\mathrm{C},2}, \lambda_{\mathrm{C},2}(3.00)$ (i) $\mathbf{u}_{\mathrm{C},3}, \lambda_{\mathrm{C},3}(4.41)$ (j) $\mathbf{u}_{\mathrm{H}}, \lambda_{\mathrm{H}}(0)$

Figure 2.15: (a)-(f) Six gradient frequencies and the corresponding Fourier basis. We also annotate their divergences, and we see that these eigenvectors with a small eigenvalue have a small magnitude of total divergence, i.e., the edge flow variation in terms of the nodes. Gradient frequencies reflect the nodal variations. (g)-(i) Three curl frequencies and the corresponding Fourier basis. We annotate their curls and we see that these eigenvectors with a small eigenvalue have a small magnitude of total curl, i.e., the edge flow variation in terms of the triangles. Curl frequencies reflect the rotational variations. (j) Harmonic basis with a zero frequency, which has a zero nodal and zero rotational variation.

3

Hodge-Aware Convolutional Learning on Simplicial Complexes

In the previous chapter, we introduced the simplicial convolutional filters that are based on the Hodge Laplacians. They are able to filter the simplicial signals independently in the different Hodge subspaces. This linear convolution operation allows us to construct neural network models to perform learning on simplicial complexes, in analogy to how graph neural networks are built upon graph convolutions. In this chapter, we consider a more general framework, simplicial complex convolutional neural networks (SCCNNs), where the learning is performed over simplices of different orders. The success of neural network models has been attributed to their ability to exploit the symmetry of the data, and their stability against perturbations. This motivates us to investigate the properties of SCCNNs in terms of their equivariance and stability, and more importantly, the Hodge decomposition. This chapter is based on the work of Yang et al. [2022a; 2025].

3.1 Introduction

In the line of geometric deep learning [Bronstein et al., 2021], there is a growing interest in learning from data defined on simplicial complexes. The motivation behind this comes from two limitations of standard graph neural networks (GNNs). First, graphs are limited to model pairwise interactions between data entites on nodes, yet polyadic (multi-way) interactions often arise in real-world networks [Battiston et al., 2020; Benson et al., 2021; Torres et al., 2021], such as friendship networks [Newman et al., 2002], collaboration networks [Benson et al., 2018], gene regulatory networks [Masoomy et al., 2021]. Second, graphs are often used to support signals on the nodes, and standard graph signal processing and GNN approaches often revolve around signals and features on nodes. Yet, signals involved with multiple entities are less researched compared to signals on nodes (with one

entity). They arise as signal flows on edges, signals on triangles and so on. For example, in physical networks, we may encounter water flows in a water supply network [Money et al., 2022], traffic flows in a road network [Jia et al., 2019], trading flows in financial networks [Lim, 2020] and information flows in brain networks [Anand et al., 2022], as well as in human-generated networks, we have collaboration data, such as triadic collaborations in coauthorship networks [Benson et al., 2018].

Simplicial complexes are a popular higher-order network model and have been shown effective to address both limitations of graph-based models [Bick et al., 2023]. They are composed of topological objects, namely, nodes, edges, triangles, etc., which are simplices of different orders. Simplicial complexes naturally describe more topological (higher-order) relationships in networks, thus, having more topological expressive power than graphs. This has been the main motivation of recent neural networks developed on simplicial complexes [Bodnar et al., 2021b; Bunch et al., 2020; Chen et al., 2022d; Ebli et al., 2020; Giusti et al., 2022; Roddenberry & Segarra, 2019; Roddenberry et al., 2021]. We also refer readers to the recent surveys [Besta et al., 2024; Papamarkou et al., 2024]. In analogy to standard GNNs relying on the adjacency between nodes, the central idea behind these works is to rely on the relationships between simplices to enable learning. Such relations can be twofold: first, two simplices can be lower and upper adjacent to each other, e.g., an edge can be (lower) adjacent to another via a shared node, and can also be (upper) adjacent to another by locating in a common triangle; and second, there exist inter-simplicial couplings (or simplicial incidences) between simplices of different orders, as shown in Fig. 4.1a. The aforementioned works mainly vary in terms of either message-passing or convolutional flavor, or the type of simplicial relationships relying, either on only simplicial adjacencies or on both adjacencies and incidences.

Furthermore, signals can be defined on simplices to model the data related to multiple entites in networks. This has been the main focus of topological signal processing literature [Barbarossa & Sardellitti, 2020; Schaub et al., 2021; Yang et al., 2022b]. The celebrated *combinatorial Hodge decomposition* arising from discrete calculus [Grady & Polimeni, 2010; Lim, 2020] provides a unique and characteristic decomposition of simplicial signals into three components. This is particularly intuitive for edge flows which allows their decomposition into *gradient flows, curl flows and harmonic flows*, that are, respectively, curl-free, divergence-free or both. These notions from discrete calculus interestingly allow us to capture some physical properties of the simplicial signals, such as the conservation laws [Grady & Polimeni, 2010]. More importantly, this decomposition offers a tool to better analyze simplicial signals, as reported in statistical ranking problems, financial exchange markets [Jiang et al., 2011], traffic networks [Jia et al., 2019], brain networks [Anand et al., 2022] and game theory [Candogan et al., 2011]. We hypothesize it will further promote better principled and effective learning methods on simplicial complexes.

Given this context, we reckon that the aforementioned works on simplicial neural networks mostly focus on the pure topological aspect of simplicial complexes. It lacks theoretical analyses of their learning capabilities from the Hodge spectral perspective. Also, since SCs are often built from data and are prone to estimation uncertainty, the learning on SCs benefits from a stability analysis to investigate their robustness against perturbations on the simplicial topologies. Thus, in this chapter, we propose a more general and unified

framework, namely, *simplicial complex convolutional neural network* (SCCNN), and we focus on the following three theoretical aspects. Our **contributions** are as follows:

- In Section 3.2 we introduce SCCNN and emphasize its three principles, namely, uncoupling the lower and upper simplicial adjacencies, accounting for the inter-simplicial couplings, and performing higher-order convolutions. We then use the Dirichlet energy minimization on SCs to understand how uncoupling the lower and upper adjacencies in Hodge Laplacians, as well as the inter-simplicial couplings can mitigate simplicial oversmoothing.
- In Section 3.3, we characterize the spectral behavior of SCCNN and its expressive power under the help of spectral simplicial theory [Barbarossa & Sardellitti, 2020; Steenbergen, 2013; Yang et al., 2021]. We show that an SCCNN performs independent and expressive learning in the three subspaces of the Hodge decomposition, which are invariant under its learning operators. This Hodge-awareness (or Hodge-aided bias) allows for effective and rational learning on SCs compared to MLPs or simplicial message-passing networks [Bodnar et al., 2021b].
- In Section 3.4, we obtain a theoretical stability bound on the SCCNN outputs against small perturbations on the simplicial connections. This allows us to see how the three principles and other network factors can affect the stability, as well as the limitations of SCCNNs. This analysis in turn guides the design of convolutional architectures.

In Section 3.5, we validate our theoretical findings and highlight the effect of the three principles, the need for the Hodge-aware learning, as well as the stability, based on different simplicial tasks including recovering foreign currency exchange (forex) rates, predicting triadic and tetradic collaborations, and ocean current trajectories. Finally, we conclude the chapter with a discussion on this work and its relations to existing works.

3.2 Simplicial Complex CNNs

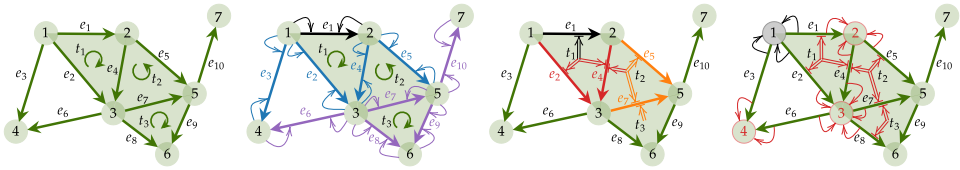
We first introduce the general convolutional architecture on SCs, then discuss the propreties of SCCNN and study the effects of the three principles from an energy minimization perspective.

In an SC, taking $\mathbf{x}_{k-1}^{l-1}, \mathbf{x}_k^{l-1}$ and $\mathbf{x}_{k+1}^{l-1}$ as inputs, an SCCNN at layer $l = 1, \dots, L$ computes the k-simplicial output $\mathbf{x}_k^l$ via a map

$$\mathrm{SCCNN}_k^l : \{\mathbf{x}_{k-1}^{l-1}, \mathbf{x}_k^{l-1}, \mathbf{x}_{k+1}^{l-1}\} \to \mathbf{x}_k^l, \quad \mathbf{x}_k^l = \sigma(\mathbf{H}_{k,\mathrm{d}}^l \mathbf{x}_{k,\mathrm{d}}^{l-1} + \mathbf{H}_k^l \mathbf{x}_k^{l-1} + \mathbf{H}_{k,\mathrm{u}}^l \mathbf{x}_{k,\mathrm{u}}^{l-1}) \tag{3.1}$$

with

$$\mathbf{H}_k = \sum_{t=0}^{T_\mathrm{d}} w_{k,\mathrm{d},t} \mathbf{L}_{k,\mathrm{d}}^t + \sum_{t=0}^{T_\mathrm{u}} w_{k,\mathrm{u},t} \mathbf{L}_{k,\mathrm{u}}^t, \quad \mathbf{H}_{k,\mathrm{d}} = \sum_{t=0}^{T_\mathrm{d}} w'_{k,\mathrm{d},t} \mathbf{L}_{k,\mathrm{d}}^t, \quad \mathbf{H}_{k,\mathrm{u}} = \sum_{t=0}^{T_\mathrm{u}} w'_{k,\mathrm{u},t} \mathbf{L}_{k,\mathrm{u}}^t. \tag{3.2}$$



(a) SC example (b) Lower edge conv. (c) Upper edge conv. (d) Inter-simplicial locality

Figure 3.1: (a) An SC where arrows indicate the reference orientations of edges and triangles. 2-simplices are (filled) triangles shaded in green and open triangle $\{1,3,4\}$ is not in the SC. (b) Lower convolution via $\mathbf{H}_1$ and $\mathbf{H}_{1,\mathrm{d}}$ on edge e_1. (c) Upper convolution via $\mathbf{H}_1$ and $\mathbf{H}_{1,\mathrm{u}}$ on e_1. (d) Node 1 (in black) contains information from its neighbors $\{2,3,4\}$ (nodes in red), and projected information from edges which contribute to these neighbors (denoted by arrows in red from edges to nodes), and from triangles $\{t_1,t_2,t_3\}$ which contribute to those edges (denoted by double arrows in red from triangle centers to edges). This interaction is the coupling between the intra- and the extended inter-simplicial locality.

Here, $\mathbf{H}_k$ denotes a *simplicial convolutional filter* (SCF, [Yang et al., 2022b]) with two sets of learnable coefficients $\{w_{k,\mathrm{d},t}\},\{w_{k,\mathrm{u},t}\}$, while $\mathbf{H}_{k,\mathrm{d}}$ and $\mathbf{H}_{k,\mathrm{u}}$ are the lower and upper SCFs, respectively. Moreover, $\mathbf{x}_{k,\mathrm{d}}^{l-1}=\mathbf{B}_k^\top\mathbf{x}_{k-1}^{l-1}$ and $\mathbf{x}_{k,\mathrm{u}}^{l-1}=\mathbf{B}_{k+1}\mathbf{x}_{k+1}^{l-1}$ are the lower and upper projections from $(k\pm1)$-simplices via incidence relations to k-simplices, respectively, and $\sigma(\cdot)$ is an elementwise nonlinearity. The convolution operations in this SCCNN can be understood as follows: 1) The previous k-simplicial output $\mathbf{x}_k^{l-1}$ is passed to an SCF $\mathbf{H}_k^l$ of orders $T_\mathrm{d},T_\mathrm{u}$, which performs a linear combination of the signals from the lower-adjacent (up to T_d-hop) and upper-adjacent (up to T_u-hop) simplices. 2) The previous $k\pm1$-simplicial outputs $\mathbf{x}_{k\pm1}^{l-1}$ are first projected to k-simplices, which are then convolved using a lower SCF and an upper SCF, respectively.

We give two examples where the first is a SCCNN on a SC of order two, and the second is the form of SCCNN with multi-features.

Example 3.1. For $k=0,1,2$, a SCCNN layer reads as

$$\begin{aligned}\mathbf{x}_0^l&=\sigma(\mathbf{H}_0^l\mathbf{x}_0^{l-1}+\mathbf{H}_{0,\mathrm{u}}^l\mathbf{B}_1\mathbf{x}_1^{l-1}),\\ \mathbf{x}_1^l&=\sigma(\mathbf{H}_{1,\mathrm{d}}^l\mathbf{B}_1^\top\mathbf{x}_0^{l-1}+\mathbf{H}_1^l\mathbf{x}_1^{l-1}+\mathbf{H}_{1,\mathrm{u}}^l\mathbf{B}_2\mathbf{x}_2^{l-1}),\\ \mathbf{x}_2^l&=\sigma(\mathbf{H}_{2,\mathrm{d}}^l\mathbf{B}_2^\top\mathbf{x}_1^{l-1}+\mathbf{H}_2^l\mathbf{x}_2^{l-1}).\end{aligned}\tag{3.3}$$

Recursively, we see that a SCCNN layer takes as inputs $\{\mathbf{x}_0^{l-1},\mathbf{x}_0^{l-2},\mathbf{x}_1^{l-2},\mathbf{x}_2^{l-2}\}$ to compute $\mathbf{x}_0^l$. One may find this familar as some type of skip connections in GNNs [Xu et al., 2018b].

Example 3.2 (Multi-Feature SCCNN). A multi-feature SCCNN at layer l takes $\{\mathbf{X}_{k-1}^{l-1},\mathbf{X}_k^{l-1},\mathbf{X}_{k+1}^{l-1}\}$ as inputs, each of which has F_{l-1} features, and generates an output $\mathbf{X}_k^l$ with F_l features as

$$\begin{aligned}\mathbf{X}_k^l=\sigma\Bigg(&\sum_{t=0}^{T_\mathrm{d}}\mathbf{L}_{k,\mathrm{d}}^t\mathbf{B}_k^\top\mathbf{X}_{k-1}^{l-1}\mathbf{W}_{k,\mathrm{d},t}'^l+\sum_{t=0}^{T_\mathrm{d}}\mathbf{L}_{k,\mathrm{d}}^t\mathbf{X}_k^{l-1}\mathbf{W}_{k,\mathrm{d},t}^l\\&+\sum_{t=0}^{T_\mathrm{u}}\mathbf{L}_{k,\mathrm{u}}^t\mathbf{X}_k^{l-1}\mathbf{W}_{k,\mathrm{u},t}^l+\sum_{t=0}^{T_\mathrm{u}}\mathbf{L}_{k,\mathrm{u}}^t\mathbf{B}_{k+1}\mathbf{X}_{k+1}^{l-1}\mathbf{W}_{k,\mathrm{u},t}'^l\Bigg)\end{aligned}\tag{3.4}$$

where $\mathbf{L}^t$ indicates the matrix t-power of $\mathbf{L}$, while superscript l indicates the layer index.

In Fig. 3.1, we provide an example of SCCNN for the edge case $k = 1$. We focus on edge e_1 and consider the cases $T_{\mathrm{d}} = T_{\mathrm{u}} = 2$. On edge e_1, the SCF $\mathbf{H}_1$ linearly combines the signals from its direct lower neighbors (edges in blue) and two-hop lower neighbors (edges in purple), as shown in Fig. 3.1b. It also combines the signals from the direct upper neighbors (edges in red) and two-hop upper neighbors (edges in orange), as shown in Fig. 3.1c. At the same time, the signals on nodes are projected on the edges, denoted by arrows in blue and purple from nodes to edges in Fig. 3.1b, which are then combined to edge e_1 by the lower SCF $\mathbf{H}_{1,\mathrm{d}}$. The signals on triangles are projected on the edges as well, denoted by double arrows in red and orange in Fig. 3.1c, which are combined to edge e_1 by the upper SCF $\mathbf{H}_{1,\mathrm{u}}$.

This architecture subsumes the convolutional learning methods on SCs in Bunch et al. [2020]; Chen et al. [2022d]; Ebli et al. [2020]; Roddenberry et al. [2021]; Yang et al. [2022a;c]. We refer to Appendix 3.B for a detailed discussion. Particularly, we here emphasize on the key three principles of an SCCNN layer:

(P1) It uncouples the lower and upper parts in the Hodge Laplacian. This leads to an independent treatment of the lower and upper adjacencies, achieved by using two sets of learnable weights. We shall see in Section 3.3 that how this relates to the independent and expressive learning in the Hodge subspaces.

(P2) It accounts for the inter-simplicial couplings via the incidence relations. The projections $\mathbf{x}_{k,\mathrm{d}}$ and $\mathbf{x}_{k,\mathrm{u}}$ carry nontrivial information contained in the faces and cofaces of simplices (by Hodge decomposition [cf. Section 2.4.1]).

(P3) It performs higher-order convolutions. We consider $T_{\mathrm{d}}, T_{\mathrm{u}} \geq 1$ in SCFs which leads to a multi-hop receptive field on SCs.

In short, each SCCNN layer propagates information across SCs based on two simplicial adjacencies and two incidences in a multi-hop fashion.

3.2.1 Properties

Simplicial locality. The simplicial convolutions admit an intra-simplicial locality where the output $\mathbf{H}_k\mathbf{x}_k$ is localized in T_{d}-hop lower and T_{u}-hop upper k-simplicial neighborhoods [Yang et al., 2022b]. An SCCNN preserves such property as $\sigma(\cdot)$ does not alter the information locality. It also admits an inter-simplicial locality between k- and $(k \pm 1)$-simplices due to the inter-simplicial couplings. This further extends to simplices of orders $k \pm l$ if $L \geq l$ because $\mathbf{B}_k\sigma(\mathbf{B}_{k+1}) \neq \mathbf{0}$ [Schaub et al., 2021]. Moreover, the intra- and inter-simplicial localities are coupled in a multi-hop way through higher-order convolutions such that, for example, a node not only interacts with its incident edges and the triangles including it, but also with those further hops away, as shown in Fig. 3.1d. We refer to Appendix 3.A.1 for a more formal discussion.

Complexity. An SCCNN layer has a parameter complexity of order $\mathcal{O}(T_{\mathrm{d}} + T_{\mathrm{u}})$ and a computational complexity $\mathcal{O}(k(n_k + n_{k+1}) + n_k m_k(T_{\mathrm{d}} + T_{\mathrm{u}}))$, which are linear in the

simplex dimensions. Here, m_k is the maximum number of neighbors for k-simplices. We refer to Appendix 3.A.2 for more details.

3.2.2 Symmetries of SCs and simplicial data, Equivariance of SCCNNs

Permutation symmetry of SCs. There exists a permutation group P_{n_k} for each set $\mathcal{S}^k$ in a SC of order K. For $K=0$, this gives the graph permutation group. We can combine these groups for different simplex orders by a group product to form a larger permutation group $P = \times_k P_{n_k}$, which is a symmetry group of SCs and simplicial data, assuming vertices in each simplex are consistently ordered. That is, we have, for $p = (p_0, p_1, \ldots, p_K) \in P$, $[p \cdot \mathbf{L}_k]_{ij} = [\mathbf{L}_k]_{p_k^{-1}(i)p_k^{-1}(j)}$, $[p \cdot \mathbf{B}_k]_{ij} = [\mathbf{B}_k]_{p_{k-1}^{-1}(i)p_k^{-1}(j)}$, and $[p \cdot \mathbf{x}_k]_i = [\mathbf{x}_k]_{p_k^{-1}(i)}$. This permutation symmetry of SCs gives us the freedom to list simplices in any order.

Orientation symmetry of simplicial data. The orientation of a simplex is an equivalence class that two orientations are equivalent if they differ by an even permutation Lim [2020]; Munkres [2018]. Thus, for a simplex $s_i^k = \{i_0, \ldots, i_k\}$ with $k > 0$, we have an *orientation symmetry* group $O_{k,i} = \{o_{k,i}^+, o_{k,i}^-\}$ by a group homomorphism which maps all the even permutations of $\{i_0, \ldots, i_k\}$ to the identity element $o_{k,i}^+$ and all the odd permutations to the reverse operation $o_{k,i}^-$.

We can further combine the orientation groups of all simplices in a SC as $O = \times_{i,k} O_{k,i}$ by using a group product. This however is not a symmetry group of an oriented SC because $o_{k,i}^- \cdot \mathbf{L}_k$ changes the signs of $\mathbf{L}_k$ elements in ith column and row, and $o_{k,i}^- \cdot \mathbf{B}_k$ changes the ith row, resulting in a different SC topology. Instead, it is a symmetry group of the data space, due to its alternating nature w.r.t. simplices. For $o \in O$ we have $[o \cdot \mathbf{x}_k]_i = o_{k,i} \cdot f_k(s_i^k) = f_k(o_{k,i}^{-1} \cdot s_i^k)$, i.e., $[\mathbf{x}_k]_i$ remains unchanged w.r.t. the changed orientation of s_i^k. This gives us the freedom to choose reference orientations of simplices when working with simplicial data.

Theorem 3.3 (Permutation Equivariance). *A SCCNN in (3.1) is P-equivariant. For all $p \in P$, we have $p \cdot \mathrm{SCCNN}_k : \{p_{k-1} \cdot \mathbf{x}_{k-1}, p_k \cdot \mathbf{x}_k, p_{k+1} \cdot \mathbf{x}_{k+1}\} \to p_k \mathbf{x}_k$.*

Theorem 3.4 (Orientation Equivariance). *A SCCNN in (3.1) is O-equivariant if $\sigma(\cdot)$ is odd. For all $o \in O$, we have $o \cdot \mathrm{SCCNN}_k : \{o_{k-1} \cdot \mathbf{x}_{k-1}, o_k \cdot \mathbf{x}_k, o_{k+1} \cdot \mathbf{x}_{k+1}\} \to o_k \cdot \mathbf{x}_k$.*

Proof. (informal) Both the permutation group and orientation group have linear matrix representations. By following the same procedure in [Bodnar et al., 2021b, Appendix D] or Roddenberry et al. [2021], we can prove the equivariance. □

3.2.3 A simplicial Dirichlet energy perspective

Here we analyze the convolution architecture in (3.1) from an energy minimization perspective. First, we extend the notion of Dirichlet energy from graphs to SCs.

Definition 3.5. The *Dirichlet energy* of a k-simplicial signal $\mathbf{x}_k$ is

$$D(\mathbf{x}_k) = D_{\mathrm{d}}(\mathbf{x}_k) + D_{\mathrm{u}}(\mathbf{x}_k) := \|\mathbf{B}_k \mathbf{x}_k\|_2^2 + \|\mathbf{B}_{k+1}^\top \mathbf{x}_k\|_2^2. \tag{3.5}$$

This Dirichlet energy returns the graph Dirichlet energy when $k=0$. In this case, $D(\mathbf{x}_0)=\|\mathbf{B}_1^\top\mathbf{x}_0\|_2^2=\sum_i\sum_j\|x_{0,i}-x_{0,j}\|^2$ is the ℓ_2-norm of the *gradient* of the node signal $\mathbf{x}_0$. For edge flow $\mathbf{x}_1$, $D(\mathbf{x}_1)$ consists of two parts, $\|\mathbf{B}_1\mathbf{x}_1\|_2^2$ and $\|\mathbf{B}_2^\top\mathbf{x}_1\|_2^2$, which measure the total divergence and curl of $\mathbf{x}_1$, respectively, i.e., the edge flow variations w.r.t. nodes and triangles. In the general case, $D_{\mathrm{d}}(\mathbf{x}_k)$ and $D_{\mathrm{u}}(\mathbf{x}_k)$ measure the lower and upper k-simplicial signal variations w.r.t. the faces and cofaces, respectively. A harmonic k-simplicial signal $\mathbf{x}_k$ has zero Dirichlet energy, e.g., a constant node signal and a div- and curl-free edge flow.

Simplicial shifting as Hodge Laplacian smoothing. Bunch et al. [2020]; Yang et al. [2022c] considered $\mathbf{H}_k$ to be a weighted variant of $\mathbf{I}-\mathbf{L}_k$, generalizing the graph convolutional network (GCN) [Kipf & Welling, 2017]. This is necessarily a Hodge Laplacian smoothing as in Schaub et al. [2021]—given an initial signal $\mathbf{x}_k^0$, we consider the Dirichlet energy minimization:

$$\begin{aligned}&\min_{\mathbf{x}_k}\|\mathbf{B}_k\mathbf{x}_k\|_2^2+\gamma\|\mathbf{B}_{k+1}^\top\mathbf{x}_k\|_2^2,\ \gamma>0,\\ &\text{gradient descent: }\mathbf{x}_{k,\mathrm{gd}}^{l+1}=(\mathbf{I}-\eta\mathbf{L}_{k,\mathrm{d}}-\eta\gamma\mathbf{L}_{k,\mathrm{u}})\mathbf{x}_k^l\end{aligned}\tag{3.6}$$

with step size $\eta>0$. The simplicial shifting $\mathbf{x}_k^{l+1}=w_0(\mathbf{I}-\mathbf{L}_k)\mathbf{x}_k^l$ is a gradient descent step with $\eta=\gamma=1$ and weighted by w_0. A minimizer of (3.6) with $\gamma=1$ is in fact in the harmonic space $\ker(\mathbf{L}_k)$. Thus, a neural network composed of simplicial shifting layers may generate an output with an exponentially decreasing Dirichlet energy as it deepens, formulated by the following proposition. We refer to this as *simplicial oversmoothing*, a notion that generalizes the oversmoothing of a GCN and its variants [Cai & Wang, 2020; Nt & Maehara, 2019; Rusch et al., 2023].

Proposition 3.6. *If $w_0^2\|\mathbf{I}-\mathbf{L}_k\|_2^2<1$, $D(\mathbf{x}_k^{l+1})$ in a neural network of simplicial shifting layers exponentially converges to zero.*

However, when uncoupling the lower and upper parts of $\mathbf{L}_k$ in this shifting, associated to the case $\gamma\neq1$, the decrease of $D(\mathbf{x}_k)$ can slow down or cease because the objective function in (3.6) instead looks for a solution primarily in either $\ker(\mathbf{B}_k)$ (for $\gamma\ll1$) or $\ker(\mathbf{B}_{k+1}^\top)$ (for $\gamma\gg1$), not necessarily in $\ker(\mathbf{L}_k)$, as we shall corroborate in Section 3.5.

Inter-simplicial couplings as sources. Given some nontrivial $\mathbf{x}_{k-1}$ and $\mathbf{x}_{k+1}$, we consider the optimization

$$\begin{aligned}&\min_{\mathbf{x}_k}\|\mathbf{B}_k\mathbf{x}_k-\mathbf{x}_{k-1}\|_2^2+\|\mathbf{B}_{k+1}^\top\mathbf{x}_k-\mathbf{x}_{k+1}\|_2^2,\\ &\text{gradient descent: }\mathbf{x}_{k,\mathrm{gd}}^{l+1}=(\mathbf{I}-\eta\mathbf{L}_k)\mathbf{x}_k^l+\eta(\mathbf{x}_{k,\mathrm{d}}+\mathbf{x}_{k,\mathrm{u}})\end{aligned}\tag{3.7}$$

with step size $\eta>0$. It resembles the convolutional layer, $\mathbf{x}_k^{l+1}=w_0(\mathbf{I}-\mathbf{L}_k)\mathbf{x}_k^l+w_1\mathbf{x}_{k,\mathrm{d}}+w_2\mathbf{x}_{k,\mathrm{u}}$ with some learnable weights, in Bunch et al. [2020]; Yang et al. [2022c].

Proposition 3.7. *We have*

$$\begin{aligned}\|\mathbf{x}_{k-1}\|_2^2+\|x_{k+1}\|_2^2\le D(\mathbf{x}_k^{l+1})\le w_0^2\|\mathbf{I}-\mathbf{L}_k\|_2^2D(\mathbf{x}_k^l)\\ +w_1^2\lambda_{\max}(\mathbf{L}_{k,\mathrm{d}})\|\mathbf{x}_{k,\mathrm{d}}\|_2^2+w_2^2\lambda_{\max}(\mathbf{L}_{k,\mathrm{u}})\|\mathbf{x}_{k,\mathrm{u}}\|_2^2.\end{aligned}\tag{3.8}$$

The signal projections from the lower and upper simplices act as energy sources for $\mathbf{x}_k^l$, and also the objective function in (3.7) looks for an $\mathbf{x}_k$ in the image spaces of $\mathbf{B}_{k+1}$ and $\mathbf{B}_k^\top$, instead of $\ker(\mathbf{L}_k)$. This ensues that $D(\mathbf{x}_k^{l+1})$ may not converge to zero, but rather to a nontrivial value $\|\mathbf{x}_{k-1}\|_2^2 + \|x_{k+1}\|_2^2$. Thus, inter-simplicial couplings pay a role in mitigating the oversmoothing.

Here we showed that simply generalzing GCNs to simplices will inherit its oversmoothing risks. However, by uncoupling the lower and upper Laplacians and accounting for the inter-simplicial couplings we could mitigate this issue. This can also be explained by means of a diffusion process on SCs [Ziegler et al., 2022], which we discuss in Appendix 3.A.3.

3.3 From Convolutional to Hodge-Aware

In this section, we first introduce the *Hodge-invariant operator*, which is an operator such that the three Hodge subspaces are invariant under it. Then, we show that the SCF is such an operator and SCCNN, guided by the three principles (P1-P3), performs *Hodge-invariant learning*, allowing for rational and effective learning on SCs while remaining expressive. Throughout the exposition, we rely on the simplicial spectral theory developed in Section 2.4, which also allows us to characterize the expressive power of SCCNNs. We refer to the detailed derivations and proofs in Appendix 3.D.

Definition 3.8 (Invariant subspace)**.** Let V be a finite-dimensional vector space over $\mathbb{R}$ with $\dim(V) \geq 1$, and let $T : V \to V$ be an operator in V. A subspace $U \subset V$ is an *invariant subspace* under T if $Tu \in U$ for all $u \in U$, i.e., the image of every vector in U under T remains within U. We denote this as $T|_U : U \to U$ where $T|_U$ is the restriction of T on U.

Given the notion of invariant subspace, we then define the Hodge-invariant operators.

Definition 3.9 (Hodge-invariant operator)**.** Let $\square \in \{\mathrm{im}(\mathbf{B}_k^\top), \mathrm{im}(\mathbf{B}_{k+1}), \ker(\mathbf{L}_k)\}$ be any Hodge subspace of $\mathbb{R}^{n_k}$. A linear transformation $F : \mathbb{R}^{n_k} \to \mathbb{R}^{n_k}$ is a *Hodge-invariant operator* if for all $\mathbf{x}_k \in \square$ it holds that $F(\mathbf{x}_k) \in \square$. That is, any simplicial signal in a certain Hodge subspace remains in that subspace under F.

Proposition 3.10. *The SCF $\mathbf{H}_k$ is a Hodge-invariant operator. That is, for any $\mathbf{x}_k \in \square$, we have $\mathbf{H}_k\mathbf{x}_k \in \square$, for $\square \in \{\mathrm{im}(\mathbf{B}_k^\top), \mathrm{im}(\mathbf{B}_{k+1}), \ker(\mathbf{L}_k)\}$. Moreover, the SCF operation can be implicitly written as*

$$\mathbf{H}_k\mathbf{x}_k = \mathbf{H}_k|_{\mathrm{im}(\mathbf{B}_k^\top)}\mathbf{x}_{k,\mathrm{G}} + \mathbf{H}_k|_{\ker(\mathbf{L}_k)}\mathbf{x}_{k,\mathrm{H}} + \mathbf{H}_k|_{\mathrm{im}(\mathbf{B}_{k+1})}\mathbf{x}_{k,\mathrm{C}} \tag{3.9}$$

where $\mathbf{H}_k|_{\mathrm{im}(\mathbf{B}_k^\top)} = \sum_{t=1}^{T_\mathrm{d}} w_{k,\mathrm{d},t}\mathbf{L}_{k,\mathrm{d}}^t + (w_{k,\mathrm{d},0} + w_{k,\mathrm{u},0})\mathbf{I}$ is the restriction of $\mathbf{H}_k$ on the gradient space $im(\mathbf{B}_k^\top)$, $\mathbf{H}_k|_{\ker(\mathbf{L}_k)} = (w_{k,\mathrm{d},0} + w_{k,\mathrm{u},0})\mathbf{I}$ is the restriction of $\mathbf{H}_k$ on the harmonic space, and $\mathbf{H}_k|_{\mathrm{im}(\mathbf{B}_{k+1})} = \sum_{t=0}^{T_\mathrm{u}} w_{k,\mathrm{u},t}\mathbf{L}_{k,\mathrm{u}}^t + (w_{k,\mathrm{d},0} + w_{k,\mathrm{u},0})\mathbf{I}$ is the restriction on the curl space.

Provided with the Hodge-invariance of $\mathbf{H}_k$ and the SFT, we can perform a spectral analysis, which is of interest to further understand the SCCNN since simplicial frequencies reflect the variation characteristics of simplicial signals.

3.3.1 Spectral analysis

Consider the SFT $\tilde{\mathbf{x}}_k = [\tilde{\mathbf{x}}_{k,\mathrm{H}}^\top, \tilde{\mathbf{x}}_{k,\mathrm{G}}^\top, \tilde{\mathbf{x}}_{k,\mathrm{C}}^\top]^\top$ of $\mathbf{x}_k$ where each component is the intensity of $\mathbf{x}_k$ at a certain simplicial frequency. We can understand how an SCCNN convolutional layer $\mathbf{y}_k = \mathbf{H}_{k,\mathrm{d}}\mathbf{x}_{k,\mathrm{d}} + \mathbf{H}_k\mathbf{x}_k + \mathbf{H}_{k,\mathrm{u}}\mathbf{x}_{k,\mathrm{u}}$ regulates/learns from the simplicial signals at different frequencies by performing the SFT

$$\begin{cases} \tilde{\mathbf{y}}_{k,\mathrm{H}} = \tilde{\mathbf{h}}_{k,\mathrm{H}} \odot \tilde{\mathbf{x}}_{k,\mathrm{H}} \\ \tilde{\mathbf{y}}_{k,\mathrm{G}} = \tilde{\mathbf{h}}_{k,\mathrm{d}} \odot \tilde{\mathbf{x}}_{k,\mathrm{d}} + \tilde{\mathbf{h}}_{k,\mathrm{G}} \odot \tilde{\mathbf{x}}_{k,\mathrm{G}} \\ \tilde{\mathbf{y}}_{k,\mathrm{C}} = \tilde{\mathbf{h}}_{k,\mathrm{C}} \odot \tilde{\mathbf{x}}_{k,\mathrm{C}} + \tilde{\mathbf{h}}_{k,\mathrm{u}} \odot \tilde{\mathbf{x}}_{k,\mathrm{u}}, \end{cases} \tag{3.10}$$

with $\odot$ the elementwise multiplication. The n_k-dimensional vector $\tilde{\mathbf{h}}_k = \mathrm{diag}(\mathbf{U}_k^\top \mathbf{H}_k \mathbf{U}_k)$ $= [\tilde{\mathbf{h}}_{k,\mathrm{H}}^\top \; \tilde{\mathbf{h}}_{k,\mathrm{G}}^\top \; \tilde{\mathbf{h}}_{k,\mathrm{C}}^\top]^\top$ is the frequency response vector of $\mathbf{H}_k$ with

$$\begin{aligned} &\tilde{\mathbf{h}}_{k,\mathrm{H}} = (w_{k,\mathrm{d},0} + w_{k,\mathrm{u},0})\mathbf{1}, \quad \tilde{\mathbf{h}}_{k,\mathrm{G}} = \textstyle\sum_{t=0}^{T_\mathrm{d}} w_{k,\mathrm{d},t} \boldsymbol{\lambda}_{k,\mathrm{G}}^{\odot t} + w_{k,\mathrm{u},0}\mathbf{1}, \text{ and} \\ &\tilde{\mathbf{h}}_{k,\mathrm{C}} = \sum_{t=0}^{T_\mathrm{u}} w_{k,\mathrm{u},t} \boldsymbol{\lambda}_{k,\mathrm{C}}^{\odot t} + w_{k,\mathrm{d},0}\mathbf{1}, \end{aligned} \tag{3.11}$$

where $\cdot^{\odot t}$ is the elementwise t-th power of a vector. Likewise,

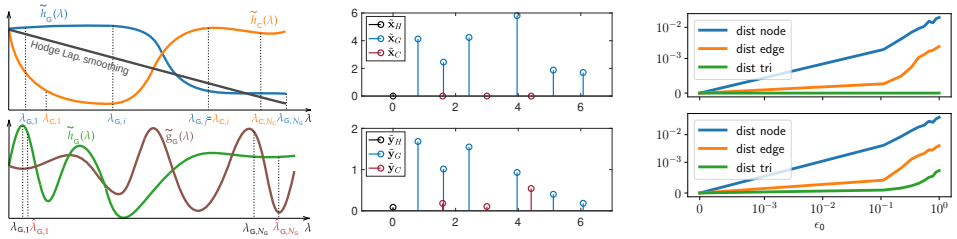
$$\tilde{\mathbf{h}}_{k,\mathrm{d}} = \textstyle\sum_{t=0}^{T_\mathrm{d}} w'_{k,\mathrm{d},t} \boldsymbol{\lambda}_{k,\mathrm{G}}^{\odot t} + w'_{k,\mathrm{u},0}\mathbf{1}, \text{ and } \tilde{\mathbf{h}}_{k,\mathrm{u}} = \textstyle\sum_{t=0}^{T_\mathrm{u}} w'_{k,\mathrm{u},t} \boldsymbol{\lambda}_{k,\mathrm{C}}^{\odot t} + w'_{k,\mathrm{d},0}\mathbf{1} \tag{3.12}$$

are the frequency response vectors of $\mathbf{H}_{k,\mathrm{d}}$ and $\mathbf{H}_{k,\mathrm{u}}$. The spectral relation in (3.10) shows that the gradient SFT $\tilde{\mathbf{x}}_{k,\mathrm{G}}$ is learned by a gradient response $\tilde{\mathbf{h}}_{k,\mathrm{G}}$, while the curl SFT $\tilde{\mathbf{x}}_{k,\mathrm{C}}$ is learned by a curl response $\tilde{\mathbf{h}}_{k,\mathrm{C}}$. The two learnable responses are independent and they only coincide at the trivial harmonic frequency, as shown by the two individual curves in Fig. 3.2a. Moreover, the lower and upper projections are independently learned by $\tilde{\mathbf{h}}_{k,\mathrm{d}}$ and $\tilde{\mathbf{h}}_{k,\mathrm{u}}$, respectively.

The elementwise nonlinearity induces an *information spillage* that one type of spectra could be spread over other types. As illustrated in Fig. 3.2b, the top figure shows the SFT of an input with only gradient components, and the bottom figure plots the SFT of $\sigma(\mathbf{y}_k)$, showing that it also contains information in harmonic or curl subspaces. This results from the nonlinearity, since applying a linear SCF leads to an output with only gradient components. In the following, we characterize the expressive power of SCCNN.

Proposition 3.11. *An SCCNN layer with inputs $\mathbf{x}_{k,\mathrm{d}}, \mathbf{x}_k, \mathbf{x}_{k,\mathrm{u}}$ is at most expressive as an MLP $\sigma(\mathbf{G}'_{k,\mathrm{d}}\mathbf{x}_{k,\mathrm{d}} + \mathbf{G}_k\mathbf{x}_k + \mathbf{G}'_{k,\mathrm{u}}\mathbf{x}_{k,\mathrm{u}})$ with $\mathbf{G}_k = \mathbf{G}_{k,\mathrm{d}} + \mathbf{G}_{k,\mathrm{u}}$ where $\mathbf{G}_{k,\mathrm{d}}$ and $\mathbf{G}'_{k,\mathrm{d}}$ are analytical matrix functions of $\mathbf{L}_{k,\mathrm{d}}$, while $\mathbf{G}_{k,\mathrm{u}}$ and $\mathbf{G}'_{k,\mathrm{u}}$ are analytical matrix functions of $\mathbf{L}_{k,\mathrm{u}}$. This expressivity can be achieved by setting $T_\mathrm{d} = T'_\mathrm{d} = n_{k,\mathrm{G}}$ and $T_\mathrm{u} = T'_\mathrm{u} = n_{k,\mathrm{C}}$ in (3.1) with $n_{k,\mathrm{G}}$ the number of distinct gradient frequencies and $n_{k,\mathrm{C}}$ the number of distinct curl frequencies.*

The proof follows from Cayley-Hamilton theorem [Horn & Johnson, 2012]. This expressive power can be better understood from the spectral perspective — The gradient SFT $\tilde{\mathbf{x}}_{k,\mathrm{G}}$ can be learned as expressive as by an analytical vector-valued function $\tilde{\mathbf{g}}_{k,\mathrm{G}}$, which collects the



(a) (b) (c)

Figure 3.2: (a) *(top)*: Independent gradient and curl learning responses. *(bottom)*: Stability-selectivity tradeoff of SCFs where $\tilde{h}_{\mathrm{G}}$ has better stability but smaller selectivity than $\tilde{g}_{\mathrm{G}}$. (b) Information spillage of nonlinearity. *(top)*: the SFT of an input with only gradient components. *(bottom)*: the SFT of the output shows that after applying a nonlinearity the output also contains information in non-gradient frequencies. (c) The distance between the perturbed outputs and true when node adjacencies are perturbed. *(top)*: $L = 1$, triangle output remains clean. *(bottom)*: $L = 2$, triangle output is perturbed.

eigenvalues of $\mathbf{G}_{k,\mathrm{d}}$ at gradient frequencies. The curl SFT $\tilde{\mathbf{x}}_{k,\mathrm{C}}$ can be learned as expressive as by another analytical vector-valued function $\tilde{\mathbf{g}}_{k,\mathrm{C}}$, which collects the eigenvalues of $\mathbf{G}_{k,\mathrm{u}}$ at curl frequencies. These two functions need only to coincide at the harmonic frequency. In addition, the SFTs of lower and upper projections can be learned as expressive as by two independent analytical vector-valued functions as well.

3.3.2 Hodge-aware learning

Given the expressive power in Proposition 3.11 and the spectral relation in (3.10), we show that SCCNN performs a Hodge-aware learning in the following sense, which comes with advantages over the existing approaches.

Theorem 3.12. *An SCCNN is Hodge-aware:*

1. *The SCF $\mathbf{H}_k$ is a Hodge-invariant learning operator. Specifically, three Hodge subspaces are invariant under $\mathbf{H}_k$;*
2. *The lower SCF $\mathbf{H}_{k,\mathrm{d}}$ and upper SCF $\mathbf{H}_{k,\mathrm{u}}$ are, respectively, gradient- and curl-invariant learning operators;*
3. *The learnings in the gradient and curl spaces are **independent**; and*
4. *the learnings in the gradient and curl spaces are **expressive** as in Proposition 3.11.*

This theorem shows that an SCCNN performs an expressive and independent learning in the gradient and curl subspaces from the three inputs while preserving the three subspaces to be invariant w.r.t its learnable SCFs. This allows for the *rational and effective learning* on SCs, as illustrated in Fig. 3.3, from the two aspects. These three-fold properties of an SCCNN, respectively, come from the convolutional architecture choice, the uncoupling of the lower and upper adjacencies, and the higher-order convolutions in the SCCNN.

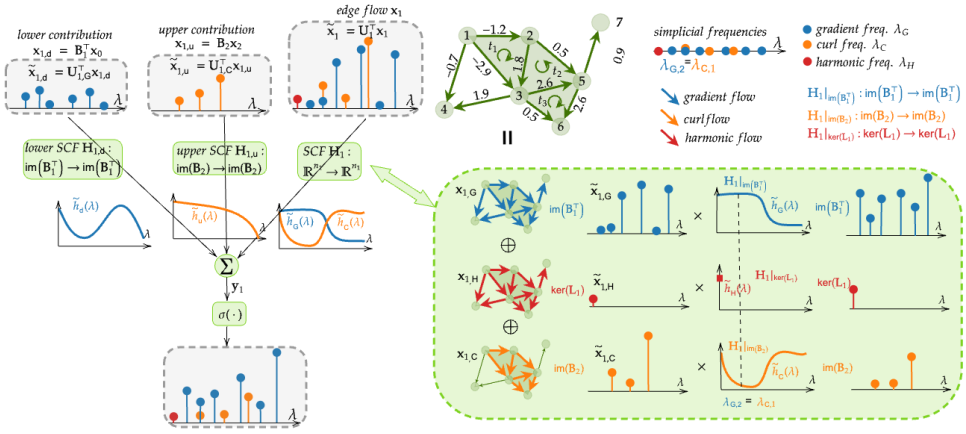


Figure 3.3: An illustration of the Hodge-aware learning of an SCCNN. We show that how an edge flow $\mathbf{x}_1$, together with the lower and upper projections $\mathbf{x}_{1,\mathrm{d}}$ and $\mathbf{x}_{1,\mathrm{u}}$, are transformed by an SCCNN in the spectral domain. The *implicit* operation $\mathbf{H}_1\mathbf{x}_1$ (in the dashed box on the right) reflects the Hodge-aware learning: 1) $\mathbf{H}_1$ is Hodge invariant: each component is learned within their own subspace, and $\mathbf{H}_1$ does not *mix up* the three subspaces; 2) The learning in the gradient and curl subspaces are independent where features at shared frequencies $\lambda_{G,2}$ and $\lambda_{C,1}$ can be separately learned; and 3) The learning operators are expressive in the sense that the spectral responses are as expressive as any analytical functions in the gradient and curl frequencies.

On the one hand, Proposition 3.10 shows that the operation of $\mathbf{H}_k$ on the simplicial signal space is equivalent to a summation of its restrictions $\mathbf{H}_k|_\square$ on three smaller subspaces $\square$. This Hodge-invariant nature of the learnable SCFs substantially shrinks the learning space of an SCCNN and allows for an effective learning. On the other hand, simplicial signals often present implicit or explicit properties that different Hodge subspaces can capture. For example, water flows, traffic flows, electrical currents [Grady & Polimeni, 2010; Jia et al., 2019] follow flow conservation, i.e., being div-free in the gradient space $\ker(\mathbf{B}_1)$, while exchange rates can be modelled as curl-free edge flows [Jiang et al., 2011]. Owing to the Hodge-invariance of $\mathbf{H}_1$ and its independent learning in the nontrivial subspaces, an SCCNN can capture such characteristics of real-world edge flows effectively. When it comes to regression tasks on SCs, an SCCNN can generate outputs respecting these physicial laws.

Remark 3.13 (Relation to message passing networks). Message-passing simplicial networks (MPSNs) [Bodnar et al., 2021b] using MLP to aggregate and update are non-Hodge-aware. Their learning functions pursue direct mappings between the much larger signal space $\mathbb{R}^{n_k}$, thus, requiring more training data for accurate learning, as well as a larger computational complexity. Moreover, MPSN does not preserve the Hodge subspaces, i.e., it is not Hodge-invariant. Thus, they might generate outputs with small losses (e.g., mean-squared-errors) in regression tasks, yet not respecting the physical laws being either div- or curl-free properties such as the above simplicial signals. We shall corroborate this in Appendix 3.F.

Remark 3.14 (Relation to other convolutional methods). While most convolutional networks on SCs use Hodge-invariant learning operators, they are not strictly Hodge-aware, resulting in practical limits. For example, Ebli et al. [2020] considered $\mathbf{H}_k = \sum_i w_i \mathbf{L}_k^i$, which

preserves the Hodge subspaces yet does not uncouple the lower and upper parts of $\mathbf{L}_k$. This makes it *strictly less expressive* and non-Hodge-aware. Consider two frequencies $\lambda_G = \lambda_C$ which share a common value but correspond to the gradient and curl subspaces, respectively. The simplicial signal components at these two frequencies are always learned in the same fashion, which induces contradicting issues when the underlying component in one subspace should be diminished while the one in the other subspace should be preserved. This underlines the importance of uncoupling the two adjacencies because the lower and upper Laplacians operate in different subspaces. Roddenberry et al. [2021] applied $\mathbf{H}_k$ with $T_{\mathrm{d}} = T_{\mathrm{u}} = 1$. Spatially, this limits the receptive field of each simplex to its direct neighbors. Spectrally, it leads to a linear learning frequency response. A similar treatment was considered in Bunch et al. [2020]; Yang et al. [2022c] which simply generalized the GCN without uncoupling the two adjacencies, and gave a limited low-pass linear spectral response, as shown in Fig. 3.2a and discussed in Section 3.2.3.

3.4 Stability Analysis

In this section, we investigate the stability of SCCNNs to domain perturbations on SCs, because in practice we often perform convolutional learning on weighted SCs to capture the strengths of simplicial adjacencies and incidences which may suffer perturbations.

3.4.1 SCCNN on weighted SCs

A weighted SC can be defined through specifying the weights of simplices. We give the definition of a commonly used weighted SC with weighted Hodge Laplacians in Grady & Polimeni [2010]; Horak & Jost [2013].

Definition 3.15 (Weighted SC and Hodge Laplacians)**.** In an oriented and weighted SC, we have diagonal weighting matrices $\mathbf{M}_k$ with $[\mathbf{M}]_{ii}$ measuring the weight of ith k-simplex. A weighted kth Hodge Laplacian is given by

$$\mathbf{L}_k = \mathbf{L}_{k,\mathrm{d}} + \mathbf{L}_{k,\mathrm{u}} = \mathbf{M}_k \mathbf{B}_k^\top \mathbf{M}_{k-1}^{-1} \mathbf{B}_k + \mathbf{B}_{k+1} \mathbf{M}_{k+1} \mathbf{B}_{k+1}^\top \mathbf{M}_k^{-1}. \tag{3.13}$$

where $\mathbf{L}_{k,\mathrm{d}}$ and $\mathbf{L}_{k,\mathrm{u}}$ are the weighted lower and upper Laplacians. A symmetric version follows $\mathbf{L}_k^s = \mathbf{M}_k^{-1/2} \mathbf{L}_k \mathbf{M}_k^{1/2}$, and likewise, we have $\mathbf{L}_{k,\mathrm{d}}^s = \mathbf{M}_k^{1/2} \mathbf{B}_k^\top \mathbf{M}_{k-1}^{-1} \mathbf{B}_k \mathbf{M}_k^{1/2}$ and $\mathbf{L}_{k,\mathrm{u}}^s = \mathbf{M}_k^{-1/2} \mathbf{B}_{k+1} \mathbf{M}_{k+1} \mathbf{B}_{k+1}^\top \mathbf{M}_k^{-1/2}$, with the weighted incidence matrix $\mathbf{M}_{k-1}^{-1/2} \mathbf{B}_k \mathbf{M}_k^{1/2}$ [Guglielmi et al., 2023; Horak & Jost, 2013; Schaub et al., 2020].

SCCNNs in weighted SC. The SCCNN layer defined in a weighted SC is of form

$$\mathbf{x}_k^l = \sigma(\mathbf{H}_{k,\mathrm{d}}^l \mathbf{R}_{k,\mathrm{d}} \mathbf{x}_{k-1}^{l-1} + \mathbf{H}_k^l \mathbf{x}_k^{l-1} + \mathbf{H}_{k,\mathrm{u}}^l \mathbf{R}_{k,\mathrm{u}} \mathbf{x}_{k+1}^{l-1}) \tag{3.14}$$

where the three SCFs are defined based on the weighted Laplacians (3.13), and the lower and upper contributions $\mathbf{x}_{k,\mathrm{d}}^l$ and $\mathbf{x}_{k,\mathrm{u}}^l$ are obtained via projection matrices $\mathbf{R}_{k,\mathrm{d}} \in \mathbb{R}^{n_k \times n_{k-1}}$ and $\mathbf{R}_{k,\mathrm{u}} \in \mathbb{R}^{n_k \times n_{k+1}}$, instead of $\mathbf{B}_k^\top$ and $\mathbf{B}_{k+1}$. For example, Bunch et al. [2020] considered $\mathbf{R}_{1,\mathrm{d}} = \mathbf{M}_1 \mathbf{B}_1^\top \mathbf{M}_0^{-1}$ and $\mathbf{R}_{1,\mathrm{u}} = \mathbf{B}_2 \mathbf{M}_2$.

3.4.2 Stability of SCCNNs to Domain Perturbations

To highlight the need for a stability analysis, note that, on the one hand, we may lack the true underlying topologies in SCs as they are often estimated from noisy data; and we may undergo adversarial attacks on the topologies. On the other hand, we want to characterize the stability-selectivity tradeoff of SCCNN, in analogy to the study for CNNs [Bietti & Mairal, 2017; Bruna & Mallat, 2013; Qiu et al., 2018] and GNNs [Gama et al., 2019b; 2020a; Kenlay et al., 2021; Parada-Mayorga et al., 2022].

This motivates us to investigate that *how far are the outputs of an SCCNN before and after perturbations are applied to SCs?* We consider the following relative perturbation model, generalizing the graph perturbation model in Gama et al. [2019b]

Definition 3.16 (Relative perturbation)**.** Consider some perturbation matrix of an appropriate dimension. For the weighted Hodge Laplacian $\mathbf{L}_{k,\mathrm{d}}$, its relative perturbed version is $\widehat{\mathbf{L}}_{k,\mathrm{d}} = \mathbf{L}_{k,\mathrm{d}} + \mathbf{E}_{k,\mathrm{d}}\mathbf{L}_{k,\mathrm{d}} + \mathbf{L}_{k,\mathrm{d}}\mathbf{E}_{k,\mathrm{d}}$ with perturbation $\mathbf{E}_{k,\mathrm{d}}$; likewise for $\widehat{\mathbf{L}}_{k,\mathrm{u}}$ by $\mathbf{E}_{k,\mathrm{u}}$. For the weighted incidence matrix $\mathbf{R}_{k,\mathrm{d}}$, its relative perturbed version is $\widehat{\mathbf{R}}_{k,\mathrm{d}} = \mathbf{R}_{k,\mathrm{d}} + \mathbf{J}_{k,\mathrm{d}}\mathbf{R}_{k,\mathrm{d}}$ with perturbation $\mathbf{J}_{k,\mathrm{d}}$; likewise for $\widehat{\mathbf{R}}_{k,\mathrm{u}}$ by $\mathbf{J}_{k,\mathrm{u}}$.

This models the *perturbations* on the strengths of adjacent and incident relations, e.g., a large weight is misused when two edges are weakly or not adjacent, or data on a node is projected on an edge which is yet not incident to it. Moreover, this quantifies the relative perturbations with respect to the local simplicial topology in the sense that weaker connections in an SC are deviated by perturbations proportionally less than stronger connections. We further define the integral Lipschitz property of spectral filters to measure the variability of spectral response functions of $\mathbf{H}_k$.

Definition 3.17 (Intergral Lipschitz SCF)**.** An SCF $\mathbf{H}_k$ is *integral Lipschitz* with constants $c_{k,\mathrm{d}}$, $c_{k,\mathrm{u}} \geq 0$ if the derivatives of its spectral response functions $\tilde{h}_{k,\mathrm{G}}(\lambda)$ and $\tilde{h}_{k,\mathrm{C}}(\lambda)$ follow that $|\lambda \tilde{h}'_{k,\mathrm{G}}(\lambda)| \leq c_{k,\mathrm{d}}$ and $|\lambda \tilde{h}'_{k,\mathrm{C}}(\lambda)| \leq c_{k,\mathrm{u}}$.

This property provides a stability-selectivity tradeoff of SCFs independently in the gradient and curl frequencies. A spectral response can have both a good selectivity and stability in small frequencies (a large $|\tilde{h}'_{k,\cdot}|$ for $\lambda \to 0$), while it tends to be flat for having better stability at the cost of selectivity (a small variability for large λ) in large frequencies, as shown in Fig. 3.2a. As of the polynomial nature of responses, all SCFs of SCCNN are integral Lipschitz. We also denote the integral Lipschitz constant for the lower SCFs $\mathbf{H}_{k,\mathrm{d}}$ by $c_{k,\mathrm{d}}$ and for the upper SCFs $\mathbf{H}_{k,\mathrm{u}}$ by $c_{k,\mathrm{u}}$ without loss of generality.

Under the following assumptions, we now characterize the stability bound of SCCNN.

Assumption 3.18. *The perturbations are small such that* $\|\mathbf{E}_{k,\mathrm{d}}\|_2 \leq \epsilon_{k,\mathrm{d}}, \|\mathbf{J}_{k,\mathrm{d}}\|_2 \leq \varepsilon_{k,\mathrm{d}}$ *and* $\|\mathbf{E}_{k,\mathrm{u}}\|_2 \leq \epsilon_{k,\mathrm{u}}, \|\mathbf{J}_{k,\mathrm{u}}\|_2 \leq \varepsilon_{k,\mathrm{u}}$*, where* $\|\mathbf{A}\|_2 = \max_{|\mathbf{x}|_1=1} \|\mathbf{A}\mathbf{x}\|_2$ *is the operator norm (spectral radius) of a matrix* $\mathbf{A}$*.*

Assumption 3.19. *The SCFs* $\mathbf{H}_k$ *of an SCCNN have a normalized bounded frequency response (for simplicity), likewise for* $\mathbf{H}_{k,\mathrm{d}}$ *and* $\mathbf{H}_{k,\mathrm{u}}$*.*

Assumption 3.20. *The lower and upper projections are finite such that* $\|\mathbf{R}_{k,\mathrm{d}}\|_2 \leq r_{k,\mathrm{d}}$ *and* $\|\mathbf{R}_{k,\mathrm{u}}\|_2 \leq r_{k,\mathrm{u}}$*.*

Assumption 3.21. *The nonlinearity $\sigma(\cdot)$, e.g., $\mathrm{ReLu}, \tanh, \mathrm{sigmoid}$, is c_σ-Lipschitz with $c_\sigma \geq 0$.*

Assumption 3.22. *The initial inputs $\mathbf{x}_k^0$, for all k, are finite, such that $\|\mathbf{x}_k^0\|_2 \leq [\boldsymbol{\beta}]_k$. We collect them in $\boldsymbol{\beta} = [\beta_0, \ldots, \beta_K]^\top$.*

Theorem 3.23. *Let $\mathbf{x}_k^L$ be the k-simplicial signal output of an L-layer SCCNN on a weighted SC. Let $\hat{\mathbf{x}}_k^L$ be the output of the same SCCNN but on a relatively perturbed SC. Define $\delta_{k,\mathrm{d}} = (\|\mathbf{V}_{k,\mathrm{d}} - \mathbf{U}_k\| + 1)^2 - 1$ and $\delta_{k,\mathrm{u}} = (\|\mathbf{V}_{k,\mathrm{u}} - \mathbf{U}_k\| + 1)^2 - 1$, with $\mathbf{V}_{k,\mathrm{d}}$ and $\mathbf{V}_{k,\mathrm{u}}$ the eigenvectors of $\mathbf{E}_{k,\mathrm{d}}$ and $\mathbf{E}_{k,\mathrm{u}}$, which measure the eigenvector misalignments between the perturbations and Laplacians. Under Assumptions 3.18 to 3.22, the Euclidean distance between the two outputs is finite and upper-bounded*

$$\|\hat{\mathbf{x}}_k^L - \mathbf{x}_k^L\|_2 \leq [\mathbf{d}]_k \text{ with } \mathbf{d} = c_\sigma^L \sum_{l=1}^{L} \widehat{\mathbf{Z}}^{l-1} \mathbf{T} \mathbf{Z}^{L-l} \boldsymbol{\beta}, \tag{3.15}$$

where for $K = 2$,

$$\mathbf{T} = \begin{bmatrix} t_0 & t_{0,\mathrm{u}} & \\ t_{1,\mathrm{d}} & t_1 & t_{1,\mathrm{u}} \\ & t_{2,\mathrm{d}} & t_2 \end{bmatrix}, \mathbf{Z} = \begin{bmatrix} 1 & r_{0,\mathrm{u}} & \\ r_{1,\mathrm{d}} & 1 & r_{1,\mathrm{u}} \\ & r_{2,\mathrm{d}} & 1 \end{bmatrix}, \widehat{\mathbf{Z}} = \begin{bmatrix} 1 & \hat{r}_{0,\mathrm{u}} & \\ \hat{r}_{1,\mathrm{d}} & 1 & \hat{r}_{1,\mathrm{u}} \\ & \hat{r}_{2,\mathrm{d}} & 1 \end{bmatrix}, \tag{3.16}$$

with $\hat{r}_{k,\mathrm{d}} = r_{k,\mathrm{d}}(1 + \varepsilon_{k,\mathrm{d}})$ and $\hat{r}_{k,\mathrm{u}} = r_{k,\mathrm{u}}(1 + \varepsilon_{k,\mathrm{u}})$. Notice that $\mathbf{T}, \mathbf{Z}$ and $\widehat{\mathbf{Z}}$ are tridiagonal and follow a similar structure for a general K. The diagonal entries of $\mathbf{T}$ are $t_k = c_{k,\mathrm{d}} \Delta_{k,\mathrm{d}} \epsilon_{k,\mathrm{d}} + c_{k,\mathrm{u}} \Delta_{k,\mathrm{u}} \epsilon_{k,\mathrm{u}}$. The off-diagonal entries are $t_{k,\mathrm{d}} = r_{k,\mathrm{d}} \varepsilon_{k,\mathrm{d}}$ $+ c_{k,\mathrm{d}} \Delta_{k,\mathrm{d}} \epsilon_{k,\mathrm{d}} r_{k,\mathrm{d}}$ and $t_{k,\mathrm{u}} = r_{k,\mathrm{u}} \varepsilon_{k,\mathrm{u}} + c_{k,\mathrm{u}} \Delta_{k,\mathrm{u}} \epsilon_{k,\mathrm{u}} r_{k,\mathrm{u}}$, where $\Delta_{k,\mathrm{d}} = 2(1 + \delta_{k,\mathrm{d}}) \sqrt{n_k}$ and $\Delta_{k,\mathrm{u}} = 2(1 + \delta_{k,\mathrm{u}}) \sqrt{n_k}$.

We refer to Appendix 3.E.1 for a two-step proof. This result bounds the outputs of an SCCNN on all simplicial levels, showing that they are stable to small perturbations on the simplicial adjacencies and incidences. Specifically, we make two observations from the complicated expression. First, the stability bound depends on i) the degree of perturbations including their magnitudes $\epsilon_{k,\cdot}$ and $\varepsilon_{k,\cdot}$, and the eigenspace misalignment $\Delta_{k,\cdot}$; ii) the integral Lipschitz properties $c_{k,\cdot}$ of SCFs; and, iii) the degree of projections $r_{k,\cdot}$. Second, the stability of the k-output depends on the factors related to not only k-simplices, but also simplices of adjacent orders due to inter-simplicial couplings. For example, when $L = 1$, the node output bound d_0 is affected by factors in the node space, as well as the edge space factored by the projection degree. As the layer deepens, this mutual dependence expands further. When $L = 2$, the factors in the triangle space also affect the stability of the node output d_0, as we observe in Fig. 3.2c.

More importantly, this stability bound provides intuitive practical implications for convolutional learning on SCs. While inter-simplicial couplings may be beneficial, SCCNN becomes less stable as the number of layers increases due to the mutual dependence between outputs on different simplicial levels. Thus, to maintain the expressive power, we expect to use higher-order SCFs in exchange for shallow layers. This yet does not harm the stability in the following two aspects:

- First, high-frequency components can be spread over the low frequencies due to the information spillage of nonlinearity [cf. Fig. 3.2b] where the spectral responses are more selective and have better stability. If the signal has large high gradient frequency components, we need the SCCNN to be selective in high gradient frequencies. However, to guarantee the stability, the frequency response should be smooth (less selective) in these frequencies, as illustrated by $\tilde{h}_{\mathrm{G}}$ in Fig. 3.2a *(bottom)*. This selectivity-stability tradeoff can be mitigated by the nonlinearity — the information in high gradient frequencies could spill over in lower frequencies, where the spectral responses are more selective and have better discriminating ability.

- Second, higher-order SCFs are easier to be learned with smaller integral Lipschitz constants than lower-order ones due to the increased degree of freedom, thus, leading to an increased stability. This can be easily seen by comparing one-order and two-order cases. We experimentally investigate this in Section 3.5.4. Moreover, we introduce the following regularizer to the loss function during training so to promote the integral Lipschitz property.

$$\begin{aligned} r_{\mathrm{IL}} &= \|\lambda_{k,\mathrm{G}}\tilde{h}'_{k,\mathrm{G}}(\lambda_{k,\mathrm{G}})\| + \|\lambda_{k,\mathrm{C}}\tilde{h}'_{k,\mathrm{C}}(\lambda_{k,\mathrm{C}})\| \\ &= \left\| \sum_{t=0}^{T_{\mathrm{d}}} t w_{k,\mathrm{d},t} \lambda_{k,\mathrm{G}}^{t} \right\| + \left\| \sum_{t=0}^{T_{\mathrm{u}}} t w_{k,\mathrm{u},t} \lambda_{k,\mathrm{C}}^{t} \right\| \end{aligned} \tag{3.17}$$

for $\lambda_{k,\mathrm{G}} \in \{\lambda_{k,\mathrm{G},i}\}_{i=1}^{n_{k,\mathrm{G}}}$ and $\lambda_{k,\mathrm{C}} \in \{\lambda_{k,\mathrm{C},i}\}_{i=1}^{n_{k,\mathrm{C}}}$, which are the gradient and curl frequencies. To avoid computing the eigendecomposition of the Hodge Laplacian, we can approximate the true frequencies by sampling certain number of points in the frequency band $(0, \lambda_{k,\mathrm{G,m}}]$ and $(0, \lambda_{k,\mathrm{C,m}}]$ where the maximal gradient and curl frequencies can be computed by efficient algorithms, e.g., power iteration.

3.5 Experiments

The goal of this section is to answer the following four research questions with experiments on various simplicial-level regression and classification tasks:

RQ 1 What are the effects of the three principles of SCCNN, i.e., uncoupling the lower and upper parts of Hodge Laplacians (P1), the inter-simplicial couplings (P2), and higher-order convolutions (P3)?

RQ 2 How do the uncoupling of the lower and upper parts of Hodge Laplacians and the inter-simplicial couplings affect the simplicial oversmoothing?

RQ 3 How does the Hodge-aware property of SCCNN play a role in different tasks on SCs, compared to non-Hodge-aware methods?

RQ 4 How do different factors affect the stability of SCCNN, and how can we maintain the stability while keeping the expressive power?

For comparison, we consider the following learning methods on single-level simplices:

Table 3.1: Forex results (nmse|total arbitrage, ↓).

Methods	Random Noise	Curl Noise	Interpolation
Input	$0.119_{\pm 0.004}$|$29.19_{\pm 0.874}$	$0.552_{\pm 0.027}$|$122.4_{\pm 5.90}$	$0.717_{\pm .030}$|$106.4_{\pm 0.902}$
Baseline (ℓ_2 regularization)	$0.036_{\pm 0.005}$|$2.29_{\pm 0.079}$	$0.050_{\pm 0.002}$|$11.12_{\pm 0.537}$	$0.534_{\pm 0.043}$|$9.67_{\pm 0.082}$
SNN	$0.110_{\pm 0.005}$|$23.24_{\pm 1.03}$	$0.446_{\pm 0.017}$|$86.95_{\pm 2.20}$	$0.702_{\pm 0.033}$|$104.74_{\pm 1.04}$
PSNN	$0.008_{\pm 0.001}$|$0.984_{\pm 0.170}$	$0.000_{\pm 0.000}$|$0.000_{\pm 0.000}$	$0.009_{\pm 0.001}$|$1.13_{\pm 0.329}$
MPSN	$0.039_{\pm 0.004}$|$7.74_{\pm 0.88}$	$0.076_{\pm 0.012}$|$14.92_{\pm 2.49}$	$0.117_{\pm 0.063}$|$23.15_{\pm 11.7}$
SCCNN, id	$0.027_{\pm 0.005}$|$0.000_{\pm 0.000}$	$0.000_{\pm 0.000}$|$0.000_{\pm 0.000}$	$0.265_{\pm 0.036}$|$0.000_{\pm 0.000}$
SCCNN, tanh	$\mathbf{0.002_{\pm 0.000}}$|$\mathbf{0.325_{\pm 0.082}}$	$0.000_{\pm 0.000}$|$0.003_{\pm 0.003}$	$\mathbf{0.003_{\pm 0.002}}$|$\mathbf{0.279_{\pm 0.151}}$

- simplicial neural network (SNN) [Ebli et al., 2020], which does not respect P1 and P2 and is non-Hodge-aware;
- principled simplicial neural network (PSNN) [Roddenberry et al., 2021], which does not respect P2 and P3 and is non-Hodge-aware;
- simplicial convolutional neural networks (SCNN)[1] [Yang et al., 2022a], which does not respect P2 but is Hodge-aware;

and the following learning methods on simplicial complexes:

- Bunch [Bunch et al., 2020], which does not respect P1 and P2 and is non-Hodge-aware;
- MPSN [Bodnar et al., 2021b], which is based on message-passing and not Hodge-aware.

We also considered the MLP and standard GNN [Defferrard et al., 2016] as baselines to highlight the effect of SC topology on simplicial-level tasks. We refer to Appendix 3.B for the detailed comparisons between these methods and SCCNN, as well as to Appendix 3.F for the full experimental details.

3.5.1 Foreign currency exchange (RQs 1, 3)

In forex problems, to build a fair market, the *arbitray-free* condition implies that for any currencies i, j, k, it follows that $r^{i/j} r^{j/k} = r^{i/k}$ where $r^{i/j}$ is the exchange rate between i and j. That is, the exchange path $i \to j \to k$ provides no profit or loss over a direct exchange $i \to k$. Following Jiang et al. [2011], we model exchange rates as edge flows in an SC of order two, specifically, via $[\mathbf{x}_1]_{[i,j]} = \log(r^{i/j})$. This conveniently translates the arbitrage-free condition into $\mathbf{x}_1$ being curl-free, i.e., $[\mathbf{x}_1]_{[i,j]} + [\mathbf{x}_1]_{[j,k]} - [\mathbf{x}_1]_{[i,k]} = 0$ in any triangle $[i, j, k]$. We consider a real-world forex market at three timestamps, which contains certain degree of arbitrage [Jia et al., 2019; Yang et al., 2024]. We focus on recovering a fair market in two scenarios, first, from noisy exchange rates where random noise and noise only in the curl space modelling random arbitrage ("curl noise") are added, and second, when only 50% of the total rates are observed. To evaluate the performance, we measure

[1] Note that the difference between SCCNN and SCNN lies in that the latter does not include the inter-layer projections, as detailed in Appendix 3.B thus, we refer to our method, simplicial complex CNN.

Table 3.2: Simplex prediction (AUC, ↑) .

Methods	2-simplex	3-simplex
Mean	62.8±2.7	63.6±1.6
MLP	68.5±1.6	69.0±2.2
GNN	93.9±1.0	96.6±0.5
SNN	92.0±1.8	95.1±1.2
PSNN	95.6±1.3	98.1±0.5
SCNN	96.5±1.5	98.3±0.4
Bunch	98.3±0.5	98.5±0.5
MPSN	98.1±0.5	99.2±0.3
SCCNN	**98.7±0.5**	**99.4±0.3**

Table 3.3: Ablation study on SCCNN and the hyperparameters for the best results.

Missing component	2-simplex	Hyper Params.
–	98.7±0.5	$L = 2, T = 2$
Edge-to-Node	93.9±1.0	$L = 5, T = 2$
Node-to-Node	98.7±0.4	$L = 4, T = 2$
Edge-to-Edge	98.5±1.0	$L = 3, T = 2$
Node-to-Edge	98.8±0.3	$L = 4, T = 2$
Node input	98.2±0.5	$L = 2, T = 4$
Edge input	98.1±0.4	$L = 2, T = 3$

the normalized mean squared error (nmse) and total arbitrage (total curl), both equally important for achieving a fair market.

From Table 3.1, we make the following observations on the impacts of P1 and P3, as well as the Hodge-awareness.

1) MPSN performs poorly at this task: although it reduces the nmse, it outputs unfair rates with large arbitrage, against the forex principle, because it is not Hodge-aware and unable to capture the arbitrage-free property with small amount of data (cf. Remark 3.13).
2) SNN performs poorly as well: as discussed in Remark 3.14, it restricts the gradient and curl spaces to be always learned in the same fashion and makes it impossible to perform disjoint learning in two subspaces. However, since there are eigenvalues which share a common value but live in different subspaces in this SC, it requires preserving the gradient component while removing the curl one here.
3) PSNN can reconstruct relatively fair forex rates with small nmse. The reconstruction from curl noise is perfect, while in the other two cases, the nmse and arbitrage are three times larger than the proposed SCCNN due to the limited expressivity of linear learning responses.
4) SCCNN performs the best in both reducing the total error and the total arbitrage, ultimately, corroborating the impact of performing Hodge-aware learning.

We notice that with an identity activation function ($\sigma = \mathrm{id}$), the arbitrage-free rule is fully learned by an SCCNN. However, it has relatively large errors in the random noise and interpolation cases due to its limited linear expressive power. With a nonlinearity $\sigma = \tanh$, an SCCNN can tackle these more challenging cases, finding a good compromise between overall errors and data characteristics.

3.5.2 Simplicial oversmoothing analysis (RQ 2)

We use simplicial shifting layers (i.e., (3.6) composed with $\sigma = \tanh$) to illustrate the evolution of Dirichlet energies of the outputs on nodes, edges and triangles in an SC of

order two with respect to the number of layers. The corresponding inputs are randomly sampled from a uniform distribution $\mathcal{U}([-5,5])$. Table 3.4 (the dashed lines) shows that simply generalizing the GCN on SCs as in Bunch method could lead to oversmoothing on simplices of all orders. This aligns with our theoretical results in Section 3.2.3. However, uncoupling the lower and upper parts of $\mathbf{L}_1$ (e.g., by setting $\gamma = 2$ in (3.6)) could mitigate the oversmoothing on edges, as shown by the dotted line. Lastly, when we account for the inter-simplicial coupling, as shown by the solid lines (where we applied (3.7)), it could almost prevent the oversmoothing, since it provides energy sources. We refer to Appendix 3.F.1 for other results.

3.5.3 Simplex prediction (RQs 1, 3-4)

We consider the prediction task of 2- and 3-simplices which extends the link (1-simplex) prediction in graphs. Our approach is to first learn the representations of lower-order simplices and then use an MLP with their concatenation as inputs to identify if a simplex is closed or open, which generalizes the link prediction method of Zhang & Chen [2018]. Considering a coauthorship dataset [Ammar et al., 2018], we built an SC following Ebli et al. [2020] where nodes represent authors and $(k-1)$-simplices thus represent the collaborations of k-authors. The input simplicial signals are the numbers of citations, e.g., $\mathbf{x}_1$ and $\mathbf{x}_2$ are those of dyadic and triadic collaborations. Thus, 2-simplex (3-simplex) prediction amounts to predicting triadic (tetradic) collaborations. We evaluate the AUC (area under the curve) performance.

From Table 3.2, we make three observations on the effect of the three key principles. 1) SCCNN, MPSN and Bunch methods outperform the ones without inter-simplicial couplings. This highlights that accounting for contributions from faces and cofaces increases the representation power of the network. 2) SCNN performs better than an SNN, which shows that uncoupling the lower and upper parts in $\mathbf{L}_k$ improves the representation learning. 3) SCCNN performs better than Bunch (similarly, SCNN better than PSNN), showing that higher-order convolution further improves predictions. 4) While MPSN performs similar to SCCNN, it has three times more parameters than an SCCNN (Appendix 3.F.3) under the settings of the best results.

Ablation study. We then perform an ablation study to investigate the roles of different components in an SCCNN. As reported in Table 3.3, we remove certain simplicial relations in the SCCNN and evaluate the prediction performance. Without the edge-to-node incidence, when inputting the node features to the MLP predictor, it is equivalent to a GNN, which has a poor performance. When removing other adjacencies or incidences, the best performance remains similar but with an increased model complexity (more layers required). This however is not preferred, because the stability decreases as the architecture deepens and the model gets influenced by factors in other simplicial spaces, as discussed in Section 3.4 and shown in Fig. 3.2c. We also consider the case with limited input where the input on nodes or on edges is missing. The best performance of an SCCNN only slightly drops with an increase of the convolution order.

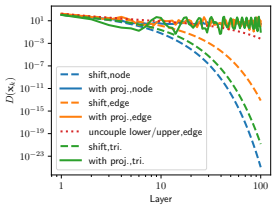


Table 3.4: Simplicial oversmoothing.

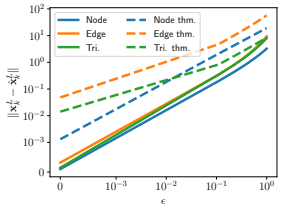


Figure 3.4: Stability bound versus perturbations on simplicial adjacencies and incidences.

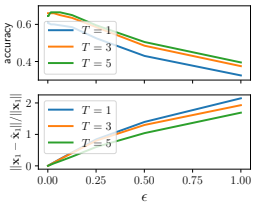


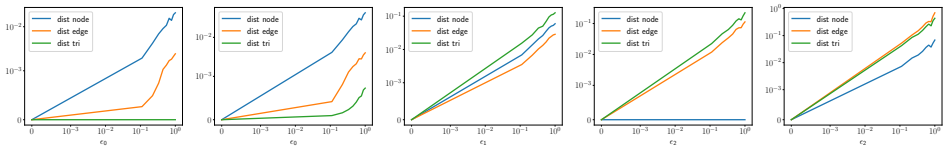
Figure 3.5: Stability and accuracy versus T.

Stability analysis (RQ 4)

Stability bounds. To investigate the stability bound in (3.15), we add perturbations to relatively shift the eigenvalues of the Hodge Laplacians and the singular values of the projection matrices by $\epsilon \in [0, 1]$ (cf. Assumption 3.18). We compare the bound in (3.15) to the experimental ℓ_2 distance on each simplex level. As shown in Fig. 3.4 where the dashed lines are the theoretical stability bounds whereas the solid ones are the experimental stability bounds, we see the bounds become tighter as perturbation increases.

Stability dependence across simplices. For 2-simplex prediction of $K = 2$, we measure the distance between the simplicial outputs of SCCNN with and without perturbations on nodes, edges, and triangles, i.e., $\|\mathbf{x}_k^L - \hat{\mathbf{x}}_k^L\|/\|\mathbf{x}_k^L\|$, for $k = 0, 1, 2$. Fig. 3.6 shows that overall the stabilities of different simplicial outputs are dependent on each other. Specifically, we see that the triangle output is not influenced by the perturbation on node weights until $L = 2$; likewise, the node output is not influenced by the perturbations on triangle weights when $L = 1$. Also, perturbations on the edge weights will perturbe the outputs on nodes, edges, triangles when $L = 1$. This corroborates our discussions in Section 3.4.

Effect of number of simplices We observe that the same degree of perturbations added to different simplices causes different degrees of instability, owing to the number n_k of k-simplices in (3.15). Since $n_0 < n_1 < n_2$, the perturbations on node weights cause less instability than those on edge and triangle weights.



(a) Node pert. $L = 1$ (b) Node pert. $L = 2$ (c) Edge pert. $L = 1$ (d) Tri. pert. $L = 1$ (e) Tri. pert. $L = 2$

Figure 3.6: The relative difference of SCCNN outputs on simplices of different orders when perturbations are applied to only nodes, edges and triangles and the number of layers varies.

Effect of number of layers. As the number of layers increases, Fig. 3.6 also shows that the stability of SCCNN degrades, which corresponds to our analysis of using shallow layers.

Table 3.5: Trajectory prediction (accuracy, ↑).

Methods	Synthetic trajectories	Ocean drifters
SNN	65.5±2.4	52.5±6.0
PSNN	63.1±3.1	49.0±8.0
SCNN	**67.7±1.7**	53.0±7.8
Bunch	62.3±4.0	46.0±6.2
SCCNN	65.2±4.1	**54.5±7.9**

3.5.4 Trajectory prediction (RQ 1, 4)

We consider the task of predicting trajectories in a synthetic SC and ocean drifters from Schaub et al. [2020], following Roddenberry et al. [2021]. From Table 3.5, we first observe that the SCCNN and Bunch methods do not always perform better than those without inter-simplicial couplings. This is because zero inputs are applied on nodes and triangles following Roddenberry et al. [2021], which makes inter-simplicial couplings inconsequential. Secondly, an SCCNN performs better than Bunch on average, and SCNN better than PSNN, showing the advantages of higher-order convolutions. Note that the prediction here aims to find the best candidate from the neighborhood of the end node, which depends on the node degree. Since the average node degree of the synthetic SC is 5.24 and that in the ocean drifter data is 4.81, a random guess has around 20% accuracy. The high standard derivations may result from the limited ocean drifter data size.

Stability analysis (RQ 4)

Differently from Section 3.5.3, we further investigate the stability in terms of the integral Lipschitz properties and convolutional orders. We consider SCNNs [Yang et al., 2022a] with orders $T_{\mathrm{d}} = T_{\mathrm{u}} \in \{1, 3, 5\}$ and train them with regularizations on the integral Lipschitz constants. As shown in Fig. 3.5, the higher-order case has better stability (smaller ℓ_2 distance between the outputs without and with perturbations) and consistent better accuracy, compared to the lower-order case. This is because the additional flexibility in the higher-order case allows the filters to have better intergral Lipschitz properties and thus better stability, while maintaining the accuracy. We refer to Appendix 3.F.4 for a detailed design of the regularizations, as well as more in-depth experimental analysis.

3.6 Related Works

Our work is mainly related to learning methods on SCs. Roddenberry & Segarra [2019] first used $\mathbf{L}_{1,\mathrm{d}}$ to build neural networks on edges in a graph setting without the upper edge adjacency. Ebli et al. [2020] then generalized convolutional GNNs [Defferrard et al., 2016; Kipf & Welling, 2017] to simplices by using the Hodge Laplacian. Roddenberry et al. [2021]; Yang et al. [2022a] instead uncoupled the lower and upper Laplacians to perform one- and multi-order convolutions, to which Giusti et al. [2022]; Goh et al. [2022]; Lee et al. [2022] added attention schemes. Keros et al. [2022] considered a variant of

Table 3.6: Comparisons between SCCNN and other architectures on if they respect the three principles.

Methods	Scheme	P1	P2	P3
MPSN [Bodnar et al., 2021b]	message-passing	yes	yes	no, only direct neighborhoods
Eq. (11) of MPSN, or [Bunch et al., 2020]	convolutional	no	yes	no, only direct neighborhoods
Eq. (27) of MPSN	convolutional	yes	yes	no, only direct neighborhoods
SNN [Ebli et al., 2020]	convolutional	no	no	yes
PSNN [Roddenberry et al., 2021]	convolutional	yes	no	no, only direct neighborhoods
SCNN [Yang et al., 2022a]	convolutional	yes	no	yes
SCCNN	convolutional	yes	yes	yes

Roddenberry et al. [2021] to identify topological "holes" and Chen et al. [2022d] combined shifting on nodes and edges for link prediction. These works learned within a simplicial level and did not consider the incidence relations (inter-simplicial couplings) in SCs, which was included by Bunch et al. [2020]; Yang et al. [2022c]. These works considered convolutional-type methods, which can be subsumed by SCCNNs. Meanwhile, Bodnar et al. [2021b]; Hajij et al. [2021] generalized the message passing on graphs [Xu et al., 2018a] to SCs, relying on both adjacencies and incidences. Most of these works focused on extending GNNs to SCs by varying the information propagation on SCs with limited theoretical insights into their components. Among them, Roddenberry et al. [2021] discussed the equivariance of PSNN to permutation and orientation, which SCCNNs admit as well. Bodnar et al. [2021b] studied the messgae-passing on SCs in terms of WL test of SCs built by completing cliques in a graph. The more closely related work is [Yang et al., 2022a], which gave only a spectral formulation based on SCFs but not SCCNNs. We refer to Table 3.6 for a comparion between these architectures, as well as to Besta et al. [2024]; Papamarkou et al. [2024] for an overview of the current progress on learning on SCs.

3.7 Discussion and Conclusion

3.7.1 Discussion and Conclusion

In our opinion, the advantage of SCs is not only about them being able to model higher-order network structures, but also support simplicial data, which can be both human-generated data like coauthorship, and physical data like flow-type data. This is why we approached the analysis from the perspectives of both simplicial structures and the simplicial data, i.e., the Hodge theory and spectral simplicial theory [Barbarossa & Sardellitti, 2020; Govek et al., 2018; Hodge, 1989; Lim, 2020; Steenbergen, 2013; Yang et al., 2021; 2022a]. We provided insights into why the three principles (P1-P3) are needed and how they can guide the effective and rational learning from simplicial data. As we have practically found, SCCNNs perform well in applications where data exhibits properties characterized by the Hodge decomposition due to the Hodge-awareness, while non-Hodge-aware learners fail at giving rational results. In cases where data does not possess such properties, SCCNNs have better or comparable performance than the ones which violate or do not respect the three principles.

Concurrently, there are works on more general cell complexes, e.g., [Bodnar et al., 2021a; Hajij et al., 2020; 2022; Roddenberry et al., 2022; Sardellitti et al., 2021], where 2-cells inlcude not only triangles, but also general polygon faces. We focus on SCs because a regular cell complex can be subdivided into an SC [Grady & Polimeni, 2010; Lundell et al., 1969] to which the analysis in this chapter applies, or we can generalize our analysis by allowing $\mathbf{B}_2$ to include 2-cells. This is however informal and does not exploit the power of cell complexes, which relies on cellular sheaves, as studied in [Bodnar et al., 2022; Hansen & Ghrist, 2019].

We proposed three principles (P1-P3) for convolutional learning on SCs, summarized in a general architecture, SCCNNs. Our analysis showed this architecture, guided by the three principles, demonstrates an awareness of the Hodge decomposition and performs rational, effective and expressive learning from simplicial data. Furthermore, our study reveals that SCCNNs exhibit stability and robustness against perturbations in the strengths of simplicial connections. Experimental results validate the benefits of respecting the three principles and the Hodge-awareness, as well as the stability results. Overall, our work establishes a solid theoretical fundation for convolutional learning on SCs, highlighting the importance of the Hodge theorem.

Appendix

3.A Simplicial 2-Complex CNNs and Details on Properties

3.A.1 Simplicial locality in details

The construction of SCFs has an intra-simplicial locality. $\mathbf{H}_k\mathbf{x}_k$, which consists of basic operations $\mathbf{L}_{k,\mathrm{d}}\mathbf{x}_k$ and $\mathbf{L}_{k,\mathrm{u}}\mathbf{x}_k$. They are given, on simplex s_i^k, by

$$[\mathbf{L}_{k,\mathrm{d}}\mathbf{x}_k]_i = \sum_{j\in\mathcal{N}_{i,\mathrm{d}}^k\cup\{i\}}[\mathbf{L}_{k,\mathrm{d}}]_{ij}[\mathbf{x}_k]_j, \quad [\mathbf{L}_{k,\mathrm{u}}\mathbf{x}_k]_i = \sum_{j\in\mathcal{N}_{i,\mathrm{u}}^k\cup\{i\}}[\mathbf{L}_{k,\mathrm{u}}]_{ij}[\mathbf{x}_k]_j, \tag{3.18}$$

where s_i^k aggregates signals from its lower and upper neighbors, $\mathcal{N}_{i,\mathrm{d}}^k$ and $\mathcal{N}_{i,\mathrm{u}}^k$. We can compute the t-step shifting recursively as $\mathbf{L}_{k,\mathrm{d}}^t\mathbf{x}_k = \mathbf{L}_{k,\mathrm{d}}(\mathbf{L}_{k,\mathrm{d}}^{t-1}\mathbf{x}_k)$, a one-step shifting of the $(t-1)$-shift result; likewise for $\mathbf{L}_{k,\mathrm{u}}^t\mathbf{x}_k$. A SCF linearly combines such multi-step simplicial shiftings based on lower and upper adjacencies. Thus, the output $\mathbf{H}_k\mathbf{x}_k$ is localized in T_d-hop lower and T_u-hop upper k-simplicial neighborhoods [cf. (2.6)]. SCCNNs preserve such *intra-simplicial locality* as the elementwise nonlinearity does not alter the information locality, shown in Figs. 3.1b and 3.1c.

A SCCNN takes the data on k- and $(k\pm1)$-simplices at layer $l-1$ to compute $\mathbf{x}_k^l$, causing interactions between k-simplices and their (co)faces when all SCFs are identity. In turn, $\mathbf{x}_{k-1}^{l-1}$ contains information on $(k-2)$-simplices from layer $l-2$. Likewise for $\mathbf{x}_{k+1}^{l-1}$, thus, $\mathbf{x}_k^l$ also contains information up to $(k\pm2)$-simplices if $L\geq 2$, because $\mathbf{B}_k\sigma(\mathbf{B}_{k+1})\neq\mathbf{0}$. Accordingly, this *inter-simplicial locality* extends to the whole SC if $L\geq K$, unlike linear filters in a SC where the locality happens up to the adjacent simplices [Isufi & Yang, 2022; Schaub et al., 2021], which limits its expressive power. This locality is further coupled with the intra-locality through three SCFs such that a node not only interacts with the edges incident to it and direct triangles including it, but also edges and triangles further hops away which contribute to the neighboring nodes, as shown in Fig. 3.1d.

3.A.2 Complexity

In a SCCNN layer for computing $\mathbf{x}_k^l$, there are $2+T_\mathrm{d}+T_\mathrm{u}$ filter coefficients for the SCF $\mathbf{H}_k^l$, and $1+T_\mathrm{d}$ and $1+T_\mathrm{u}$ for $\mathbf{H}_{k,\mathrm{d}}^l$ and $\mathbf{H}_{k,\mathrm{u}}^l$, respectively, which gives the parameter complexity of order $\mathcal{O}(T_\mathrm{d}+T_\mathrm{u})$. This complexity will increase by F_lF_{l-1} fold for the multi-feature case, and likewise for the computational complexity. Given the inputs $\{\mathbf{x}_{k-1}^{l-1},\mathbf{x}_k^{l-1},\mathbf{x}_{k+1}^{l-1}\}$, we discuss the computation complexity of $\mathbf{x}_k^l$ in (3.1).

First, consider the SCF operation $\mathbf{H}_k^l\mathbf{x}_k^{l-1}$. As discussed in the localities, it is a composition

of T_{d}-step lower and T_{u}-step upper simplicial shiftings. Each simplicial shifting has a computational complexity of order $\mathcal{O}(n_k m_k)$ dependent on the number of neighbors m_k where n_k is the number of k-simplices. Thus, this operation has a complexity of order $\mathcal{O}(n_k m_k (T_{\mathrm{d}} + T_{\mathrm{u}}))$.

Second, consider the lower SCF operation $\mathbf{H}^l_{k,\mathrm{d}} \mathbf{B}_k^\top \mathbf{x}^{l-1}_{k-1}$. As incidence matrix $\mathbf{B}_k$ is sparse, it has $n_k(k+1)$ nonzero entries as each k-simplex has $k+1$ faces. This leads to a complexity of order $O(n_k k)$ for operation $\mathbf{B}_k^\top \mathbf{x}^{l-1}_{k-1}$. Followed by a lower SCF operation, i.e., a T_{d}-step lower simplicial shifting, thus, a complexity of order $\mathcal{O}(kn_k + n_k m_k T_{\mathrm{d}})$ is needed.

Third, consider the upper SCF operation $\mathbf{H}^l_{k,\mathrm{u}} \mathbf{B}_{k+1} \mathbf{x}^{l-1}_{k+1}$. Likewise, incidence matrix $\mathbf{B}_{k+1}$ has $n_{k+1}(k+2)$ nonzero entries. This leads to a complexity of order $\mathcal{O}(n_{k+1}k)$ for the projection operation $\mathbf{B}_{k+1} \mathbf{x}^{l-1}_{k+1}$. Followed by an upper SCF operation, i.e., a T_{u}-step upper simplicial shifting, thus, a complexity of order $\mathcal{O}(kn_{k+1} + n_k m_k T_{\mathrm{u}})$ is needed.

Finally, we have a computational complexity of order $\mathcal{O}(k(n_k + n_{k+1}) + N_k M_k (T_{\mathrm{d}} + T_{\mathrm{u}}))$ in total.

Remark 3.24. The lower SCF operation $\mathbf{H}^l_{k,\mathrm{d}} \mathbf{B}_k^\top \mathbf{x}^{l-1}_{k-1}$ can be further reduced if $n_{k-1} \ll n_k$. Note that we have

$$\mathbf{H}^l_{k,\mathrm{d}} \mathbf{B}_k^\top \mathbf{x}^{l-1}_{k-1} = \sum_{t=0}^{T_{\mathrm{d}}} w'^l_{k,\mathrm{d},t} \mathbf{L}^t_{k,\mathrm{d}} \mathbf{B}_k^\top \mathbf{x}^{l-1}_{k-1} = \mathbf{B}_k^\top \sum_{t=0}^{T_{\mathrm{d}}} w'^l_{k,\mathrm{d},t} \mathbf{L}^t_{k-1,\mathrm{u}} \mathbf{x}^{l-1}_{k-1}, \tag{3.19}$$

where the second equality comes from that $\mathbf{L}_{k,\mathrm{d}} \mathbf{B}_k^\top = \mathbf{B}_k^\top \mathbf{B}_k \mathbf{B}_k^\top = \mathbf{B}_k^\top \mathbf{L}_{k-1,\mathrm{u}}$, and $\mathbf{L}^2_{k,\mathrm{d}} \mathbf{B}_k^\top = (\mathbf{B}_k^\top \mathbf{B}_k)(\mathbf{B}_k^\top \mathbf{B}_k)\mathbf{B}_k^\top = \mathbf{B}_k^\top (\mathbf{B}_k \mathbf{B}_k^\top)(\mathbf{B}_k \mathbf{B}_k^\top) = \mathbf{B}_k^\top \mathbf{L}_{k-1,\mathrm{u}}$ and likewise for general t. Using the RHS of (3.19) where the simplicial shifting is performed in the $(k-1)$-simplicial space, we have a complexity of order $\mathcal{O}(kn_k + n_{k-1} m_{k-1} T_{\mathrm{d}})$. Similarly, we have

$$\mathbf{H}^l_{k,\mathrm{u}} \mathbf{B}_{k+1} \mathbf{x}^{l-1}_{k+1} = \sum_{t=0}^{T_{\mathrm{u}}} w'^l_{k,\mathrm{u},t} \mathbf{L}^t_{k,\mathrm{u}} \mathbf{B}_{k+1} \mathbf{x}^{l-1}_{k+1} = \mathbf{B}_{k+1} \sum_{t=0}^{T_{\mathrm{u}}} w'^l_{k,\mathrm{u},t} \mathbf{L}^t_{k+1,\mathrm{d}} \mathbf{x}^{l-1}_{k+1} \tag{3.20}$$

where the simplicial shifting is performed in the $(k+1)$-simplicial space. If it follows that $n_{k+1} \ll n_k$, we have a smaller complexity of $\mathcal{O}(kn_{k+1} + n_{k+1} m_{k+1} T_{\mathrm{u}})$ by using the RHS of (3.20).

3.A.3 Diffusion process on SCs

Diffusion process on graphs can be generalized to SCs to characterize the evolution of simplicial data over the SC, in analogy to data diffusion on nodes Anand et al. [2022]; Grady & Polimeni [2010]; Ziegler et al. [2022]. Here we provide an informal treatment of how discretizing diffusion equations on SCs can give resemblances of simplicial shifting layers. Consider diffusion equation and its Euler discretization with a unit time step

$$\dot{\mathbf{x}_k}(t) = -\mathbf{L}_k \mathbf{x}_k(t), \text{ Euler step: } \mathbf{x}_k(t+1) = \mathbf{x}_k(t) - \mathbf{L}_k \mathbf{x}_k(t) = (\mathbf{I} - \mathbf{L}_k)\mathbf{x}_k(t) \tag{3.21}$$

with an initial condition $\mathbf{x}_k(t) = \mathbf{x}^0_k$. The solution of this diffusion is $\mathbf{x}_k(t) = \exp(-\mathbf{L}_k t)\mathbf{x}^0_k$. As the time increases, the simplicial data reaches to a steady state $\dot{\mathbf{x}_k}(t) = \mathbf{0}$, which lies in

the harmonic space $\ker(\mathbf{L}_k)$. The simplicial shifting layer resembles this Euler step with a weight and nonlinearity when viewing the time step as layer index. Thus, a NN composed of simplicial shifting layers can suffer from oversmoothing on SCs, giving outputs with decreasing Dirichlet energies as the number of layers increases.

Now let us consider the case where the two Laplacians have different coefficients

$$\dot{\mathbf{x}_k}(t) = -\mathbf{L}_{k,\mathrm{d}}\mathbf{x}_k(t) - \gamma\mathbf{L}_{k,\mathrm{u}}\mathbf{x}_k(t), \text{ Euler step: } \mathbf{x}_k(t) = (\mathbf{I} - \mathbf{L}_{k,\mathrm{d}} - \gamma\mathbf{L}_{k,\mathrm{u}})\mathbf{x}_k(t). \quad (3.22)$$

The steady state of this diffusion equation follows $(\mathbf{L}_{k,\mathrm{d}} + \gamma\mathbf{L}_{k,\mathrm{u}})\mathbf{x}_k(t) = \mathbf{0}$, where $\mathbf{x}_k(t)$ would be in the kernal space of $\mathbf{L}_k$ still. However, before reaching this state, when the time increases, $\mathbf{x}_k(t)$ would primarily approach to the kernel of $\mathbf{B}_{k+1}^\top$ if $\gamma \gg 1$, in which the lower part of the Dirichlet energy remains, i.e., the decrease of $D(\mathbf{x}(t))$ slows down.

When accounting for inter-simplicial couplings, consider there are nontrivial $\mathbf{x}_{k-1}$ and $\mathbf{x}_{k+1}$ and the diffusion equation becomes

$$\dot{\mathbf{x}_k}(t) = -\mathbf{L}_k\mathbf{x}_k(t) + \mathbf{B}_k^\top\mathbf{x}_{k-1} + \mathbf{B}_{k+1}\mathbf{x}_{k+1}, \quad (3.23)$$

which has source terms $\mathbf{B}_k^\top\mathbf{x}_{k-1} + \mathbf{B}_{k+1}\mathbf{x}_{k+1}$. Consider a steady state $\dot{\mathbf{x}_k} = 0$. We have $\mathbf{L}_k\mathbf{x}_k(t) = \mathbf{x}_{k,\mathrm{d}} + \mathbf{x}_{k,\mathrm{u}}$, where $\mathbf{x}_k$ is not in the kernel space of $\mathbf{L}_k$. The Euler discretization gives

$$\mathbf{x}_k(t+1) = (\mathbf{I} - \mathbf{L}_k)\mathbf{x}_k(t) + \mathbf{x}_{k,\mathrm{d}} + \mathbf{x}_{k,\mathrm{u}}. \quad (3.24)$$

The layer in Bunch et al. [2020] $\mathbf{x}_k^{l+1} = w_0(\mathbf{I} - \mathbf{L}_k)\mathbf{x}_k^l + w_1\mathbf{x}_{k,\mathrm{d}} + w_2\mathbf{x}_{k,\mathrm{u}}$ is a weighted variant of above step when viewing time steps as layers.

3.B Related works

We describe how the SCCNN in (3.4) generalize other NNs on graphs and SCs in Table 3.7. For simplicity, we use $\mathbf{Y}$ and $\mathbf{X}$ to denote the output and input, respectively, without the index l. Note that for GNNs, $\mathbf{L}_{0,\mathrm{d}}$ is not defined.

Table 3.7: SCCNNs generalize other convolutional architectures on SCs.

Methods	Parameters (n.d. denotes "not defined")
Ebli et al. [2020]	$w_{k,\mathrm{d},t}^l = w_{k,\mathrm{u},t}^l, \mathbf{H}_{k,\mathrm{d}}^l, \mathbf{H}_{k,\mathrm{u}}^l$ n.d.
Roddenberry et al. [2021]	$T_\mathrm{d} = T_\mathrm{u} = 1, \mathbf{H}_{k,\mathrm{d}}^l, \mathbf{H}_{k,\mathrm{u}}^l$ n.d.
Yang et al. [2022a]	$\mathbf{H}_{k,\mathrm{d}}^l, \mathbf{H}_{k,\mathrm{u}}^l$ n.d.
Bunch et al. [2020]	$T_\mathrm{d} = T_\mathrm{u} = 1, \mathbf{H}_{k,\mathrm{d}}^l = \mathbf{H}_{k,\mathrm{u}}^l = \mathbf{I}$
Bodnar et al. [2021b]	$T_\mathrm{d} = T_\mathrm{u} = 1, \mathbf{H}_{k,\mathrm{d}}^l = \mathbf{H}_{k,\mathrm{u}}^l = \mathbf{I}$

Gama et al. [2020a] proposed to build a GNN layer with the form

$$\mathbf{Y}_0 = \sigma\left(\sum_{t=0}^{T_\mathrm{u}} \mathbf{L}_0^t\mathbf{X}_0\mathbf{W}_{0,\mathrm{u},t}\right) \quad (3.25)$$

where the convolution step is performed via a graph filter [Gama et al., 2019a; 2020b; Sandryhaila & Moura, 2013; 2014]. This GNN can be easily built as a special SCCNN without contributions from edges. Furthermore, Defferrard et al. [2016] considered a fast implementation of this GNN via a Chebyshev polynomial, while Wu et al. [2019] simplified this by setting $\mathbf{W}_{0,t,\mathrm{u}}$ as zeros for $t < T_{\mathrm{u}}$. Kipf & Welling [2017] further simplified this by setting $T_{\mathrm{u}} = 1$, namely, GCN.

Yang et al. [2022a] proposed a simplicial convolutional neural network (SCNN) to learn from k-simplicial signals

$$\mathbf{Y}_k = \sigma\left(\sum_{t=0}^{T_{\mathrm{d}}} \mathbf{L}_{k,\mathrm{d}}^t \mathbf{X}_k \mathbf{W}_{k,\mathrm{d},t} + \sum_{t=0}^{T_{\mathrm{u}}} \mathbf{L}_{k,\mathrm{u}}^t \mathbf{X}_k \mathbf{W}_{k,\mathrm{u},t}\right) \tag{3.26}$$

where the linear operation is also defined as a simplicial convolution filter in Yang et al. [2022a]. This is a special SCCNN with a focus on one simplex level without taking into the lower and upper contributions consideration. The simplicial neural network (SNN) of Ebli et al. [2020] did not differentiate the lower and the upper convolutions with a form of $\mathbf{Y}_k = \sigma(\sum_{t=0}^{T} \mathbf{L}_k^t \mathbf{X}_k \mathbf{W}_{k,t})$, which leads to a joint processing in the gradient and curl subspaces as analyzed in Section 3.3.

While Roddenberry et al. [2021] proposed an architecture (referred to as PSNN)of a particular form of (3.26) with $T_{\mathrm{d}} = T_{\mathrm{u}} = 1$, performing only a one-step simplicial shifting (3.18). Keros et al. [2022] also performs a one-step simplicial shifting but with an inverted Hodge Laplacian to localize the homology group in an SC. An attention mechanism was added to both SCNNs and PSNNs by Giusti et al. [2022] and Goh et al. [2022], respectively. Battiloro et al. [2023] added the attention mechanism to SCCNNs.

To account for the information from adjacent simplices, Bunch et al. [2020] proposed a simplicial 2-complex CNN (S2CCNN)

$$\begin{aligned}
\mathbf{Y}_0 &= \sigma\big(\mathbf{L}_0 \mathbf{X}_0 \mathbf{W}_{0,\mathrm{u},1} + \mathbf{B}_1 \mathbf{X}_1 \mathbf{W}'_{0,\mathrm{u},0}\big) \\
\mathbf{Y}_1 &= \sigma\big(\mathbf{B}_1^\top \mathbf{X}_0 \mathbf{W}_{1,\mathrm{d},0} + \mathbf{L}_1 \mathbf{X}_1 \mathbf{W}_{1,1} + \mathbf{B}_2 \mathbf{X}_2 \mathbf{W}'_{1,\mathrm{u},0}\big) \\
\mathbf{Y}_2 &= \sigma\big(\mathbf{B}_2^\top \mathbf{X}_1 \mathbf{W}_{2,\mathrm{d},0} + \mathbf{L}_{2,\mathrm{u}} \mathbf{X}_2 \mathbf{W}_{2,\mathrm{u},1}\big)
\end{aligned} \tag{3.27}$$

which is limited to SCs of order two. Note that instead of Hodge Laplacians, simplicial adjacency matrices with self-loops are used in Bunch et al. [2020], which encode equivalent information as setting all filter orders in SCCNNs as one. It is a particular form of the SCCNN where the SCF is a one-step simplicial shifting operation without differentiating the lower and upper shifting, and the lower and upper contributions are simply added, not convolved or shifted by lower and upper SCFs. That is, Bunch et al. [2020] can be obtained from (3.1) by setting lower and upper SCFs as identity, $\mathbf{H}_{k,\mathrm{d}} = \mathbf{H}_{k,\mathrm{u}} = \mathbf{I}$, and setting $w_{k,\mathrm{d},t} = w_{k,\mathrm{u},t}$ and $T_{\mathrm{d}} = T_{\mathrm{u}} = 1$ for the SCF $\mathbf{H}_k$. The convolution in Yang et al. [2022c, Eq. 3] is the same as Bunch et al. [2020] though it was performed in a block matrix fashion.

The combination of graph shifting and edge shifting in Chen et al. [2022d] can be again seen as a special S2CCNN, where the implementation was performed in a block matrix fashion.

Bodnar et al. [2021b] proposed a message passing scheme which collects information from one-hop simplicial neighbors and direct faces and cofaces as Bunch et al. [2020] and Yang et al. [2022c], but replacing the one-step shifting and projections from (co)faces by some learnable functions. The same message passing was applied for simplicial representation learning by Hajij et al. [2021].

Lastly, there are works on signal processing and NNs on cell complexes. For example, Roddenberry et al. [2022]; Sardellitti et al. [2021] generalized the signal processing techniques from SCs to cell complexes, Bodnar et al. [2021a]; Hajij et al. [2020] performed message passing on cell complexes as in SCs and Hajij et al. [2022] added the attention mechanism. Cell complexes are a more general model compared to SCs, where k-cells compared to k-simplices contain any shapes homeomorphic to a k-dimensional closed balls in Euclidean space, e.g., a filled polygon is a 2-cell while only triangles are 2-simplices. We refer to Hansen & Ghrist [2019] for a more formal definition of cell complexes. Despite cell complexes are more powerful to model real-world higher-order structures, SCCNNs can be easily generalized to cell complexes by considering any k-cells instead of only k-simplices in the algebraic representations, and the theoretical analysis in this chapter can be adapted to cell complexes as well.

3.C Proofs for Section 3.2

3.C.1 Dirichlet energy minimization perspective

Hodge Laplacian smoothing. We can find the gradient of problem (3.6) as $\frac{\partial D}{\partial \mathbf{x}_k} = \mathbf{B}_k^\top \mathbf{B}_k \mathbf{x}_k + \gamma \mathbf{B}_{k+1}\mathbf{B}_{k+1}^\top \mathbf{x}_k$, thus, a gradient descent step follows as (3.6) with a step size η.

Proof of Proposition 3.6. Consider $\eta = 1$.

$$\begin{aligned} D(\mathbf{x}_k^{l+1}) &= w_0^2 \|\mathbf{B}_k(\mathbf{I} - \mathbf{L}_{k,\mathrm{d}} - \gamma \mathbf{L}_{k,\mathrm{u}})\mathbf{x}_k^l\|_2^2 + w_0^2 \|\mathbf{B}_{k+1}^\top (\mathbf{I} - \mathbf{L}_{k,\mathrm{d}} - \gamma \mathbf{L}_{k,\mathrm{u}})\mathbf{x}_k^l\|_2^2 \\ &= w_0^2 \|(\mathbf{I} - \mathbf{L}_{k-1,\mathrm{u}})\mathbf{B}_k \mathbf{x}_k^l\|_2^2 + w_0^2 \|(\mathbf{I} - \gamma \mathbf{L}_{k+1,\mathrm{d}})\mathbf{B}_{k+1}^\top \mathbf{x}_k^l\|_2^2 \\ &\leq w_0^2 \|(\mathbf{I} - \mathbf{L}_{k-1,\mathrm{u}})\|_2^2 \|\mathbf{B}_k \mathbf{x}_k^l\|_2^2 + w_0^2 \|(\mathbf{I} - \gamma \mathbf{L}_{k+1,\mathrm{d}})\|_2^2 \|\mathbf{B}_{k+1}^\top \mathbf{x}_k^l\|_2^2 \end{aligned} \tag{3.28}$$

which follows from triangle inequality. By definition, we have $\|\mathbf{I} - \mathbf{L}_{k-1,\mathrm{u}}\|_2^2 = \|\mathbf{I} - \mathbf{L}_{k,\mathrm{d}}\|_2^2$ and $\|\mathbf{I} - \mathbf{L}_{k,\mathrm{u}}\|_2^2 = \|\mathbf{I} - \mathbf{L}_{k+1,\mathrm{d}}\|_2^2$. Also, we have $\|\mathbf{I} - \mathbf{L}_k\|_2^2 = \max\{\|\mathbf{I} - \mathbf{L}_{k,\mathrm{d}}\|_2^2, \|\mathbf{I} - \mathbf{L}_{k,\mathrm{u}}\|_2^2\}$ Thus, we have $D(\mathbf{x}_k^{l+1}) \leq w_0^2 \|\mathbf{I} - \mathbf{L}_k\|_2^2 D(\mathbf{x}_k^l)$ when $\gamma = 1$. When $w_0^2 \|\mathbf{I} - \mathbf{L}_k\|_2^2 < 1$, Dirichlet energy $D(\mathbf{x}_k^{l+1})$ will exponentially decrease as l increases. □

When $\gamma \neq 1$, from (3.28), we have $D(\mathbf{x}_k^{l+1}) = D_{\mathrm{d}}(\mathbf{x}_k^{l+1}) + D_{\mathrm{u}}(\mathbf{x}_k^{l+1})$, which follows

$$D_{\mathrm{d}}(\mathbf{x}_k^{l+1}) \leq w_0^2 \|(\mathbf{I} - \mathbf{L}_{k,\mathrm{d}})\|_2^2 D_{\mathrm{d}}(\mathbf{x}_k^l) \text{ and } D_{\mathrm{u}}(\mathbf{x}_k^{l+1}) \leq w_0^2 \|(\mathbf{I} - \gamma \mathbf{L}_{k,\mathrm{u}})\|_2^2 D_{\mathrm{u}}(\mathbf{x}_k^l) \tag{3.29}$$

When $\gamma = 1$, the oversmoothing condition is $\|\mathbf{I} - \mathbf{L}_k\|_2^2 = \max\{\|\mathbf{I} - \mathbf{L}_{k,\mathrm{d}}\|_2^2, \|\mathbf{I} - \mathbf{L}_{k,\mathrm{u}}\|_2^2\} < \frac{1}{w_0^2}$. If $\|\mathbf{I} - \mathbf{L}_k\|_2^2 = \|\mathbf{I} - \mathbf{L}_{k,\mathrm{d}}\|_2^2$, under the oversmoothing condition, by not restricting

γ to be 1, $w_0^2\|(\mathbf{I}-\gamma\mathbf{L}_{k,\mathrm{u}})\|_2^2$ can be larger than 1 depending on the choice, which means $D_{\mathrm{u}}(\mathbf{x}_k^l)$ does not necessarily decrease, so does not $D(\mathbf{x}_k^l)$.

Hodge Laplacian smoothing with sources. The gradient of the objective in (3.7) is given by $\mathbf{L}_k\mathbf{x}_k^l-\mathbf{B}_k^\top\mathbf{x}_{k-1}-\mathbf{B}_{k+1}\mathbf{x}_{k+1}$, which gives the gradient descent update in (3.7) with a step size η.

Consider the layer in Bunch et al. [2020] $\mathbf{x}_k^{l+1}=w_0(\mathbf{I}-\mathbf{L}_k)\mathbf{x}_k^l+w_1\mathbf{x}_{k,\mathrm{d}}+w_2\mathbf{x}_{k,\mathrm{u}}$ with some weights. By triangle inequality, we have $D(\mathbf{x}_k^{l+1})\le w_0^2\|\mathbf{I}-\mathbf{L}_k\|_2^2D(\mathbf{x}_k^l)$ $+w_1^2\lambda_{\max}(\mathbf{L}_{k,\mathrm{d}})\|\mathbf{x}_{k,\mathrm{d}}\|_2^2+w_2^2\lambda_{\max}(\mathbf{L}_{k,\mathrm{u}})\|\mathbf{x}_{k,\mathrm{u}}\|_2^2$. If the weight w_0 is small enough following the condition in Proposition 3.6, the contribution from the projections, controled by weights w_1 and w_2, can compromise the decrease by w_0, maintaining the Dirichlet energy.

3

3.D Proofs for Section 3.3

3.D.1 The SCF is Hodge-invariant in Proposition 3.10

Proof. We first give the following lemma.

Lemma 3.25. *Any finite set of eigenfunctions of a linear operator spans an invariant subspace.*

Then, the proof follows from Lemma 3.25 and Proposition 2.5. □

3.D.2 A derivation of the spectral frequency response in (3.10)

SFT of $\mathbf{x}_k$. First, the SFT of $\mathbf{x}_k$ is given by $\tilde{\mathbf{x}}_k=[\tilde{\mathbf{x}}_{k,\mathrm{H}}^\top,\tilde{\mathbf{x}}_{k,\mathrm{G}}^\top,\tilde{\mathbf{x}}_{k,\mathrm{C}}^\top]^\top$ with the *harmonic embedding* $\tilde{\mathbf{x}}_{k,\mathrm{H}}=\mathbf{U}_{k,\mathrm{H}}^\top\mathbf{x}_k=\mathbf{U}_{k,\mathrm{H}}^\top\mathbf{x}_{k,\mathrm{H}}$ in the zero frequencies, the *gradient embedding* $\tilde{\mathbf{x}}_{k,\mathrm{G}}=\mathbf{U}_{k,\mathrm{G}}^\top\mathbf{x}_k=\mathbf{U}_{k,\mathrm{G}}^\top\mathbf{x}_{k,\mathrm{G}}$ in the gradient frequencies, and the *curl embedding* $\tilde{\mathbf{x}}_{k,\mathrm{C}}=\mathbf{U}_{k,\mathrm{C}}^\top\mathbf{x}_k=\mathbf{U}_{k,\mathrm{C}}^\top\mathbf{x}_{k,\mathrm{C}}$ in the curl frequencies.

SFT of $\mathbf{H}_k\mathbf{x}_k$. By diagonalizing an SCF $\mathbf{H}_k$ with $\mathbf{U}_k$, we have

$$\mathbf{H}_k\mathbf{x}_k=\mathbf{U}_k\widetilde{\mathbf{H}}_k\mathbf{U}_k^\top\mathbf{x}_k=\mathbf{U}_k(\tilde{\mathbf{h}}_k\odot\tilde{\mathbf{x}}_k) \tag{3.30}$$

where $\widetilde{\mathbf{H}}_k=\mathrm{diag}(\tilde{\mathbf{h}}_k)$. Here, $\tilde{\mathbf{h}}_k=[\tilde{\mathbf{h}}_{k,\mathrm{H}}^\top,\tilde{\mathbf{h}}_{k,\mathrm{G}}^\top,\tilde{\mathbf{h}}_{k,\mathrm{C}}^\top]^\top$ is the *frequency response*, given by

$$\begin{cases}\textit{harmonic response}: \tilde{\mathbf{h}}_{k,\mathrm{H}}=(w_{k,\mathrm{d},0}+w_{k,\mathrm{u},0})\mathbf{1},\\ \textit{gradient response}: \tilde{\mathbf{h}}_{k,\mathrm{G}}=\sum_{t=0}^{T_{\mathrm{d}}}w_{k,\mathrm{d},t}\boldsymbol{\lambda}_{k,\mathrm{G}}^{\odot t}+w_{k,\mathrm{u},0}\mathbf{1},\\ \textit{curl response}: \tilde{\mathbf{h}}_{k,\mathrm{C}}=\sum_{t=0}^{T_{\mathrm{u}}}w_{k,\mathrm{u},t}\boldsymbol{\lambda}_{k,\mathrm{C}}^{\odot t}+w_{k,\mathrm{d},0}\mathbf{1},\end{cases}$$

with $(\cdot)^{\odot t}$ the elementwise t-th power of a vector. Thus, we can express $\tilde{\mathbf{h}}_k\odot\tilde{\mathbf{x}}_k$ as

$$[(\tilde{\mathbf{h}}_{k,\mathrm{H}}\odot\tilde{\mathbf{x}}_{k,\mathrm{H}})^\top,(\tilde{\mathbf{h}}_{k,\mathrm{G}}\odot\tilde{\mathbf{x}}_{k,\mathrm{G}})^\top,(\tilde{\mathbf{h}}_{k,\mathrm{C}}\odot\tilde{\mathbf{x}}_{k,\mathrm{C}})^\top]^\top. \tag{3.31}$$

SFT of projections. Second, the lower projection $\mathbf{x}_{k,\mathrm{d}}\in\mathrm{im}(\mathbf{B}_k^\top)$ has only a nonzero gradient embedding $\tilde{\mathbf{x}}_{k,\mathrm{d}}=\mathbf{U}_{k,\mathrm{G}}^\top\mathbf{x}_{k,\mathrm{d}}$. Likewise, the upper projection $\mathbf{x}_{k,\mathrm{u}}\in\mathrm{im}(\mathbf{B}_{k+1})$

contains only a nonzero curl embedding $\tilde{\mathbf{x}}_{k,\mathrm{u}} = \mathbf{U}_{k,\mathrm{C}}^{\top}\mathbf{x}_{k,\mathrm{u}}$. The lower SCF $\mathbf{H}_{k,\mathrm{d}}$ has $\tilde{\mathbf{h}}_{k,\mathrm{d}} = \sum_{t=0}^{T_{\mathrm{d}}} w'_{k,\mathrm{d},t}\boldsymbol{\lambda}_{k,\mathrm{G}}^{\odot t}$ as the frequency response that modulates the gradient embedding of $\mathbf{x}_{k,\mathrm{d}}$ and the upper SCF $\mathbf{H}_{k,\mathrm{u}}$ has $\tilde{\mathbf{h}}_{k,\mathrm{u}} = \sum_{t=0}^{T_{\mathrm{u}}} w'_{k,\mathrm{u},t}\boldsymbol{\lambda}_{k,\mathrm{C}}^{\odot t}$ as the frequency response that modulates the curl embedding of $\mathbf{x}_{k,\mathrm{u}}$.

SFT of $\mathbf{y}_k$. For the output $\mathbf{y}_k = \mathbf{H}_{k,\mathrm{d}}\mathbf{x}_{k,\mathrm{d}} + \mathbf{H}_k\mathbf{x}_k + \mathbf{H}_{k,\mathrm{u}}\mathbf{x}_{k,\mathrm{u}}$, we have

$$\begin{cases} \tilde{\mathbf{y}}_{k,\mathrm{H}} = \tilde{\mathbf{h}}_{k,\mathrm{H}} \odot \tilde{\mathbf{x}}_{k,\mathrm{H}}, \\ \tilde{\mathbf{y}}_{k,\mathrm{G}} = \tilde{\mathbf{h}}_{k,\mathrm{d}} \odot \tilde{\mathbf{x}}_{k,\mathrm{d}} + \tilde{\mathbf{h}}_{k,\mathrm{G}} \odot \tilde{\mathbf{x}}_{k,\mathrm{G}}, \\ \tilde{\mathbf{y}}_{k,\mathrm{C}} = \tilde{\mathbf{h}}_{k,\mathrm{C}} \odot \tilde{\mathbf{x}}_{k,\mathrm{C}} + \tilde{\mathbf{h}}_{k,\mathrm{u}} \odot \tilde{\mathbf{x}}_{k,\mathrm{u}}. \end{cases} \tag{3.32}$$

3.D.3 Expressive power in Proposition 3.11

Proof. From the Cayley-Hamilton theorem Horn & Johnson [2012], we know that an analytical function $f(\mathbf{A})$ of a matrix $\mathbf{A}$ can be expressed as a matrix polynomial of degree at most its minimal polynomial degree, which equals to the number of distinct eigenvalues if $\mathbf{A}$ is positive semi-definite.

Consider an analytical function $\mathbf{G}_{k,\mathrm{d}}$ of $\mathbf{L}_{k,\mathrm{d}}$, defined on the spectrum of $\mathbf{L}_{k,\mathrm{d}}$ via analytical function $g_{k,\mathrm{G}}(\lambda)$ where λ is in the set of zero and the gradient frequencies. Then, $\mathbf{G}_{k,\mathrm{d}}$ can be implemented by a matrix polynomial of $\mathbf{L}_{k,\mathrm{d}}$ of order up to $n_{k,\mathrm{G}}$ where $n_{k,\mathrm{G}}$ is the number of nonzero eigenvalues of $\mathbf{L}_{k,\mathrm{d}}$, i.e., the number of distinct gradient frequencies. Likewise, any analytical function $\mathbf{G}_{k,\mathrm{u}}$ of $\mathbf{L}_{k,\mathrm{u}}$ can be implemented by a matrix polynomial of $\mathbf{L}_{k,\mathrm{u}}$ of order up to $n_{k,\mathrm{C}}$, which is the number of nonzero eigenvalues of $\mathbf{L}_{k,\mathrm{u}}$, i.e., the number of distinct curl frequencies.

Thus, as of the matrix polynomial definition of SCFs in a SCCNN, the expressive power of $\mathbf{H}_{k,\mathrm{d}}\mathbf{x}_{k,\mathrm{d}} + \mathbf{H}_k\mathbf{x}_k + \mathbf{H}_{k,\mathrm{u}}\mathbf{x}_{k,\mathrm{u}}$ is at most $\mathbf{G}'_{k,\mathrm{d}}\mathbf{x}_{k,\mathrm{d}} + (\mathbf{G}_{k,\mathrm{d}} + \mathbf{G}_{k,\mathrm{u}})\mathbf{x}_k + \mathbf{G}'_{k,\mathrm{u}}\mathbf{x}_{k,\mathrm{u}}$, when the matrix polynomial orders (convolution orders) follow $T_{k,\mathrm{d}} = T'_{k,\mathrm{d}} = n_{k,\mathrm{G}}$ and $T_{k,\mathrm{u}} = T'_{k,\mathrm{u}} = n_{k,\mathrm{C}}$. □

3.D.4 Hodge-aware of SCCNN in Theorem 3.12

Proof. Consider a linear mapping $T : V \to V$. An invariant subspace W of T has the property that all vectors $\mathbf{v} \in W$ are transformed by T into vectors also contained in W, i.e., $\mathbf{v} \in W \Longrightarrow T(\mathbf{v}) \in W$. For an input $\mathbf{x} \in \mathrm{im}(\mathbf{B}_k^{\top})$, the output $\mathbf{H}_k\mathbf{x}$ is in $\mathrm{im}(\mathbf{B}_k^{\top})$ too, because of

$$\mathbf{H}_k\mathbf{x} = \sum_t \mathbf{L}_{k,\mathrm{d}}^t\mathbf{x} + \sum_t \mathbf{L}_{k,\mathrm{u}}^t\mathbf{x} = \sum_t \mathbf{L}_{k,\mathrm{d}}^t\mathbf{x} \in \mathrm{im}(\mathbf{B}_k^{\top}) \tag{3.33}$$

where the second equality comes from the orthogonality between $\mathrm{im}(\mathbf{B}_k^{\top})$ and $\mathrm{im}(\mathbf{B}_{k+1})$. Similarly, we can show that for $\mathbf{x} \in \mathrm{im}(\mathbf{B}_{k+1})$, the output $\mathbf{H}_k\mathbf{x} \in \mathrm{im}(\mathbf{B}_{k+1})$; for $\mathbf{x} \in \ker(\mathbf{L}_k)$, the output $\mathbf{H}_k\mathbf{x} \in \ker(\mathbf{L}_k)$. This essentially says the three subspaces of the Hodge decomposition are invariant with respect to the SCF $\mathbf{H}_k$. Likewise, the gradient space is invariant with respect to the lower SCF $\mathbf{H}_{k,\mathrm{d}}$, which says any lower projection

remains in the gradient space after passed by $\mathbf{H}_{k,\mathrm{d}}$; and the curl space is invariant with respect to the upper SCF $\mathbf{H}_{k,\mathrm{u}}$.

Lastly, through the spectral relation in (3.10), the learning operator $\mathbf{H}_k$ in the gradient space is controlled by the learnable weights $\{w_{k,\mathrm{d},t}\}$, which is independent of the learnable weights $\{w_{k,\mathrm{u},t}\}$, associated to the learning of $\mathbf{H}_k$ in the curl space. Likewise, the lower SCF learns in the gradient space as well but with another set of learnable weights $\{w'_{k,\mathrm{d},t}\}$, and the upper SCF learns in the curl space with learnable weights $\{w'_{k,\mathrm{u},t}\}$. From the spectral expressive power, we see that above four independent learning in the two subspaces can be as expressive as any analytical functions of the corresponding frequencies (spectrum). This concludes the independent and expressive learning in the gradient and curl spaces. □

3.E Proofs for Section 3.4

We first give the formulation of SCCNNs on weighted SCs, then we proceed the stability proof.

3.E.1 Proof of Stability of SCCNNs in Theorem 3.23

For a SCCNN in (3.14) in a weighted SC $\mathcal{S}$, we consider its perturbed version in a perturbed SC $\widehat{\mathcal{S}}$ at layer l, given by

$$\hat{\mathbf{x}}_k^l = \sigma(\widehat{\mathbf{H}}_{k,\mathrm{d}}^l \widehat{\mathbf{R}}_{k,\mathrm{d}} \hat{\mathbf{x}}_{k-1}^{l-1} + \widehat{\mathbf{H}}_k^l \hat{\mathbf{x}}_k^{l-1} + \widehat{\mathbf{H}}_{k,\mathrm{u}}^l \widehat{\mathbf{R}}_{k,\mathrm{u}} \hat{\mathbf{x}}_{k+1}^{l-1}) \tag{3.34}$$

which is defined based on perturbed Laplacians with the same set of filter coefficients, and the perturbed projection operators following relativ perturbation model.

Given the initial input $\mathbf{x}_k^0$ for $k = 0, 1, \ldots, K$, our goal is to upper bound the Euclidean distance between the outputs $\mathbf{x}_k^l$ and $\hat{\mathbf{x}}_k^l$ for $l = 1, \ldots, L$,

$$\begin{aligned} \|\hat{\mathbf{x}}_k^l - \mathbf{x}_k^l\|_2 = \|\sigma(&\widehat{\mathbf{H}}_{k,\mathrm{d}}^l \widehat{\mathbf{R}}_{k,\mathrm{d}} \hat{\mathbf{x}}_{k-1}^{l-1} - \mathbf{H}_{k,\mathrm{d}}^l \mathbf{R}_{k,\mathrm{d}} \mathbf{x}_{k-1}^{l-1} \\ &+ \widehat{\mathbf{H}}_k^l \hat{\mathbf{x}}_k^{l-1} - \mathbf{H}_k^l \mathbf{x}_k^{l-1} + \widehat{\mathbf{H}}_{k,\mathrm{u}}^l \widehat{\mathbf{R}}_{k,\mathrm{u}} \hat{\mathbf{x}}_{k+1}^{l-1} - \mathbf{H}_{k,\mathrm{u}}^l \mathbf{R}_{k,\mathrm{u}} \mathbf{x}_{k+1}^{l-1})\|_2. \end{aligned} \tag{3.35}$$

We proceed the proof in two steps: first, we analyze the operator norm $\|\widehat{\mathbf{H}}_k^l - \mathbf{H}_k^l\|_2$ of a SCF $\mathbf{H}_k^l$ and its perturbed version $\widehat{\mathbf{H}}_k^l$; then we look for the bound of the output distance for a general L-layer SCCNN. To ease notations, we omit the subscript such that $\|\mathbf{A}\| = \max_{\|\mathbf{x}\|_2=1} \|\mathbf{A}\mathbf{x}\|_2$ is the operator norm (spectral radius) of a matrix $\mathbf{A}$, and $\|\mathbf{x}\|$ is the Euclidean norm of a vector $\mathbf{x}$.

In the first step we omit the indices k and l for simplicity since they hold for general k and l. We first give a useful lemma.

Lemma 3.26. *Given the ith eigenvector $\mathbf{u}_i$ of $\mathbf{L} = \mathbf{U}\mathbf{\Lambda}\mathbf{U}^\top$, for lower and upper perturbations $\mathbf{E}_\mathrm{d}$ and $\mathbf{E}_\mathrm{u}$, we have*

$$\mathbf{E}_\mathrm{d}\mathbf{u}_i = q_{\mathrm{d}i}\mathbf{u}_i + \mathbf{E}_1\mathbf{u}_i, \quad \mathbf{E}_\mathrm{u}\mathbf{u}_i = q_{\mathrm{u}i}\mathbf{u}_i + \mathbf{E}_2\mathbf{u}_i \tag{3.36}$$

with eigendecompositions $\mathbf{E}_{\mathrm{d}} = \mathbf{V}_{\mathrm{d}}\mathbf{Q}_{\mathrm{d}}\mathbf{V}_{\mathrm{d}}^{\top}$ *and* $\mathbf{E}_{\mathrm{u}} = \mathbf{V}_{\mathrm{u}}\mathbf{Q}_{\mathrm{u}}\mathbf{V}_{\mathrm{u}}^{\top}$ *where* $\mathbf{V}_{\mathrm{d}}$, $\mathbf{V}_{\mathrm{u}}$ *collect the eigenvectors and* $\mathbf{Q}_{\mathrm{d}}$, $\mathbf{Q}_{\mathrm{u}}$ *the eigenvalues. It holds that* $\|\mathbf{E}_1\| \leq \epsilon_{\mathrm{d}}\delta_{\mathrm{d}}$ *and* $\|\mathbf{E}_2\| \leq \epsilon_{\mathrm{u}}\delta_{\mathrm{u}}$, *with* $\delta_{\mathrm{d}} = (\|\mathbf{V}_{\mathrm{d}} - \mathbf{U}\|+1)^2 - 1$ *and* $\delta_{\mathrm{u}} = (\|\mathbf{V}_{\mathrm{u}} - \mathbf{U}\|+1)^2 - 1$ *measuring the eigenvector misalignments.*

Proof. We first prove that $\mathbf{E}_{\mathrm{d}}\mathbf{u}_i = q_{\mathrm{d}i}\mathbf{u}_i + \mathbf{E}_1\mathbf{u}_i$. The perturbation matrix on the lower Laplacian can be written as $\mathbf{E}_{\mathrm{d}} = \mathbf{E}_{\mathrm{d}}^{'} + \mathbf{E}_1$ with $\mathbf{E}_{\mathrm{d}}' = \mathbf{U}\mathbf{Q}_{\mathrm{d}}\mathbf{U}^{\top}$ and $\mathbf{E}_1 = (\mathbf{V}_{\mathrm{d}} - \mathbf{U})\mathbf{Q}_{\mathrm{d}}$ $(\mathbf{V}_{\mathrm{d}} - \mathbf{U})^{\top} + \mathbf{U}\mathbf{Q}_{\mathrm{d}}(\mathbf{V}_{\mathrm{d}} - \mathbf{U})^{\top} + (\mathbf{V}_{\mathrm{d}} - \mathbf{U})\mathbf{Q}_{\mathrm{d}}\mathbf{U}^{\top}$. For the ith eigenvector $\mathbf{u}_i$, we have that

$$\mathbf{E}_{\mathrm{d}}\mathbf{u}_i = \mathbf{E}_{\mathrm{d}}^{'}\mathbf{u}_i + \mathbf{E}_1\mathbf{u}_i = q_{\mathrm{d}i}\mathbf{u}_i + \mathbf{E}_1\mathbf{u}_i \tag{3.37}$$

where the second equality follows from $\mathbf{E}_{\mathrm{d}}'\mathbf{u}_i = q_{\mathrm{d}i}\mathbf{u}_i$. Since $\|\mathbf{E}_{\mathrm{d}}\| \leq \epsilon_{\mathrm{d}}$, it follows that $\|\mathbf{Q}_{\mathrm{d}}\| \leq \epsilon_{\mathrm{d}}$. Then, applying the triangle inequality, we have that

$$\begin{aligned}\|\mathbf{E}_1\| \leq& \|(\mathbf{V}_{\mathrm{d}} - \mathbf{U})\mathbf{Q}_{\mathrm{d}}(\mathbf{V}_{\mathrm{d}} - \mathbf{U})^{\top}\| + \|\mathbf{U}\mathbf{Q}_{\mathrm{d}}(\mathbf{V}_{\mathrm{d}} - \mathbf{U})^{\top}\| + \|(\mathbf{V}_{\mathrm{d}} - \mathbf{U})\mathbf{Q}_{\mathrm{d}}\mathbf{U}\| \\ \leq& \|\mathbf{V}_{\mathrm{d}} - \mathbf{U}\|^2\|\mathbf{Q}_{\mathrm{d}}\| + 2\|\mathbf{V}_{\mathrm{d}} - \mathbf{U}\|\|\mathbf{Q}_{\mathrm{d}}\|\|\mathbf{U}\| \leq \epsilon_{\mathrm{d}}\|\mathbf{V}_{\mathrm{d}} - \mathbf{U}\|^2 + 2\epsilon_{\mathrm{d}}\|\mathbf{V}_{\mathrm{d}} - \mathbf{U}\| \\ =& \epsilon_{\mathrm{d}}((\|\mathbf{V}_{\mathrm{d}} - \mathbf{U}\|+1)^2 - 1) = \epsilon_{\mathrm{d}}\delta_{\mathrm{d}},\end{aligned} \tag{3.38}$$

which completes the proof for the lower perturbation matrix. Likewise, we can prove for $\mathbf{E}_{\mathrm{u}}\mathbf{u}_i$. □

Step I: Stability of the SCF

Proof. **1. Low-order approximation of $\widehat{\mathbf{H}} - \mathbf{H}$.** Given a SCF $\mathbf{H} = \sum_{t=0}^{T_{\mathrm{d}}} w_{\mathrm{d},t}\mathbf{L}_{\mathrm{d}}^t + \sum_{t=0}^{T_{\mathrm{u}}} w_{\mathrm{u},t}\mathbf{L}_{\mathrm{u}}^t$, we denote its perturbed version by $\widehat{\mathbf{H}} = \sum_{t=0}^{T_{\mathrm{d}}} w_{\mathrm{d},t}\widehat{\mathbf{L}}_{\mathrm{d}}^t + \sum_{t=0}^{T_{\mathrm{u}}} w_{\mathrm{u},t}\widehat{\mathbf{L}}_{\mathrm{u}}^t$, where the filter coefficients are the same. The difference between $\mathbf{H}$ and $\widehat{\mathbf{H}}$ can be expressed as

$$\widehat{\mathbf{H}} - \mathbf{H} = \sum_{t=0}^{T_{\mathrm{d}}} w_{\mathrm{d},t}(\widehat{\mathbf{L}}_{\mathrm{d}}^t - \mathbf{L}_{\mathrm{d}}^t) + \sum_{t=0}^{T_{\mathrm{u}}} w_{\mathrm{u},t}(\widehat{\mathbf{L}}_{\mathrm{u}}^t - \mathbf{L}_{\mathrm{u}}^t), \tag{3.39}$$

in which we can compute the first-order Taylor expansion of $\widehat{\mathbf{L}}_{\mathrm{d}}^t$ as

$$\widehat{\mathbf{L}}_{\mathrm{d}}^t = (\mathbf{L}_{\mathrm{d}} + \mathbf{E}_{\mathrm{d}}\mathbf{L}_{\mathrm{d}} + \mathbf{L}_{\mathrm{d}}\mathbf{E}_{\mathrm{d}})^t = \mathbf{L}_{\mathrm{d}}^t + \mathbf{D}_{\mathrm{d},t} + \mathbf{C}_{\mathrm{d}} \tag{3.40}$$

with $\mathbf{D}_{\mathrm{d},t} := \sum_{r=0}^{t-1}(\mathbf{L}_{\mathrm{d}}^r\mathbf{E}_{\mathrm{d}}\mathbf{L}_{\mathrm{d}}^{t-r} + \mathbf{L}_{\mathrm{d}}^{r+1}\mathbf{E}_{\mathrm{d}}\mathbf{L}_{\mathrm{d}}^{t-r-1})$ parameterized by t and $\mathbf{C}_{\mathrm{d}}$ following $\|\mathbf{C}_{\mathrm{d}}\| \leq \sum_{r=2}^{t}\binom{t}{r}\|\mathbf{E}_{\mathrm{d}}\mathbf{L}_{\mathrm{d}} + \mathbf{L}_{\mathrm{d}}\mathbf{E}_{\mathrm{d}}\|^r\|\mathbf{L}_{\mathrm{d}}\|^{t-r}$. Likewise, we can expand $\widehat{\mathbf{L}}_{\mathrm{u}}^t$ as

$$\widehat{\mathbf{L}}_{\mathrm{u}}^t = (\mathbf{L}_{\mathrm{u}} + \mathbf{E}_{\mathrm{u}}\mathbf{L}_{\mathrm{d}} + \mathbf{L}_{\mathrm{d}}\mathbf{u})^t = \mathbf{L}_{\mathrm{u}}^t + \mathbf{D}_{\mathrm{u},t} + \mathbf{C}_{\mathrm{u}} \tag{3.41}$$

with $\mathbf{D}_{\mathrm{u},t} := \sum_{r=0}^{t-1}(\mathbf{L}_{\mathrm{u}}^r\mathbf{E}_{\mathrm{u}}\mathbf{L}_{\mathrm{u}}^{t-r} + \mathbf{L}_{\mathrm{u}}^{r+1}\mathbf{E}_{\mathrm{u}}\mathbf{L}_{\mathrm{u}}^{t-r-1})$ parameterized by t and $\mathbf{C}_{\mathrm{u}}$ following $\|\mathbf{C}_{\mathrm{u}}\| \leq \sum_{r=2}^{t}\binom{t}{r}\|\mathbf{E}_{\mathrm{u}}\mathbf{L}_{\mathrm{u}} + \mathbf{L}_{\mathrm{u}}\mathbf{E}_{\mathrm{u}}\|^r\|\mathbf{L}_{\mathrm{u}}\|^{t-r}$. Then, by substituting (3.40) and (3.41) into (3.39), we have

$$\widehat{\mathbf{H}} - \mathbf{H} = \sum_{t=0}^{T_{\mathrm{d}}} w_{\mathrm{d},t}\mathbf{D}_{\mathrm{d},t} + \sum_{t=0}^{T_{\mathrm{u}}} w_{\mathrm{u},t}\mathbf{D}_{\mathrm{u},t} + \mathbf{F}_{\mathrm{d}} + \mathbf{F}_{\mathrm{u}} \tag{3.42}$$

with negligible terms $\|\mathbf{F}_{\mathrm{d}}\| = \mathcal{O}(\|\mathbf{E}_{\mathrm{d}}\|^2)$ and $\|\mathbf{F}_{\mathrm{u}}\| = \mathcal{O}(\|\mathbf{E}_{\mathrm{u}}\|^2)$ because perturbations are small and the coefficients of higher-order power terms are the derivatives of analytic functions $\tilde{h}_{\mathrm{G}}(\lambda)$ and $\tilde{h}_{\mathrm{C}}(\lambda)$, which are bounded [cf. Definition 3.17].

2. Spectrum of $(\widehat{\mathbf{H}} - \mathbf{H})\mathbf{x}$. Consider a simplicial signal $\mathbf{x}$ with an SFT $\tilde{\mathbf{x}} = \mathbf{U}^\top \mathbf{x} = [\tilde{x}_1, \ldots, \tilde{x}_n]^\top$, thus, $\mathbf{x} = \sum_{i=1}^n \tilde{x}_i \mathbf{u}_i$. Then, we study the effect of the difference of the SCFs on a simplicial signal from the spectral perspective via

$$(\widehat{\mathbf{H}} - \mathbf{H})\mathbf{x} = \sum_{i=1}^{n} \tilde{x}_i \sum_{t=0}^{T_{\mathrm{d}}} w_{\mathrm{d},t} \mathbf{D}_{\mathrm{d},t}^t \mathbf{u}_i + \sum_{i=1}^{n} \tilde{x}_i \sum_{t=0}^{T_{\mathrm{d}}} w_{\mathrm{u},t} \mathbf{D}_{\mathrm{u},t}^t \mathbf{u}_i + \mathbf{F}_{\mathrm{d}}\mathbf{x} + \mathbf{F}_{\mathrm{u}}\mathbf{x} \tag{3.43}$$

where we have

$$\begin{aligned} \mathbf{D}_{\mathrm{d},t}^t \mathbf{u}_i &= \sum_{r=0}^{t-1} (\mathbf{L}_{\mathrm{d}}^r \mathbf{E}_{\mathrm{d}} \mathbf{L}_{\mathrm{d}}^{t-r} + \mathbf{L}_{\mathrm{d}}^{r+1} \mathbf{E}_{\mathrm{d}} \mathbf{L}_{\mathrm{d}}^{t-r-1}) \mathbf{u}_i, \text{ and} \\ \mathbf{D}_{\mathrm{u},t}^t \mathbf{u}_i &= \sum_{r=0}^{t-1} (\mathbf{L}_{\mathrm{u}}^r \mathbf{E}_{\mathrm{u}} \mathbf{L}_{\mathrm{u}}^{t-r} + \mathbf{L}_{\mathrm{u}}^{r+1} \mathbf{E}_{\mathrm{u}} \mathbf{L}_{\mathrm{u}}^{t-r-1}) \mathbf{u}_i. \end{aligned} \tag{3.44}$$

Since the lower and upper Laplacians admit the eigendecompositions for an eigenvector[2] $\mathbf{u}_i$

$$\mathbf{L}_{\mathrm{d}} \mathbf{u}_i = \lambda_{\mathrm{d}i} \mathbf{u}_i, \quad \mathbf{L}_{\mathrm{u}} \mathbf{u}_i = \lambda_{\mathrm{u}i} \mathbf{u}_i, \tag{3.45}$$

we can express the terms in (3.43) as

$$\mathbf{L}_{\mathrm{d}}^r \mathbf{E}_{\mathrm{d}} \mathbf{L}_{\mathrm{d}}^{t-r} \mathbf{u}_i = \mathbf{L}_{\mathrm{d}}^r \mathbf{E}_{\mathrm{d}} \lambda_{\mathrm{d}i}^{t-r} \mathbf{u}_i = \lambda_{\mathrm{d}i}^{t-r} \mathbf{L}_{\mathrm{d}}^r (q_{\mathrm{d}i} \mathbf{u}_i + \mathbf{E}_1 \mathbf{u}_i) = q_{\mathrm{d}i} \lambda_{\mathrm{d}i}^t \mathbf{u}_i + \lambda_{\mathrm{d}i}^{t-r} \mathbf{L}_{\mathrm{d}}^r \mathbf{E}_1 \mathbf{u}_i, \tag{3.46}$$

where the second equality holds from Lemma 3.26. Thus, we have

$$\mathbf{L}_{\mathrm{d}}^{r+1} \mathbf{E}_{\mathrm{d}} \mathbf{L}_{\mathrm{d}}^{t-r-1} \mathbf{u}_i = q_{\mathrm{d}i} \lambda_{\mathrm{d}i}^t \mathbf{u}_i + \lambda_{\mathrm{d}i}^{t-r-1} \mathbf{L}_{\mathrm{d}}^{r+1} \mathbf{E}_1 \mathbf{u}_i. \tag{3.47}$$

With the results in (3.46) and (3.47), we can write the first term in (3.43) as

$$\begin{aligned} \sum_{i=1}^{n} \tilde{x}_i \sum_{t=0}^{T_{\mathrm{d}}} w_{\mathrm{d},t} \mathbf{D}_{\mathrm{d},t}^t \mathbf{u}_i = & \underbrace{\sum_{i=1}^{n} \tilde{x}_i \sum_{t=0}^{T_{\mathrm{d}}} w_{\mathrm{d},t} \sum_{r=0}^{t-1} 2 q_{\mathrm{d}i} \lambda_{\mathrm{d}i}^t \mathbf{u}_i}_{\text{term 1}} \\ & + \underbrace{\sum_{i=1}^{n} \tilde{x}_i \sum_{t=0}^{T_{\mathrm{d}}} w_{\mathrm{d},t} \sum_{r=0}^{t-1} (\lambda_{\mathrm{d}i}^{t-r} \mathbf{L}_{\mathrm{d}}^r \mathbf{E}_1 \mathbf{u}_i + \lambda_{\mathrm{d}i}^{t-r-1} \mathbf{L}_{\mathrm{d}}^{r+1} \mathbf{E}_1 \mathbf{u}_i)}_{\text{term 2}}. \end{aligned} \tag{3.48}$$

Term 1 can be further expanded as

$$\text{term 1} = 2 \sum_{i=1}^{n} \tilde{x}_i q_{\mathrm{d}i} \sum_{t=0}^{T_{\mathrm{d}}} t w_{\mathrm{d},t} \lambda_{\mathrm{d}i}^t \mathbf{u}_i = 2 \sum_{i=1}^{n} \tilde{x}_i q_{\mathrm{d}i} \lambda_{\mathrm{d}i} \tilde{h}'_{\mathrm{G}}(\lambda_{\mathrm{d}i}) \mathbf{u}_i \tag{3.49}$$

[2] Note that they can be jointly diagonalized.

where we used the fact that $\sum_{t=0}^{T_\mathrm{d}} t w_{\mathrm{d},t} \lambda_{\mathrm{d}i}^t = \lambda_{\mathrm{d}i} \tilde{h}'_{\mathrm{G}}(\lambda_{\mathrm{d}i})$. Using $\mathbf{L}_\mathrm{d} = \mathbf{U}\mathbf{\Lambda}_\mathrm{d}\mathbf{U}^\top$ we can write term 2 in (3.48) as

$$\text{term 2} = \sum_{i=1}^{n} \tilde{x}_i \mathbf{U}\mathrm{diag}(\mathbf{g}_{\mathrm{d}i})\mathbf{U}^\top \mathbf{E}_1 \mathbf{u}_i \tag{3.50}$$

where $\mathbf{g}_{\mathrm{d}i} \in \mathbb{R}^n$ has the jth entry

$$\begin{aligned}[\mathbf{g}_{\mathrm{d}i}]_j &= \sum_{t=0}^{T_\mathrm{d}} w_{\mathrm{d},t} \sum_{r=0}^{t-1} \left(\lambda_{\mathrm{d}i}^{t-r} [\mathbf{\Lambda}_\mathrm{d}]_j^r + \lambda_{\mathrm{d}i}^{t-r-1} [\mathbf{\Lambda}_\mathrm{d}]_j^{r+1} \right) \\ &= \begin{cases} 2\lambda_{\mathrm{d}i} \tilde{h}'_{\mathrm{G}}(\lambda_{\mathrm{d}i}) & \text{for } j = i, \\ \frac{\lambda_{\mathrm{d}i} + \lambda_{\mathrm{d}j}}{\lambda_{\mathrm{d}i} - \lambda_{\mathrm{d}j}} (\tilde{h}_{\mathrm{G}}(\lambda_{\mathrm{d}i}) - \tilde{h}_{\mathrm{G}}(\lambda_{\mathrm{d}j})) & \text{for } j \neq i. \end{cases}\end{aligned} \tag{3.51}$$

Now, substituting (3.49) and (3.50) into (3.48), we have

$$\sum_{i=1}^{n} \tilde{x}_i \sum_{t=0}^{T_\mathrm{d}} w_{\mathrm{d},t} \mathbf{D}_{\mathrm{d},t}^t \mathbf{u}_i = 2 \sum_{i=1}^{n} \tilde{x}_i q_{\mathrm{d}i} \lambda_{\mathrm{d}i} \tilde{h}'_{\mathrm{G}}(\lambda_{\mathrm{d}i}) \mathbf{u}_i + \sum_{i=1}^{n} \tilde{x}_i \mathbf{U}\mathrm{diag}(\mathbf{g}_{\mathrm{d}i})\mathbf{U}^\top \mathbf{E}_1 \mathbf{u}_i. \tag{3.52}$$

By following the same steps as in (3.48)-(3.51), we can express also the second term in (3.43) as

$$\sum_{i=1}^{n} \tilde{x}_i \sum_{t=0}^{T_\mathrm{d}} w_{\mathrm{u},t} \mathbf{D}_{\mathrm{u},t}^t \mathbf{u}_i = 2 \sum_{i=1}^{n} \tilde{x}_i q_{\mathrm{u}i} \lambda_{\mathrm{u}i} \tilde{h}'_{\mathrm{C}}(\lambda_{\mathrm{u}i}) \mathbf{u}_i + \sum_{i=1}^{n} \tilde{x}_i \mathbf{U}\mathrm{diag}(\mathbf{g}_{\mathrm{u}i})\mathbf{U}^\top \mathbf{E}_2 \mathbf{u}_i \tag{3.53}$$

where $\mathbf{g}_{\mathrm{u}i} \in \mathbb{R}^n$ is defined as

$$\begin{aligned}[\mathbf{g}_{\mathrm{u}i}]_j &= \sum_{t=0}^{T_\mathrm{d}} w_{\mathrm{u},t} \sum_{r=0}^{t-1} \left(\lambda_{\mathrm{u}i}^{t-r} [\mathbf{\Lambda}_\mathrm{u}]_j^r + \lambda_{\mathrm{u}i}^{t-r-1} [\mathbf{\Lambda}_\mathrm{u}]_j^{r+1} \right) \\ &= \begin{cases} 2\lambda_{\mathrm{u}i} \tilde{h}'_{\mathrm{C}}(\lambda_{\mathrm{u}i}) & \text{for } j = i, \\ \frac{\lambda_{\mathrm{u}i} + \lambda_{\mathrm{u}j}}{\lambda_{\mathrm{u}i} - \lambda_{\mathrm{u}j}} (\tilde{h}_{\mathrm{C}}(\lambda_{\mathrm{u}i}) - \tilde{h}_{\mathrm{C}}(\lambda_{\mathrm{u}j})) & \text{for } j \neq i. \end{cases}\end{aligned} \tag{3.54}$$

3. Bound of $\|(\widehat{\mathbf{H}} - \mathbf{H})\mathbf{x}\|$. Now we are ready to bound $\|(\widehat{\mathbf{H}} - \mathbf{H})\mathbf{x}\|$ based on triangle inequality. First, given the small perturbations $\|\mathbf{E}_\mathrm{d}\| \leq \epsilon_\mathrm{d}$ and $\|\mathbf{E}_\mathrm{u}\| \leq \epsilon_\mathrm{u}$, we have for the last two terms in (3.43)

$$\|\mathbf{F}_\mathrm{d}\mathbf{x}\| \leq \mathcal{O}(\epsilon_\mathrm{d}^2)\|\mathbf{x}\|, \text{ and } \|\mathbf{F}_\mathrm{u}\mathbf{x}\| \leq \mathcal{O}(\epsilon_\mathrm{u}^2)\|\mathbf{x}\|. \tag{3.55}$$

Second, for the first term $\|\sum_{i=1}^n \tilde{x}_i \sum_{t=0}^{T_\mathrm{d}} w_{\mathrm{d},t} \mathbf{D}_\mathrm{d}^t \mathbf{u}_i\|$ in (3.43), we can bound its two terms in (3.49) and (3.50) as

$$\begin{aligned}&\left\| \sum_{i=1}^{n} \tilde{x}_i \sum_{t=0}^{T_\mathrm{d}} w_{\mathrm{d},t} \mathbf{D}_{\mathrm{d},t}^t \mathbf{u}_i \right\| \\ \leq &\left\| 2 \sum_{i=1}^{n} \tilde{x}_i q_{\mathrm{d}i} \lambda_{\mathrm{d}i} \tilde{h}'_{\mathrm{G}}(\lambda_{\mathrm{d}i}) \mathbf{u}_i \right\| + \left\| \sum_{i=1}^{n} \tilde{x}_i \mathbf{U}\mathrm{diag}(\mathbf{g}_{\mathrm{d}i})\mathbf{U}^\top \mathbf{E}_1 \mathbf{u}_i \right\|.\end{aligned} \tag{3.56}$$

For the first term on the RHS of (3.56), we can write

$$\left\| 2\sum_{i=1}^{n} \tilde{x}_i q_{\mathrm{d}i} \lambda_{\mathrm{d}i} \tilde{h}'_{\mathrm{G}}(\lambda_{\mathrm{d}i}) \mathbf{u}_i \right\|^2 \leq 4 \sum_{i=1}^{n} |\tilde{x}_i|^2 |q_{\mathrm{d}i}|^2 |\lambda_{\mathrm{d}i} \tilde{h}'_{\mathrm{G}}(\lambda_{\mathrm{d}i})|^2 \leq 4\epsilon_d^2 c_{\mathrm{d}}^2 \|\mathbf{x}\|^2, \tag{3.57}$$

which results from, first, $|q_{\mathrm{d}i}| \leq \epsilon_{\mathrm{d}} = \|\mathbf{E}_{\mathrm{d}}\|$ since $q_{\mathrm{d}i}$ is an eigenvalue of $\mathbf{E}_{\mathrm{d}}$; second, the integral Lipschitz property of the SCF $|\lambda \tilde{h}'_{\mathrm{G}}(\lambda)| \leq c_{\mathrm{d}}$; and lastly, the fact that $\sum_{i=1}^{n} |\tilde{x}_i|^2 = \|\tilde{\mathbf{x}}\|^2 = \|\mathbf{x}\|^2$ and $\|\mathbf{u}_i\|^2 = 1$. We then have

$$\left\| 2\sum_{i=1}^{n} \tilde{x}_i q_{\mathrm{d}i} \lambda_{\mathrm{d}i} \tilde{h}'_{\mathrm{G}}(\lambda_{\mathrm{d}i}) \mathbf{u}_i \right\| \leq 2\epsilon_{\mathrm{d}} c_{\mathrm{d}} \|\mathbf{x}\|. \tag{3.58}$$

For the second term in RHS of (3.56), we have

$$\left\| \sum_{i=1}^{n} \tilde{x}_i \mathbf{U} \mathrm{diag}(\mathbf{g}_{\mathrm{d}i}) \mathbf{U}^\top \mathbf{E}_1 \mathbf{u}_i \right\| \leq \sum_{i=1}^{n} |\tilde{x}_i| \|\mathbf{U} \mathrm{diag}(\mathbf{g}_{\mathrm{d}i}) \mathbf{U}^\top\| \|\mathbf{E}_1\| \|\mathbf{u}_i\|, \tag{3.59}$$

which stems from the triangle inequality. We further have $\|\mathbf{U}\mathrm{diag}(\mathbf{g}_{\mathrm{d}i})\mathbf{U}^\top\|$ $= \|\mathrm{diag}(\mathbf{g}_{\mathrm{d}i})\| \leq 2C_{\mathrm{d}}$ resulting from $\|\mathbf{U}\| = 1$ and the c_{d}-integral Lipschitz of $\tilde{h}_{\mathrm{G}}(\lambda)$ [cf. Definition 3.17]. Moreover, it follows that $\|\mathbf{E}_1\| \leq \epsilon_{\mathrm{d}} \delta_{\mathrm{d}}$ from Lemma 3.26, which results in

$$\left\| \sum_{i=1}^{n} \tilde{x}_i \mathbf{U} \mathrm{diag}(\mathbf{g}_{\mathrm{d}i}) \mathbf{U}^\top \mathbf{E}_1 \mathbf{u}_i \right\| \leq 2C_{\mathrm{d}} \epsilon_{\mathrm{d}} \delta_{\mathrm{d}} \sqrt{n} \|\mathbf{x}\| \tag{3.60}$$

where we use that $\sum_{i=1}^{n} |\tilde{x}_i| = \|\tilde{\mathbf{x}}\|_1 \leq \sqrt{n}\|\tilde{\mathbf{x}}\| = \sqrt{n}\|\mathbf{x}\|$. By combining (3.57) and (3.60), we have

$$\left\| \sum_{i=1}^{n} \tilde{x}_i \sum_{t=0}^{T_{\mathrm{d}}} w_{\mathrm{d},t} \mathbf{D}_{\mathrm{d},t}^t \mathbf{u}_i \right\| \leq 2\epsilon_{\mathrm{d}} c_{\mathrm{d}} \|\mathbf{x}\| + 2C_{\mathrm{d}} \epsilon_{\mathrm{d}} \delta_{\mathrm{d}} \sqrt{n} \|\mathbf{x}\|. \tag{3.61}$$

Analogously, we can show that

$$\left\| \sum_{i=1}^{n} \tilde{x}_i \sum_{t=0}^{T_{\mathrm{d}}} w_{\mathrm{u},t} \mathbf{D}_{\mathrm{u},t}^t \mathbf{u}_i \right\| \leq 2\epsilon_{\mathrm{u}} c_{\mathrm{u}} \|\mathbf{x}\| + 2C_{\mathrm{u}} \epsilon_{\mathrm{u}} \delta_{\mathrm{u}} \sqrt{n} \|\mathbf{x}\|. \tag{3.62}$$

Now by combining (3.55), (3.61) and (3.62), we can bound $\|(\widehat{\mathbf{H}} - \mathbf{H})\mathbf{x}\|$ as

$$\begin{aligned} \|(\widehat{\mathbf{H}} - \mathbf{H})\mathbf{x}\| \leq\ & 2\epsilon_{\mathrm{d}} c_{\mathrm{d}} \|\mathbf{x}\| + 2C_{\mathrm{d}} \epsilon_{\mathrm{d}} \delta_{\mathrm{d}} \sqrt{n} \|\mathbf{x}\| + \mathcal{O}(\epsilon_{\mathrm{d}}^2) \|\mathbf{x}\| \\ & + 2\epsilon_{\mathrm{u}} c_{\mathrm{u}} \|\mathbf{x}\| + 2C_{\mathrm{u}} \epsilon_{\mathrm{u}} \delta_{\mathrm{u}} \sqrt{n} \|\mathbf{x}\| + \mathcal{O}(\epsilon_{\mathrm{u}}^2) \|\mathbf{x}\|. \end{aligned} \tag{3.63}$$

By defining $\Delta_{\mathrm{d}} = 2(1 + \delta_{\mathrm{d}} \sqrt{n})$ and $\Delta_{\mathrm{u}} = 2(1 + \delta_{\mathrm{u}} \sqrt{n})$, we can obtain that

$$\|\widehat{\mathbf{H}} - \mathbf{H}\| \leq c_{\mathrm{d}} \Delta_{\mathrm{d}} \epsilon_{\mathrm{d}} + c_{\mathrm{u}} \Delta_{\mathrm{u}} \epsilon_{\mathrm{u}} + \mathcal{O}(\epsilon_{\mathrm{d}}^2) + \mathcal{O}(\epsilon_{\mathrm{u}}^2). \tag{3.64}$$

Thus, we have $\|\mathbf{H}_k^l - \widehat{\mathbf{H}}_k^l\| \leq c_{k,\mathrm{d}} \Delta_{k,\mathrm{d}} \epsilon_{k,\mathrm{d}} + c_{k,\mathrm{u}} \Delta_{k,\mathrm{u}} \epsilon_{k,\mathrm{u}}$ with $\Delta_{k,\mathrm{d}} = 2(1 + \delta_{k,\mathrm{d}} \sqrt{n_k})$ and $\Delta_{k,\mathrm{u}} = 2(1 + \delta_{k,\mathrm{u}} \sqrt{n_k})$ where we ignore the second and higher order terms on $\epsilon_{k,\mathrm{d}}$ and $\epsilon_{k,\mathrm{u}}$. Likewise, we have $\|\mathbf{H}_{k,\mathrm{d}}^l - \widehat{\mathbf{H}}_{k,\mathrm{d}}^l\| \leq c_{k,\mathrm{d}} \Delta_{k,\mathrm{d}} \epsilon_{k,\mathrm{d}}$ for the lower SCF and $\|\mathbf{H}_{k,\mathrm{u}}^l - \widehat{\mathbf{H}}_{k,\mathrm{u}}^l\| \leq c_{k,\mathrm{u}} \Delta_{k,\mathrm{u}} \epsilon_{k,\mathrm{u}}$ for the upper SCF. □

Step II: Stability of SCCNNs

Proof. Given the initial input $\mathbf{x}_k^0$, the Euclidean distance between $\mathbf{x}_k^l$ and $\hat{\mathbf{x}}_k^l$ at layer l can be bounded by using triangle inequality and the c_σ-Lipschitz property of $\sigma(\cdot)$ [cf. Assumption 3.21] as

$$\|\hat{\mathbf{x}}_k^l - \mathbf{x}_k^l\|_2 \leq c_\sigma(\phi_{k,\mathrm{d}}^l + \phi_k^l + \phi_{k,\mathrm{u}}^l), \tag{3.65}$$

with

$$\begin{aligned} \phi_{k,\mathrm{d}}^l :=& \|\widehat{\mathbf{H}}_{k,\mathrm{d}}^l \widehat{\mathbf{R}}_{k,\mathrm{d}} \hat{\mathbf{x}}_{k-1}^{l-1} - \mathbf{H}_{k,\mathrm{d}}^l \mathbf{R}_{k,\mathrm{d}} \mathbf{x}_{k-1}^{l-1}\|, \\ \phi_k^l :=& \|\widehat{\mathbf{H}}_k^l \hat{\mathbf{x}}_k^{l-1} - \mathbf{H}_k^l \mathbf{x}_k^{l-1}\|, \\ \phi_{k,\mathrm{u}}^l :=& \|\widehat{\mathbf{H}}_{k,\mathrm{u}}^l \widehat{\mathbf{R}}_{k,\mathrm{u}} \hat{\mathbf{x}}_{k+1}^{l-1} - \mathbf{H}_{k,\mathrm{u}}^l \mathbf{R}_{k,\mathrm{u}} \mathbf{x}_{k+1}^{l-1}\|. \end{aligned} \tag{3.66}$$

We now focus on upper bounding each of the terms.

1. Term ϕ_k^l. By subtracting and adding $\widehat{\mathbf{H}}_k^l \mathbf{x}_k^{l-1}$ within the norm, and using the triangle inequality, we obtain

$$\begin{aligned} \phi_k^l &\leq \|\widehat{\mathbf{H}}_k^l(\hat{\mathbf{x}}_k^{l-1} - \mathbf{x}_k^{l-1})\| + \|(\widehat{\mathbf{H}}_k^l - \mathbf{H}_k^l)\mathbf{x}_k^{l-1}\| \leq \|\hat{\mathbf{x}}_k^{l-1} - \mathbf{x}_k^{l-1}\| + \|\widehat{\mathbf{H}}_k^l - \mathbf{H}_k^l\|\|\mathbf{x}_k^{l-1}\| \\ &\leq \|\hat{\mathbf{x}}_k^{l-1} - \mathbf{x}_k^{l-1}\| + (c_{k,\mathrm{d}}\Delta_{k,\mathrm{d}}\epsilon_{k,\mathrm{d}} + c_{k,\mathrm{u}}\Delta_{k,\mathrm{u}}\epsilon_{k,\mathrm{u}})\|\mathbf{x}_k^{l-1}\| \end{aligned} \tag{3.67}$$

where we used the SCF stability in (3.64) and that all SCFs have a normalized bounded frequency response in Assumption 3.19. Note that $\widehat{\mathbf{H}}_k^l$ is also characterized by $\tilde{h}_{\mathrm{G}}(\lambda)$ with the same set of filter coefficients as $\mathbf{H}_k^l$.

2. Term $\phi_{k,\mathrm{d}}^l$ and $\phi_{k,\mathrm{u}}^l$. By subtracting and adding a term $\widehat{\mathbf{H}}_{k,\mathrm{d}}^l \widehat{\mathbf{R}}_{k,\mathrm{d}} \mathbf{x}_{k-1}^{l-1}$ within the norm, we have

$$\begin{aligned} \phi_{k,\mathrm{d}}^l &\leq \|\widehat{\mathbf{H}}_{k,\mathrm{d}}^l \widehat{\mathbf{R}}_{k,\mathrm{d}}(\hat{\mathbf{x}}_{k-1}^{l-1} - \mathbf{x}_{k-1}^{l-1})\| + \|(\widehat{\mathbf{H}}_{k,\mathrm{d}}^l \widehat{\mathbf{R}}_{k,\mathrm{d}} - \mathbf{H}_{k,\mathrm{d}}^l \mathbf{R}_{k,\mathrm{d}})\mathbf{x}_{k-1}^{l-1}\| \\ &\leq \|\widehat{\mathbf{R}}_{k,\mathrm{d}}\|\|\hat{\mathbf{x}}_{k-1}^{l-1} - \mathbf{x}_{k-1}^{l-1}\| + \|\widehat{\mathbf{H}}_{k,\mathrm{d}}^l \widehat{\mathbf{R}}_{k,\mathrm{d}} - \mathbf{H}_{k,\mathrm{d}}^l \mathbf{R}_{k,\mathrm{d}}\|\|\mathbf{x}_{k-1}^{l-1}\|, \end{aligned} \tag{3.68}$$

where we used again triangle inequality and $\|\widehat{\mathbf{H}}_{k,\mathrm{d}}^l\| \leq 1$ from Assumption 3.19. For the term $\|\widehat{\mathbf{R}}_{k,\mathrm{d}}^l\|$, we have $\|\widehat{\mathbf{R}}_{k,\mathrm{d}}^l\| \leq \|\mathbf{R}_{k,\mathrm{d}}^l\| + \|\mathbf{J}_{k,\mathrm{d}}\|\|\mathbf{R}_{k,\mathrm{d}}^l\| \leq r_{k,\mathrm{d}}(1+\boldsymbol{\epsilon}_{k,\mathrm{d}})$ where we used $\|\mathbf{R}_{k,\mathrm{d}}^l\| \leq r_{k,\mathrm{d}}$ in Assumption 3.20 and $\|\mathbf{J}_{k,\mathrm{d}}^l\| \leq \boldsymbol{\epsilon}_{k,\mathrm{d}}$. For the second term of RHS in (3.68), by adding and subtracting $\widehat{\mathbf{H}}_{k,\mathrm{d}}^l \mathbf{R}_{k,\mathrm{d}}^l$ we have

$$\begin{aligned} \|\widehat{\mathbf{H}}_{k,\mathrm{d}}^l \widehat{\mathbf{R}}_{k,\mathrm{d}} - \mathbf{H}_{k,\mathrm{d}}^l \mathbf{R}_{k,\mathrm{d}}\| &= \|\widehat{\mathbf{H}}_{k,\mathrm{d}}^l \widehat{\mathbf{R}}_{k,\mathrm{d}} - \widehat{\mathbf{H}}_{k,\mathrm{d}}^l \mathbf{R}_{k,\mathrm{d}}^l + \widehat{\mathbf{H}}_{k,\mathrm{d}}^l \mathbf{R}_{k,\mathrm{d}}^l - \mathbf{H}_{k,\mathrm{d}}^l \mathbf{R}_{k,\mathrm{d}}\| \\ &\leq \|\widehat{\mathbf{H}}_{k,\mathrm{d}}^l\|\|\widehat{\mathbf{R}}_{k,\mathrm{d}} - \mathbf{R}_{k,\mathrm{d}}\| + \|\widehat{\mathbf{H}}_{k,\mathrm{d}}^l - \mathbf{H}_{k,\mathrm{d}}^l\|\|\mathbf{R}_{k,\mathrm{d}}\| \\ &\leq r_{k,\mathrm{d}}\boldsymbol{\epsilon}_{k,\mathrm{d}} + C'_{k,\mathrm{d}}\Delta_{k,\mathrm{d}}\epsilon_{k,\mathrm{d}} r_{k,\mathrm{d}} \end{aligned} \tag{3.69}$$

where we use the stability result of the lower SCF $\mathbf{H}_{k,\mathrm{d}}^l$ in (3.64). By substituting (3.69) into (3.68), we have

$$\phi_{k,\mathrm{d}}^l \leq \hat{r}_{k,\mathrm{d}}\|\hat{\mathbf{x}}_{k-1}^{l-1} - \mathbf{x}_{k-1}^{l-1}\| + (r_{k,\mathrm{d}}\boldsymbol{\epsilon}_{k,\mathrm{d}} + C'_{k,\mathrm{d}}\Delta_{k,\mathrm{d}}\epsilon_{k,\mathrm{d}} r_{k,\mathrm{d}})\|\mathbf{x}_{k-1}^{l-1}\|. \tag{3.70}$$

By following the same procedure [cf. (3.68) and (3.69)], we obtain

$$\phi_{k,\mathrm{u}}^l \leq \hat{r}_{k,\mathrm{u}}\|\hat{\mathbf{x}}_{k+1}^{l-1} - \mathbf{x}_{k+1}^{l-1}\| + (r_{k,\mathrm{u}}\boldsymbol{\epsilon}_{k,\mathrm{u}} + C'_{k,\mathrm{u}}\Delta_{k,\mathrm{u}}\epsilon_{k,\mathrm{u}} r_{k,\mathrm{u}})\|\mathbf{x}_{k+1}^{l-1}\|. \tag{3.71}$$

3. Bound of $\|\hat{\mathbf{x}}_k^l - \mathbf{x}_k^l\|$. Using the notations $t_k, t_{k,\mathrm{d}}$ and $t_{k,\mathrm{u}}$ in Theorem 3.23, we then have a set of recursions, for $k = 0, 1, \ldots, K$

$$\begin{aligned}\|\hat{\mathbf{x}}_k^l - \mathbf{x}_k^l\| \leq & c_\sigma(\hat{r}_{k,\mathrm{d}}\|\hat{\mathbf{x}}_{k-1}^{l-1} - \mathbf{x}_{k-1}^{l-1}\| + t_{k,\mathrm{d}}\|\mathbf{x}_{k-1}^{l-1}\| + \|\hat{\mathbf{x}}_k^{l-1} - \mathbf{x}_k^{l-1}\| + t_k\|\mathbf{x}_k^{l-1}\| \\ & + \hat{r}_{k,\mathrm{u}}\|\hat{\mathbf{x}}_{k+1}^{l-1} - \mathbf{x}_{k+1}^{l-1}\| + t_{k,\mathrm{u}}\|\mathbf{x}_{k+1}^{l-1}\|).\end{aligned} \tag{3.72}$$

Define vector $\mathbf{b}^l$ as $[\mathbf{b}^l]_k = \|\hat{\mathbf{x}}_k^l - \mathbf{x}_k^l\|$ with $\mathbf{b}^0 = \mathbf{0}$. Let $\boldsymbol{\beta}^l$ collect the energy of all outputs at layer l, with $[\boldsymbol{\beta}^l]_k := \|\mathbf{x}_k^{l-1}\|$. We can express the Euclidean distances of all k-simplicial signal outputs for $k = 0, 1, \ldots, K$, as

$$\mathbf{b}^l \preceq c_\sigma \widehat{\mathbf{Z}}\mathbf{b}^{l-1} + c_\sigma \mathbf{T}\boldsymbol{\beta}^{l-1} \tag{3.73}$$

where $\preceq$ indicates elementwise smaller than or equal, and we have

$$\mathbf{T} = \begin{bmatrix} t_0 & t_{0,\mathrm{u}} & & & \\ t_{1,\mathrm{d}} & t_1 & t_{1,\mathrm{u}} & & \\ & \ddots & \ddots & \ddots & \\ & & t_{K-1,\mathrm{d}} & t_{K-1} & t_{K-1,\mathrm{u}} \\ & & & t_{K,\mathrm{d}} & t_K \end{bmatrix} \text{ and}$$
$$\widehat{\mathbf{Z}} = \begin{bmatrix} 1 & \hat{r}_{0,\mathrm{u}} & & & \\ \hat{r}_{1,\mathrm{d}} & 1 & \hat{r}_{1,\mathrm{u}} & & \\ & \ddots & \ddots & \ddots & \\ & & \hat{r}_{K-1,\mathrm{d}} & 1 & \hat{r}_{K-1,\mathrm{u}} \\ & & & \hat{r}_{K,\mathrm{d}} & 1 \end{bmatrix}. \tag{3.74}$$

We are now interested in building a recursion for (3.73) for all layers l. We start with term $\mathbf{x}_k^l$. Based on its expression in (3.14), we bound it as

$$\begin{aligned}\|\mathbf{x}_k^l\| &\leq c_\sigma(\|\mathbf{H}_{k,\mathrm{d}}^l\|\|\mathbf{R}_{k,\mathrm{d}}\|\|\mathbf{x}_{k-1}^{l-1}\| + \|\mathbf{H}_k^l\|\|\mathbf{x}_x^{l-1}\| + \|\mathbf{H}_{k,\mathrm{u}}^l\|\|\mathbf{R}_{k,\mathrm{u}}\|\|\mathbf{x}_{k+1}^{l-1}\|) \\ &\leq c_\sigma(r_{k,\mathrm{d}}\|\mathbf{x}_{k-1}^{l-1}\| + \|\mathbf{x}_x^{l-1}\| + r_{k,\mathrm{u}}\|\mathbf{x}_{k+1}^{l-1}\|),\end{aligned} \tag{3.75}$$

which holds for $k = 0, 1, \ldots, K$. Thus, it can be expressed in the vector form as $\boldsymbol{\beta}^l \preceq c_\sigma \mathbf{Z}\boldsymbol{\beta}^{l-1}$, with

$$\mathbf{Z} = \begin{bmatrix} 1 & r_{0,\mathrm{u}} & & & \\ r_{1,\mathrm{d}} & 1 & r_{1,\mathrm{u}} & & \\ & \ddots & \ddots & \ddots & \\ & & r_{K-1,\mathrm{d}} & 1 & r_{K-1,\mathrm{u}} \\ & & & r_{K,\mathrm{d}} & 1 \end{bmatrix}. \tag{3.76}$$

Similarly, we have $\boldsymbol{\beta}^{l-1} \preceq c_\sigma \mathbf{Z}\boldsymbol{\beta}^{l-2}$, leading to $\boldsymbol{\beta}^l \preceq c_\sigma^l \mathbf{Z}^l\boldsymbol{\beta}^0$ with $\boldsymbol{\beta}^0 = \boldsymbol{\beta}$ [cf. Assumption 3.22]. We can then express the bound (3.73) as

$$\mathbf{b}^l \preceq c_\sigma \widehat{\mathbf{Z}}\mathbf{b}^{l-1} + c_\sigma^l \mathbf{T}\mathbf{Z}^{l-1}\boldsymbol{\beta}. \tag{3.77}$$

Thus, we have

$$\mathbf{b}^0 = \mathbf{0},\ \mathbf{b}^1 \preceq c_\sigma \mathbf{T}\boldsymbol{\beta},\ \mathbf{b}^2 \preceq c_\sigma^2(\widehat{\mathbf{Z}}\mathbf{T}\boldsymbol{\beta} + \mathbf{T}\mathbf{Z}\boldsymbol{\beta}),\ \mathbf{b}^3 \preceq c_\sigma^3(\widehat{\mathbf{Z}}^2\mathbf{T}\boldsymbol{\beta} + \widehat{\mathbf{Z}}\mathbf{T}\mathbf{Z}\boldsymbol{\beta} + \mathbf{T}\mathbf{Z}^2\boldsymbol{\beta}), \quad (3.78)$$

and so on, which, inductively, leads to

$$\mathbf{b}^l \preceq c_\sigma^l \sum_{i=1}^{l} \widehat{\mathbf{Z}}^{i-1}\mathbf{T}\mathbf{Z}^{l-i}\boldsymbol{\beta}. \quad (3.79)$$

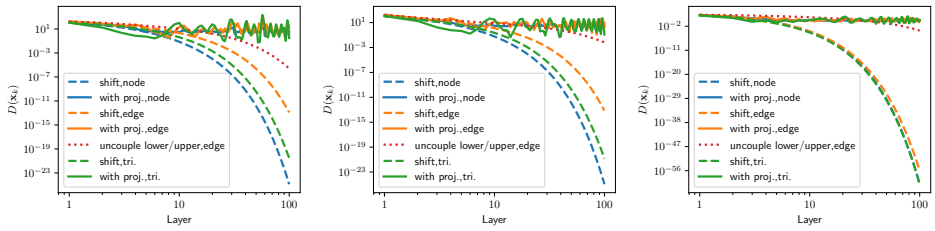
Bt setting $l = L$, we obtain the bound $\mathbf{b}^L \preceq \mathbf{d} = c_\sigma^L \sum_{l=1}^{L} \widehat{\mathbf{Z}}^{l-1}\mathbf{T}\mathbf{Z}^{L-l}\boldsymbol{\beta}$ in Theorem 3.23. □

3

3.F Experiment details

3.F.1 Synthetic experiments on Dirichlet energy evolution

We created a synthetic SC with 100 nodes, 241 edges and 135 triangles with the GUDHI toolbox Rouvreau [2015], and we set the initial inputs on three levels of simplices to be random sampled from $\mathcal{U}([-5, 5])$. We then built a SCCNN composed of simplicial shifting layers with weight w_0 and nonlinearities including $\mathrm{id}, \tanh$ and relu. When the weight follows the condition in Proposition 3.6, from Fig. 3.7 (the dashed lines labled as "shift"), we see that the Dirichlet energies of all three outputs exponentially decrease as the number of layers increases. We then uncoupled the lower and upper parts of the Laplacians in the edge space in the shifting layers by setting $\gamma \neq 1$. As shown in Fig. 3.7 (the dotted lines), the Dirichlet energies of the edge outputs decrease at a slower rate than before. Lastly, we added the inter-simplicial couplings, which overcome the oversmoothing problems, as shown by the solid lines.



(a) identity, $\gamma = 2$ (b) tanh, $\gamma = 2$ (c) relu, $\gamma = 5$

Figure 3.7: Oversmoothing effects of simplicial shifting and the mitigation effects of uncoupling lower and upper adjacencies and accounting for inter-simplicial couplings.

3.F.2 Additional details on Forex experiments

In the forex dataset, there are 25 currencies which can be exchanged pairwise at three timestamps. We first represented their exchange rates on the edges and took the logrithm,

Table 3.8: Forex results (nmse, arbitrage) and the corresponding hyperparameters.

Methods	Random noise	"Curl noise"	Interpolation
Input	0.119 ± 0.004, 25.19 ± 0.874	0.552 ± 0.027, 122.36 ± 5.90	0.717 ± 0.030, 106.40 ± 0.902
ℓ_2-norm	0.036 ± 0.005, 2.29 ± 0.079	0.050 ± 0.002, 11.12 ± 0.537	0.534 ± 0.043 , 9.67 ± 0.082
SNN	0.11 ± 0.005, 23.24 ± 1.03 $L = 5, F = 64, T = 4$, tanh	0.446 ± 0.017, 86.947 ± 2.197 $L = 6, F = 64, T = 3$, tanh	0.702 ± 0.033, 104.738 ± 1.042 $L = 2, F = 64, T = 1$, tanh
PSNN	0.008 ± 0.001, 0.984 ± 0.17 $L = 6, F = 64$, tanh	0.000 ± 0.000, 0.000 ± 0.000 $L = 5, F = 1$, id	0.009 ± 0.001, 1.128 ± 0.329 $L = 6, F = 64$, tanh
Bunch	0.981 ± 0.0 , 22.912 ± 1.228 –	0.981 ± 0.0, 22.912 ± 1.228 –	0.983 ± 0.005 , 19.887 ± 6.341 –
MPSN	0.039 ± 0.004, 7.748 ± 0.943 $L = 2, F = 64$, id, sum	0.076 ± 0.012, 14.922 ± 2.493 $L = 4, F = 64$, tanh, mean	0.117 ± 0.063, 23.147 ± 11.674 $L = 2, F = 64$, tanh, sum
SCCNN, id	0.027 ± 0.005, 0.000 ± 0.000 $L = 2, F = 16, T_{\mathrm{d}} = 0, T_{\mathrm{u}} = 3$	0.000 ± 0.000, 0.000 ± 0.000 $L = 5, F = 1, T_{\mathrm{d}} = 1, T_{\mathrm{u}} = 1$	0.265 ± 0.036 , 0.000 ± 0.000 $L = 2, F = 16, T_{\mathrm{d}} = 0, T_{\mathrm{u}} = 3$
SCCNN, tanh	0.002 ± 0.000, 0.325 ± 0.082 $L = 6, F = 64, T_{\mathrm{d}} = 5, T_{\mathrm{u}} = 2$	0.000 ± 0.000, 0.003 ± 0.003 $L = 1, F = 64, T_{\mathrm{d}} = 2, T_{\mathrm{u}} = 2$	0.003 ± 0.002, 0.279 ± 0.151 $L = 6, F = 64, T_{\mathrm{d}} = 5, T_{\mathrm{u}} = 1$

i.e., $[\mathbf{x}_1]_{[i,j]} = \log_{10} r^{i/j} = -[\mathbf{x}_1]_{[j,i]}$. Then, the total arbitrage can be computed as the total curl $\mathbf{B}_2^\top \mathbf{x}_1$.

We considered to recover the exchange rates under three types of settings: 1) random noise following normal distribution such that the signal-to-noise ration is $-3\mathrm{dB}$, which is spread over the whole simplicial spectrum; 2) "curl noise" projected from triangle noise following normal distribution such that the signal-to-noise ration is $-3\mathrm{dB}$, which is distributed only in the curl space; and 3) 50% of the total forex rates are recorded and the other half is not available, set as zero values.

First, as a baseline method, we chose ℓ_2 norm of the curl $\mathbf{B}_2\mathbf{x}_1$ as a regularizer to reduce the total arbitrage, i.e., $\hat{\mathbf{x}}_1 = (\mathbf{I} + w\mathbf{L}_{1,\mathrm{u}})^{-1}\mathbf{x}_1$ with a regularization weight $w \in [0, 10]$. For the learning methods, we consider the following hyperparameter ranges: the number of layers to be $L \in \{1, 2, \ldots, 6\}$, the number of intermediate features to be $F \in \{1, 16, 32, 64\}$. For the convolutional methods including SNN Ebli et al. [2020], PSNN Roddenberry et al. [2021], Bunch Bunch et al. [2020] and SCCNN, we considered the intermediate layers with nonlinearities including id and $\tanh$. The convolution orders of SNN and SCCNN are set to be $\{1, 2, \ldots, 5\}$. For the message-passing method, MPSN Bodnar et al. [2021b], we considered the setting from [Bodnar et al., 2021b, Eq. 35] where the sum and mean aggregations are used and each message update function is a two-layer MLP. With these noisy or masked rates as inputs and the clean arbitrage-free rates as outputs, we trained different learning methods at the first timestamp, and validated the hyperparameters at the second timestamp, and tested their performance at the thrid one. During the training of 1000 epochs, a normalized MSE loss function and adam optimizer with a fixed learning rate of 0.001 are used. We run the same experiments for 10 times. Table 3.8 reports the best results (nmse) and the total arbitrage, together with the hyperparameters.

3.F.3 Additional details on Simplex prediction

Method in Detail

The method for simplex prediction is generalized from link prediction based on GNNs by Zhang & Chen [2018]: For k-simplex prediction, we use an SCCNN in an SC of order up to k to first learn the features of lower-order simplices up to order $k-1$. Then, we concatenate these embedded lower-order simplicial features and input them to a two-layer MLP which predicts if a k-simplex is positive (closed, shall be included in the SC) or negative (open, not included in the SC).

For example, in 2-simplex prediction, consider an SC of order two, which is built based on nodes, edges and (existing positive) triangles. Given the initial inputs on nodes $\mathbf{x}_0$ and on edges $\mathbf{x}_1$ and zero inputs on triangles $\mathbf{x}_2 = \mathbf{0}$ since we assume no prior knowledge on triangles, for an open triangle $t = [i, j, k]$, an SCCNN is used to learn features on nodes and edges (denoted by $\mathbf{y}$). Then, we input the concatenation of the features on three nodes or three edges to an MLP, i.e., $\mathrm{MLP}_{\mathrm{node}}([\mathbf{y}_0]_i \| [\mathbf{y}_0]_j \| [\mathbf{y}_0]_k)$ or $\mathrm{MLP}_{\mathrm{edge}}([\mathbf{y}]_{[i,j]} \| [\mathbf{y}]_{[j,k]} \| [\mathbf{y}]_{[i,k]})$, to predict if triangle t is positive or negative. A MLP taking both node and edge features is possible, but we keep it on one simplex level for complexity purposes. Similarly, we consider an SCCNN in an SC of order three for 3-simplex prediction, which is followed by an MLP operating on either nodes, edges or triangles.

Data Preprocessing

We consider the data from the Semantic Scholar Open Research Corpus Ammar et al. [2018] to construct a coauthorship complex where nodes are authors and collaborations between k-author are represented by $(k-1)$-simplices. Following the preprocessing in Ebli et al. [2020], we obtain 352 nodes, 1472 edges, 3285 triangles, 5019 tetrahedrons (3-simplices) and a number of other higher-order simplices. The node signal $\mathbf{x}_0$, edge flow $\mathbf{x}_1$ and triangle flow $\mathbf{x}_2$ are the numbers of citations of single author papers and the collaborations of two and three authors, respectively.

For the 2-simplex prediction, we use the collaboration impact (the number of citations) to split the total set of triangles into the positive set $\mathcal{T}_P = \{t | [\mathbf{x}_2]_t > 7\}$ containing 1482 closed triangles and the negative set $\mathcal{T}_N = \{t | [\mathbf{x}_2]_t \leq 7\}$ containing 1803 open triangles such that we have balanced positive and negative samples. We further split the 80% of the positive triangle set for training, 10% for validation and 10% for testing; likewise for the negative triangle set. Note that in the construction of the SC, i.e., the incidence matrix $\mathbf{B}_2$, Hodge Laplacians $\mathbf{L}_{1,\mathrm{u}}$ and $\mathbf{L}_{2,\mathrm{d}}$, we ought to remove negative triangles in the training set and all triangles in the test set. That is, for 2-simplex prediction, we only make use of the training set of the positive triangles since the negative ones are not in the SC.

Similarly, we prepare the dataset for 3-simplex (tetrahedron) prediction, amounting to the tetradic collaboration prediction. We obtain balanced positive and negative tetrahedron sets based on the citation signal $\mathbf{x}_3$. In the construction of $\mathbf{B}_3$, $\mathbf{L}_{2,\mathrm{u}}$ and $\mathbf{L}_{3,\mathrm{d}}$, we again only use the tetrahedrons in the positive training set.

Models

For comparison, we first use heuristic methods proposed in Benson et al. [2018] as baselines to determine if a triangle $t = [i, j, k]$ is closed, namely, 1) Harmonic mean: $s_t = 3/([\mathbf{x}_1]^{-1}_{[i,j]} + [\mathbf{x}_1]^{-1}_{[j,k]} + [\mathbf{x}_1]^{-1}_{[i,k]})$, 2) Geometric mean: $s_t = \lim_{p\to 0}[([\mathbf{x}_1]^p_{[i,j]} + [\mathbf{x}_1]^p_{[j,k]} + [\mathbf{x}_1]^p_{[i,k]})]^{1/p}$, and 3) Arithmetic mean: $s_t = ([\mathbf{x}_1]_{[i,j]} + [\mathbf{x}_1]_{[j,k]} + [\mathbf{x}_1]_{[i,k]})/3$, which compute the triangle weight based on its three faces. Similarly, we generalized these mean methods to compute the weight of a 3-simplex $[i, j, k, m]$ based on the four triangle faces in 3-simplex prediction.

3

We then consider different learning methods. Specifically, 1) "Bunch" by Bunch et al. [2020] (we also generalized this model to 3-dimension for 3-simplex prediction); 2) Message passing simplicial network ("MPSN") by Bodnar et al. [2021b] which provides a baseline of message passing scheme in comparison to the convolution scheme; 3) Principled SNN ("PSNN") by Roddenberry et al. [2021]; 4) SNN by Ebli et al. [2020]; 5) SCNN by Yang et al. [2021]; 6) GNN by Defferrard et al. [2016]; 7) MLP: providing as a baseline for the effect of using inductive models.

For MLP, Bunch, MPSN and our SCCNN, we consider the outputs in the node and edge spaces, respectively, for 2-simplex prediction, which are denoted by a suffix "-Node" or "-Edge". For 3-simplex prediction, the output in the triangle space can be used as well, denoted by a suffix "-Tri.", where we also build SCNNs in both edge and triangle spaces.

Experimental Setup and Hyperparameters

We consider the normalized Hodge Laplacians and incidence matrices, a particular version of the weighted ones Grady & Polimeni [2010]; Horak & Jost [2013]. Specifically, we use the symmetric version of the normalized random walk Hodge Laplacians in the edge space, proposed by Schaub et al. [2020], which were used in Bunch et al. [2020]; Chen et al. [2022c] as well. We generalized the definitions for triangle predictions.

Hyperparameters 1) the number of layers: $L \in \{1, 2, 3, 4, 5\}$; 2) the number of intermediate and output features to be the same as $F \in \{16, 32\}$; 3) the convolution orders for SCCNNs are set to be the same, i.e., $T'_{\mathrm{d}} = T_{\mathrm{d}} = T_{\mathrm{u}} = T'_{\mathrm{u}} = T \in \{1, 2, 3, 4, 5\}$. We do so to avoid the exponential growth of the parameter search space. For GNNs [Defferrard et al., 2016] and SNNs [Ebli et al., 2020], we set the convolution orders to be $T \in \{1, 2, 3, 4, 5\}$ while for SCNNs [Yang et al., 2022a], we allow the lower and upper convolutions to have different orders with $T_{\mathrm{d}}, T_{\mathrm{u}} \in \{1, 2, 3, 4, 5\}$; 4) the nonlinearity in the feature learning phase: LeakyReLU with a negative slope 0.01; 5) MPSN is set as Bodnar et al. [2022]; 6) the MLP in the prediction phase: two layers with a sigmoid nonlinearity. For 2-simplex prediction, the number of the input features for the node features is $3F$, and for the edge features is $3F$. For 3-simplex prediction, the number of the input features for the node features is $4F$, for the edge features is $6F$ and for the triangle features is $4F$ since a 3-simplex has four nodes, six edges and four triangles. The number of the intermediate features is the same as the input features, and that of the output features is one; and, 7) the binary cross entropy loss and the adam optimizer with a learning rate of 0.001 are used; the number of the epochs is 1000 where an early stopping is used. We compute the AUC to compare the performance and run the same experiments for ten times with random data

splitting.

Results

In Table 3.9, we report the best results of each method with the corresponding hyperparameters. Different hyperparameters can lead to similar results, but we report the ones with the *least* complexity. All experiments for simplex predictions were run on a single NVIDIA A40 GPU with 48 GB of memory using CUDA 11.5.

Table 3.9: 2- *(Left)* and 3-Simplex *(Right)* prediction AUC (%) results.

Methods	AUC	Parameters
Harm. Mean	62.8±2.7	–
Arith. Mean	60.8±3.2	–
Geom. Mean	61.7±3.1	–
MLP-Node	68.5±1.6	$L=1, F=32$
GNN	93.9±1.0	$L=5, F=32, T=2$
SNN-Edge	92.0±1.8	$L=5, F=32, T=5$
PSNN-Edge	95.6±1.3	$L=5, F=32$
SCNN-Edge	96.5±1.5	$L=5, F=32, T_{\mathrm{d}}=5, T_{\mathrm{u}}=2$
Bunch-Node	98.3±0.5	$K=1, L=4, F=32$
MPSN-Node	98.1±0.5	$K=1, L=3, F=32$
SCCNN-Node	98.7±0.5	$K=1, L=2, F=32, T=2$

Methods	AUC	Parameters
Harm. Mean	63.6±1.6	–
Arith. Mean	62.2±1.4	–
Geom. Mean	63.1±1.4	–
MLP-Tri.	69.0±2.2	$L=3, F=32$
GNN	96.6±0.5	$L=5, F=32, T=5$
SNN-Tri.	95.1±1.2	$L=5, F=32, T=5$
PSNN-Tri.	98.1±0.5	$L=5, F=32$
SCNN-Tri.	98.3±0.4	$L=5, F=32, T_{\mathrm{d}}=2, T_{\mathrm{u}}=1$
Bunch-Edge	98.5±0.5	$K=3, L=4, F=16$
MPSN-Edge	99.2±0.3	$K=3, L=3, F=32$
SCCNN-Node	99.4±0.3	$K=3, L=3, F=32, T=3$

Complexity

Table 3.10: *(Left)* Complexity of the best three methods for 2-simplex prediction. *(Right)* Running time of SCCNN with different layers and convolution orders.

Method	#params.	Running time (seconds per epoch)
SCCNN	24288	0.073
Bunch	21728	0.140
MPSN	84256	0.028

Hyperparams.	$T=2$	$T=5$
$L=2$	0.073	0.082
$L=3$	0.110	0.130
$L=5$	0.192	0.237

Here we report the number of parameters and the running time of SCCNN for 2-simplex prediction on one NVIDIA Quadro K2200 with 4GB memory, compared with the two best alternatives. MPSN, compared to convolutional methods, has three times more parameters, analogous to the comparison between message-passing and graph convolutional NNs. We also report the running time as the layers and convolution orders increase.

Ablation Study

We perform an ablation study to observe the roles of different components in SCCNNs.

SC Order. We investigate the influence of the SC order K. Table 3.11 reports the 2-simplex prediction results for $K=\{1,2\}$ and the 3-simplex prediction results for $K=\{1,2,3\}$. We observe that for k-simplex prediction, it does not necessarily guarantee a better prediction

with a higher-order SC, which further indicates that a positive simplex could be well encoded by both its faces and other lower-order subsets. For example, in 2-simplex prediction, SC of order one gives better results than SC of order two (similar for Bunch), showing that in this coauthorship complex, triadic collaborations are better encoded by features on nodes than pairwise collaborations. In 3-simplex prediction, SCs of different orders give similar results, showing that tetradic collaborations can be encoded by nodes, as well as by pairwise and triadic collaborations.

Table 3.11: Prediction results of SCCNNs with different SC order K.

Method	2-Simplex	Parameters
SCCNN-Node	98.7±0.5	$K=1, L=2, F=32, T=2$
SCCNN-Node	98.4±0.5	$K=2, L=2, F=32, T=2$
Bunch-Node	98.3±0.4	$K=1, L=4, F=32$
Bunch-Node	98.0±0.4	$K=2, L=4, F=32$
MPSN-Node	94.5±1.5	$K=1, L=3, F=32$
MPSN-Node	98.1±0.5	$K=2, L=3, F=32$
SCCNN-Edge	97.9±0.9	$K=1, L=3, F=32, T=5$
SCCNN-Edge	95.9±1.0	$K=2, L=5, F=32, T=3$
Bunch-Edge	97.3±1.1	$K=1, L=4, F=32$
Bunch-Edge	94.6±1.2	$K=2, L=4, F=32$
MPSN-Edge	94.1±2.4	$K=1, L=3, F=32$
MPSN-Edge	97.0±1.2	$K=2, L=2, F=16$

Method	3-Simplex	Parameters
SCCNN-Node	99.3±0.3	$K=1, L=2, F=32, T=1$
SCCNN-Node	99.3±0.2	$K=2, L=2, F=32, T=5$
SCCNN-Node	99.4±0.3	$K=3, L=3, F=32, T=3$
MPSN-Node	96.0±1.2	$K=1, L=3, F=32$
MPSN-Node	98.2±0.8	$K=2, L=2, F=32$
SCCNN-Edge	98.9±0.5	$K=1, L=3, F=32, T=5$
SCCNN-Edge	99.2±0.4	$K=2, L=5, F=32, T=5$
SCCNN-Edge	99.0±1.0	$K=3, L=5, F=32, T=5$
MPSN-Edge	96.3±1.1	$K=1, L=3, F=32$
MPSN-Edge	98.3±0.8	$K=2, L=3, F=32$
SCCNN-Tri.	97.9±0.7	$K=2, L=4, F=32, T=4$
SCCNN-Tri.	97.4±0.9	$K=3, L=4, F=32, T=4$
MPSN-Tri.	99.1±0.2	$K=2, L=3, F=32$

Missing Components in SCCNN. With a focus on 2-simplex prediction with SCCNN-Node of order one, to avoid overcrowded settings, we study how each component of an SCCNN influences the prediction. We consider the following settings without: 1) "Edge-to-Node", where the projection $\mathbf{x}_{0,\mathrm{u}}$ from edge to node is not included, equivalent to GNN; 2) "Node-to-Node", where for node output, we have $\mathbf{x}_0^l = \sigma(\mathbf{H}_{0,\mathrm{u}}^l \mathbf{R}_{1,\mathrm{u}} \mathbf{x}_1^{l-1})$; 3) "Node-to-Edge", where the projection $\mathbf{x}_{1,\mathrm{d}}$ from node to edge is not included, i.e., we have $\mathbf{x}_1^l = \sigma(\mathbf{H}_1^l \mathbf{x}_1^{l-1})$; and 4) "Edge-to-Edge", where for edge output, we have $\mathbf{x}_1^l = \sigma(\mathbf{H}_{1,\mathrm{d}}^l \mathbf{R}_{1,\mathrm{d}} \mathbf{x}_0^{l-1})$.

Table 3.12: 2-Simplex prediction (SCCNN-Node without certain components or with limited inputs).

Missing Component	AUC	Parameters
–	98.7±0.5	$L=2, F=32, T=2$
Edge-to-Node	93.9±0.8	$L=5, F=32, T=2$
Node-to-Node	98.7±0.4	$L=4, F=32, T=2$
Edge-to-Edge	98.5±1.0	$L=3, F=32, T=3$
Node-to-Edge	98.8±0.3	$L=4, F=32, T=3$

Missing Input	AUC	Parameters
–	98.7±0.5	$L=2, F=32, T=2$
Node input	98.2±0.5	$L=2, F=32, T=4$
Edge input	98.1±0.4	$L=2, F=32, T=3$
Node, Edge inputs	50.0±0.0	–

From the results in Table 3.12 (Left), we see that "No Edge-to-Node", i.e., GNN, gives much worse results as it leverages no information on edges with limited expressive power. For cases with other components missing, a similar performance can be achieved, however, at a cost of the model complexity, with either a higher convolution order or a larger number of layers L, while the latter in turn degrades the stability of the SCCNNs, as discussed

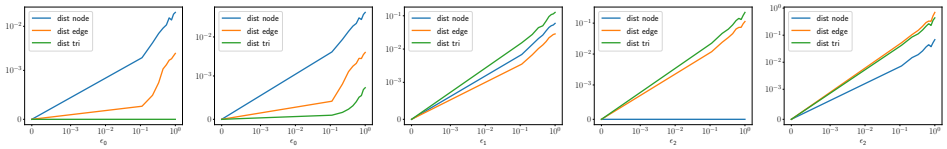
in Section 3.4. SCCNNs with certain inter-simplicial couplings pruned/missing can be powerful as well (this is similarly shown by [Bodnar et al., 2021b, Thm. 6]), but if we did not consider certain component, it comes with a cost of complexity which might degrade the model stability if more number layers are required.

Limited Input. We study the influence of limited input data for model SCCNN-Node of order two. Specifically, we consider the input on either nodes or edges is missing. From Table 3.12, we see that the prediction performance does not deteriorate at a cost of the model complexity (higher convolution orders) when a certain part of the input missing except with full zeros as input. This ability of learning from limited data shows the robustness of SCCNNs.

Stability Analysis

We then perform a stability analysis of SCCNNs. We artificially add perturbations to the normalization matrices when defining the Hogde Laplacians, which resemble the weights of simplices. We consider small perturbations $\mathbf{E}_0$ on node weights which is a diagonal matrix following that $\|\mathbf{E}_0\| \leq \epsilon_0/2$. We generate its diagonal entries from a uniform distribution $[-\epsilon_0/2, \epsilon_0/2)$ with $\epsilon_0 \in [0, 1]$, which represents one degree of deviation of the node weigths from the true ones. Similarly, perturbations on edge weights and triangle weights are applied to study the stability. In a SCCNN-Node for 2-simplex prediction of $K = 2$, we measure the distance between the simplicial outputs with and without perturbations on nodes, edges, and triangles, i.e., $\|\mathbf{x}_k^L - \hat{\mathbf{x}}_k^L\|/\|\mathbf{x}_k^L\|$, for $k = 0, 1, 2$.

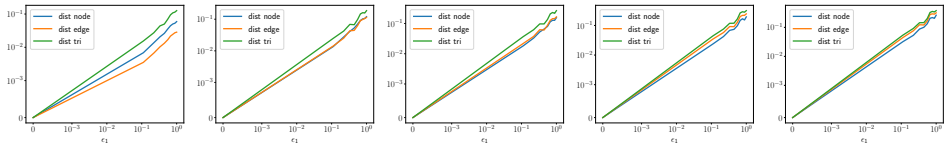
Stability dependence. We first show the stability mutual dependence between different simplices in Fig. 3.8. We see that under perturbation on node weights, the triangle output is not influenced until the number of layers becomes two; likewise, the node output is not influenced by perturbations on triangle weights with a one-layer SCCNN. Also, a one-layer SCCNN under perturbations on edge weights will cause outputs on nodes, edges, triangles perturbed. Lastly, we observe that the same degree of perturbations added to different simplices causes different degrees of instability, owing to the number n_k of k-simplices in the stability bound. Since $N_0 < N_1 < N_2$, the perturbations on node weights cause less instability than those on edge and triangle weights.



(a) Node pert. $L = 1$ (b) Node pert. $L = 2$ (c) Edge pert. $L = 1$ (d) Tri. pert. $L = 1$ (e) Tri. pert. $L = 2$

Figure 3.8: The stabilities of different simplicial outputs are dependent on each other.

Number of Layers. Fig. 3.9 shows that the stability of SCCNNs degrades as the number of layers increases as studied in Theorem 3.23. As the NN deepens, the stability deteriorates, which corresponds to our analysis of using shallow layers.



(a) $L = 1$ (b) $L = 2$ (c) $L = 3$ (d) $L = 4$ (e) $L = 5$

Figure 3.9: The relative difference of SCCNN outputs with and without perturbations in terms of different numbers of layers. We consider perturbations on edge weights.

3.F.4 Additional details on Trajectory prediction

Problem Formulation

A trajectory of length m can be modeled as a sequence of nodes $[v_0, v_1, \ldots, v_{m-1}]$ in an SC. The task is to predict the next node v_m from the neighbors of v_{m-1}, $\mathcal{N}_{v_{m-1}}$. The algorithm in Roddenberry et al. [2021] first represents the trajectory equivalently as a sequence of oriented edges $[[v_0, v_1], [v_1, v_2], \ldots, [v_{m-2}, v_{m-1}]]$. Then, an edge flow $\mathbf{x}_1$ is defined, whose value on an edge e is $[\mathbf{x}_1]_e = 1$ if edge e is traversed by the trajectory in a forward direction, $[\mathbf{x}_1]_e = -1$ if edge e is traversed in a backward direction by the trajectory, and $[\mathbf{x}_1]_e = 0$, otherwise.

With the trajectory flow $\mathbf{x}_1$ as the input, together with zero inputs on the nodes and triangles, an SCCNN of order two is used to generate a representation $\mathbf{x}_1^L$ of the trajectory, which is the output on edges. This is followed by a projection step $\mathbf{x}_{0,\mathrm{u}}^L = \mathbf{B}_1 \mathbf{W} \mathbf{x}_1^L$, where the output is first passed through a linear transformation via $\mathbf{W}$, then projected into the node space via $\mathbf{B}_1$. Lastly, a distribution over the candidate nodes $\mathcal{N}_{v_{m-1}}$ is computed via a softmax operation, $\mathbf{n}_j = \mathrm{softmax}([\mathbf{x}_{0,\mathrm{u}}^L]_j), j \in \mathcal{N}_{v_{m-1}}$. The best candidate is selected as $v_m = \mathrm{argmax}_j \mathbf{n}_j$. We refer to Roddenberry & Segarra [2019, Alg. S-2] for more details.

Given that an SCCNN of order two generates outputs also on nodes, we can directly apply the node feature output $\mathbf{x}_0^L$ to compute a distribution over the candidate nodes $\mathcal{N}_{v_{m-1}}$ without the projection step. We refer to this as SCCNN-Node, and the method of using the edge features with the projection step as SCCNN-Edge.

Model

In this experiment, we consider the following methods: 1) PSNN by Roddenberry et al. [2021]; 2) SNN by Ebli et al. [2020]; 3) SCNN by Yang et al. [2022a] where we consider different lower and upper convolution orders $T_\mathrm{d}, T_\mathrm{u}$; and 4) Bunch by Bunch et al. [2020] where we consider both the node features and edge features, namely, Bunch-Node and Bunch-Edge.

Synthetic Data. Following the procedure in Schaub et al. [2020], we generate 1000 trajectories as follows. First, we create an SC with two "holes" by uniformly drawing 400 random points in the unit square, and then a Delaunay triangulation is applied to obtain a mesh, followed by the removal of nodes and edges in two regions. To generate a trajectory, we consider a starting point at random in the lower-left corner, and then connect it via a

shortest path to a random point in the upper left, center, or lower-right region, which is connected to another random point in the upper-right corner via a shortest path.

We consider the random walk Hodge Laplacians Schaub et al. [2020]. For Bunch method, we set the shifting matrices as the simplicial adjacency matrices defined in Bunch et al. [2020]. We consider different NNs with three intermediate layers where each layer contains $F = 16$ intermediate features. The tanh nonlinearity is used such that the orientation equivariance holds. The final projection $\mathbf{n}$ generates a node feature of dimension one. In the 1000-epoch training, we use the cross-entropy loss function between the output $\mathbf{d}$ and the true candidate and we consider an adam optimizer with a learning rate of 0.001 and a batch size 100. To avoid overfitting, we apply a weight decay of $5 \cdot 10^{-6}$ and an early stopping.

As done in Roddenberry et al. [2021], besides the standard trajectory prediction task, we also perform a reverse task where the training set remains the same but the direction of the trajectories in the test set is reversed and a generalization task where the training set contains trajectories running along the upper left region and the test set contains trajectories around the other region. We evaluate the correct prediction ratio by averaging the performance over 10 different data generations.

Real Data. We also consider the Global Drifter Program dataset[3] localized around Madagascar. It consists of ocean drifters whose coordinates are logged every 12 hours. An SC can then be created as Schaub et al. [2020] by treating each mesh as a node, connecting adjacent meshes via an edge and filling the triangles, where the "hole" is yielded by the island. Following the process in Roddenberry et al. [2021], it results in 200 trajectories and we use 180 of them for training. In the training, a batch size of 10 is used and no weight decay is used. The rest experiment setup remains the same as the synthetic case.

Results

We report the prediction accuracy of different tasks for both datasets in Table 3.13. We first investigate the effects of applying higher-order SCFs in the simplicial convolution and accounting for the lower and upper contributions. From the standard accuracy for both datasets, we observe that increasing the convolution orders improves the prediction accuracy, e.g., SCNNs become better as the orders $T_{\mathrm{d}}, T_{\mathrm{u}}$ increase and perform always better than PSNN, and SCCNNs better than Bunch. Also, differentiating the lower and upper convolutions does help improve the performance as SCNN of orders $T_{\mathrm{d}} = T_{\mathrm{u}} = 3$ performs better than SNN of $T = 3$.

However, accounting for the node and triangle contributions in SCCNNs does not help the prediction compared to the SCNNs, likewise for Bunch compared to PSNN. This is due to the zero node and triangle inputs because there are no available node and triangle features. Similarly, the prediction directly via the node output features is not accurate compared to projection from edge features.

Moreover, we also observe that the performance of SCCNNs that are trained with the same data does not deteriorate in the reverse task because the orientation equivariance ensures

[3] http://www.aoml.noaa.gov/envids/gld/,

SCCNNs to be unaffected by the orientations of the simplicial data. Lastly, we see that, like other NNs on SCs, SCCNNs have good transferability to the unseen data.

Table 3.13: Trajectory Prediction Accuracy. *(Left)*: Synthetic trajectory in the standard, reverse and generalization tasks. *(Right)*: Ocean drifter trajectories. For SCCNNs, we set the lower and upper convolution orders $T_{\mathrm{d}}, T_{\mathrm{u}}$ to be the same as T.

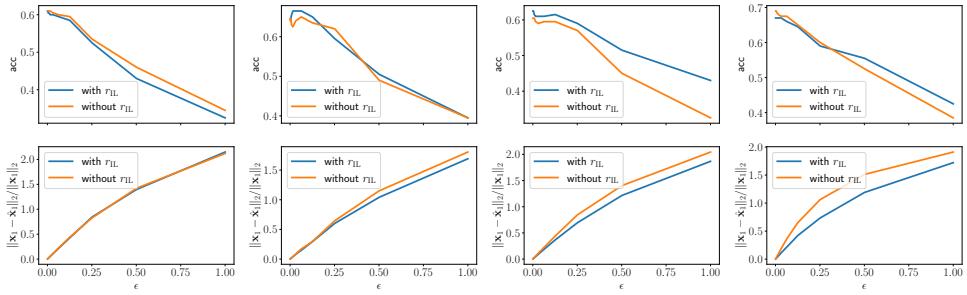
Methods	Standard	Reverse	Generalization	Parameters	Standard	Parameters
PSNN	63.1±3.1	58.4±3.9	55.3±2.5	–	49.0±8.0	–
SCNN	65.6±3.4	56.6±6.0	56.1±3.6	$T_{\mathrm{d}} = T_{\mathrm{u}} = 2$	52.5±9.8	$T_{\mathrm{d}} = T_{\mathrm{u}} = 2$
SCNN	66.5±5.8	57.7±5.4	60.6±4.0	$T_{\mathrm{d}} = T_{\mathrm{u}} = 3$	52.5±7.2	$T_{\mathrm{d}} = T_{\mathrm{u}} = 3$
SCNN	67.3±2.3	56.9±4.8	59.4±4.2	$T_{\mathrm{d}} = T_{\mathrm{u}} = 4$	52.5±8.7	$T_{\mathrm{d}} = T_{\mathrm{u}} = 4$
SCNN	67.7±1.7	55.3±5.3	61.2±3.2	$T_{\mathrm{d}} = T_{\mathrm{u}} = 5$	53.0±7.8	$T_{\mathrm{d}} = T_{\mathrm{u}} = 5$
SNN	65.5±2.4	53.6±6.1	59.5±3.7	$T = 3$	52.5±6.0	$T = 3$
Bunch-Node	35.4±3.4	38.1±4.6	29.0±3.0	–	35.0±5.9	–
Bunch-Edge	62.3±4.0	59.6±6.1	53.9±3.1	–	46.0±6.2	–
SCCNN-Node	46.8±7.3	44.5±8.2	31.9±5.0	$T = 1$	40.5±4.7	$T = 1$
SCCNN-Edge	64.6±3.9	57.2±6.3	54.0±3.0	$T = 1$	52.5±7.2	$T = 1$
SCCNN-Node	43.5±9.6	44.4±7.6	32.8±2.6	$T = 2$	45.5±4.7	$T = 2$
SCCNN-Edge	65.2±4.1	58.9±4.1	56.8±2.4	$T = 2$	54.5±7.9	$T = 2$

Convolution Order and Integral Lipschitz Property

Here, to illustrate that the integral Lipschitz property of the SCFs helps the stability of NNs on SCs, we consider the effect of regularizer r_{IL} against perturbations in PSNNs and SCNNs with different T_{d} and T_{u} for the standard synthetic trajectory prediction. The regularization weight on r_{IL} is set as $5 \cdot 10^{-4}$ and the number of samples to approximate the frequencies is set such that the sampling interval is 0.01.

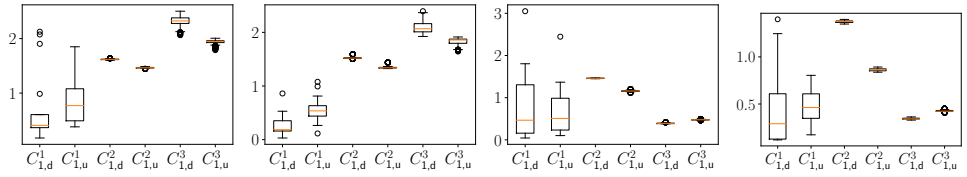
Fig. 3.10 shows the prediction accuracy and the relative distance between the edge outputs of the NNs trained with and without the integral Lipschitz regularizer in terms of different levels of perturbations. We see that the integral Lipschitz regularizer helps the stability of the NNs, especially for large SCF orders, where the edge output is less influenced by the perturbations compared to without the regularizer. Meanwhile, SCNN with higher-order SCFs, e.g., $T_{\mathrm{d}} = T_{\mathrm{u}} = 5$, achieves better prediction than PSNN (with one-step simplicial shifting), while maintaining a good stability with its output not influenced by perturbations drastically.

We also measure the lower and upper integral Lipschitz constants of the trained NNs across different layers and features, given by $\max_{\lambda_{k,\mathrm{G}}} |\lambda_{k,\mathrm{G}} \tilde{h}_{k,\mathrm{G}}(\lambda_{k,\mathrm{G}})|$ and $\max_{\lambda_{k,\mathrm{C}}} |\lambda_{k,\mathrm{C}} \tilde{h}_{k,\mathrm{C}}(\lambda_{k,\mathrm{C}})|$, shown in Fig. 3.11. We see that the SCNN trained with r_{IL} indeed has smaller integral Lipschitz constants than the one trained without the regularizer, thus, a better stability, especially for NNs with higher-order SCFs.



(a) PSNN (b) SCNN13 (c) SCNN31 (d) SCNN55

Figure 3.10: Effect of the integral Lipschitz regularizer r_{IL} in the task of synthetic trajectory prediction against different levels ϵ of random perturbations on $\mathbf{L}_{1,\mathrm{d}}$ and $\mathbf{L}_{1,\mathrm{u}}$. We show the accuracy (Top row) and the relative distance between the edge output (Bottom row) for different NNs on SCs with and without r_{IL}. SCNN13 is the SCNN with $T_{\mathrm{d}} = 1$ and $T_{\mathrm{u}} = 3$.



(a) SCNN31, without r_{IL} (b) SCNN31, with r_{IL} (c) SCNN55, without r_{IL} (d) SCNN55, with r_{IL}

Figure 3.11: The integral Lipschitz constants of SCFs at each layer of the trained SCNNs with and without the integral Lipschitz regularizer r_{IL}. We use symbols $c^l_{k,\mathrm{d}}$ and $c^l_{k,\mathrm{u}}$ to denote the lower and upper integral Lipschitz constants at layer l. Regularizer r_{IL} promotes the integral Lipschitz property, thus, the stability, especially for NNs with large SCF orders.

4

Hodge-Compositional Edge Gaussian Processes

In Chapters 2 and 3, we have developed the processing and learning of signals on simplicial complexes based on the simplicial convolution operator. Yet, these methods remain deterministic, composing of the first part of this thesis. To quantify the uncertainty of the learning, we consider a probabilistic setting from now on. When viewing simplicial signals as random variables, we first consider the Gaussian case. In this chapter, we focus on the edge flows in simplicial 2-complexes, and introduce edge Gaussian processes (GPs) that are designed to model the gradient, curl and harmonic components of edge functions in a Hodge-compositional manner. This allows them to independently learn each component, providing increased expressiveness and interpretability when compared to directly extending edge GPs from graph GPs. This chapter is based on the work of Yang et al. [2024].

4.1 Introduction

Gaussian processes (GPs) are a widely used class of statistical models capable of quantifying uncertainty associated to their own predictions [Rasmussen & Williams, 2006]. These models are determined by covariance kernels which encode prior knowledge about the unknown prediction function. Choosing an appropriate kernel is often challenging, particularly when the input space is non-Euclidean [Duvenaud, 2014].

Developing GPs on graphs has been a subject of recent work, which requires structured kernels to encode the dependence between nodes [Venkitaraman et al., 2020; Zhi et al., 2023], like the diffusion [Smola & Kondor, 2003] or random walk kernels [Vishwanathan et al., 2010]. More recently, Borovitskiy et al. [2021] derived the more general family of Matérn kernels on graphs from stochastic partial differential equations (SPDEs) thereon, mirroring the continuous approaches on manifolds [Azangulov et al., 2022; 2023; Borovitskiy et al., 2020]. Nikitin et al. [2022] incorporated the temporal factor in this framework to build temporal-graph kernels. However, GPs in these works are targeted for modeling

functions on the nodes of networks.

We instead focus on functions defined on the *edges*, of particular interest for modeling edge-based dynamical processes in many complex networks, such as flows of energy, signal or mass [Schaub et al., 2014]. For example, in water supply networks, we typically monitor the flow rates within pipes (edges) connecting tanks (nodes) [Zhou et al., 2022]. Other examples include energy flows in power grids [Jia et al., 2019], synaptic signals between neurons in brain networks [Faskowitz et al., 2022], and exchange rates on trading paths (edges) of currencies (nodes) [Jiang et al., 2011].

While it might seem intuitive to use node-based methods for edge-based tasks using line-graphs [Godsil & Royle, 2001], this often yields sub-optimal solutions [Jia et al., 2019]. Alternatively, recent successes in signal processing and neural networks for edge data have emerged from modeling flows on the edge set of a simplicial 2-complex (SC_2), including [Barbarossa & Sardellitti, 2020; Isufi & Yang, 2022; Jia et al., 2019; Roddenberry et al., 2021; Schaub et al., 2021; Yang et al., 2022b], among others. A SC_2 can be viewed as a graph with the additional set of triangular faces, encoding how edges are adjacent to each other via nodes or faces. A SC_2 also allows to characterize key properties of edge flows using discrete concepts of *divergence* (div) and *curl* [Lim, 2020; Lovász, 2004], measuring how they diverge at nodes and circulate along faces. For example, electric currents in circuit networks respecting the Kirchhoff's law are div-free [Grady & Polimeni, 2010], and arbitrage-free exchange rates are curl-free along a loop of trading paths [Jiang et al., 2011]. Moreover, edge functions on a SC_2 admit the *Hodge decomposition* into three parts: gradient, curl and harmonic components, being curl-free, div-free or both, respectively [Lim, 2020]. This provides unique insights in various applications including ranking [Jiang et al., 2011], gaming theory [Candogan et al., 2011], brain networks [Vijay Anand et al., 2022] and finance [Fujiwara & Islam, 2020]. Nevertheless, existing works on edge-based learning remain deterministic and there is a lack of principled ways to define GP priors on the edge set of SCs, which is the central goal of this work.

Our main contribution lies in the proposal of *Hodge-compositional edge GPs*. We build them as combinations of three GPs, each modeling a specific part of the Hodge decomposition of an edge function, namely the gradient, curl and harmonic parts. With a focus on the Matérn family, we show that each of them can be linked to a SPDE, extending the framework used by Borovitskiy et al. [2020; 2021; 2023]. Compared to a direct extension of graph GPs, they enable separate learning of the different Hodge components, which allows us to capture the practical behavior of edge flows. We also demonstrate their practical potential in edge-based learning tasks in foreign currency exchange markets, ocean flow analysis and water supply networks.

4.2 Background

A random function $f : \mathcal{X} \to \mathbb{R}$ defined over a set $\mathcal{X}$ is a Gaussian process $f \sim \mathrm{GP}(\mu, k)$ with mean function $\mu(\cdot)$ and kernel $k(\cdot, \cdot)$ if, for any finite set of points $\mathbf{x} = [x_1, \ldots, x_n]^\top \in \mathcal{X}^n$, the random vector $f(\mathbf{x}) = [f(x_1), \ldots, f(x_n)]^\top$ is multivariate Gaussian with mean vector $\mu(\mathbf{x})$ and covariance matrix $k(\mathbf{x}, \mathbf{x})$.

The kernel k of a *prior* GP encodes prior knowledge about the unknown function while its mean μ is usually assumed to be zero. GP regression combines such a *prior* with training data $(x_1, y_1), \ldots, (x_n, y_n)$ where $x_i \in \mathcal{X}$, $y_i \in \mathbb{R}$ with $y_i = f(x_i) + \epsilon_i$, $\epsilon_i \sim \mathcal{N}(0, \sigma_\epsilon^2)$. This results in a posterior $f_{|\mathbf{y}}$ which is another GP: $f_{|\mathbf{y}} \sim \mathrm{GP}(\mu_{|\mathbf{y}}, k_{|\mathbf{y}})$. For any new input $x^* \in \mathcal{X}$, the mean $\mu_{|\mathbf{y}}(x^*)$ is the prediction and the posterior variance $k_{|\mathbf{y}}(x^*, x^*)$ quantifies the uncertainty. We refer to Rasmussen & Williams [2006] for more details. Defining an appropriate kernel is one of the main challenges in GP modeling [Duvenaud, 2014]. For example, to define a RBF kernel, one often requires a distance function on the input space, which is not always available especially in non-Euclidean spaces.

4.2.1 GPs on Graphs

Let $\mathcal{G} = (\mathcal{V}, \mathcal{E})$ be an unweighted graph where $\mathcal{V} = \{1, \ldots, N_0\}$ is the set of nodes and $\mathcal{E}$ is the set of N_1 edges such that if nodes i, j are connected, then $e = \{i, j\} \in \mathcal{E}$. We can define real-valued functions on its node set $f_0 : \mathcal{V} \to \mathbb{R}$, collected into a vector $\mathbf{f}_0 = (f_0(1), \ldots, f_0(N_0))^\top \in \mathbb{R}^{N_0}$. Denote the node-to-edge incidence matrix by $\mathbf{B}_1$ of dimension $N_0 \times N_1$. Its entries are $[\mathbf{B}_1]_{ie} = -1$ and $[\mathbf{B}_1]_{je} = 1$, and zero otherwise, for edge $e = \{i, j\}$. The *graph Laplacian* is then given by $\mathbf{L}_0 = \mathbf{B}_1 \mathbf{B}_1^\top$, which is a positive semi-definite linear operator on the space $\mathbb{R}^{N_0}$ of node functions. It admits an eigendecomposition $\mathbf{L}_0 = \mathbf{U}_0 \mathbf{\Lambda}_0 \mathbf{U}_0^\top$ where $\mathbf{\Lambda}_0$ collects its eigenvalues on the diagonal and $\mathbf{U}_0$ collects the orthogonal eigenvectors of $\mathbf{L}_0$ [Chung, 1997].

A GP on graphs $\mathbf{f}_0 \sim \mathrm{GP}(\mathbf{0}, \mathbf{K}_0)$ assumes $\mathbf{f}_0$ is a random function with zero mean and a graph kernel $\mathbf{K}_0$ which encodes the covariance between pairs of nodes. To construct justified graph GPs, Borovitskiy et al. [2021] extended the idea of deriving continuous GPs from SPDEs[1] [Lindgren et al., 2011; Whittle, 1963] to the domain of graphs. Specifically, given the following SPDE on graphs with a Gaussian noise $\mathbf{w}_0 \sim \mathcal{N}(\mathbf{0}, \mathbf{I})$

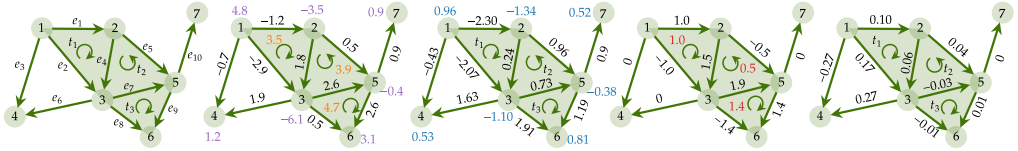
$$\Phi(\mathbf{L}_0)\mathbf{f}_0 = \mathbf{w}_0, \text{ with } \Phi(\mathbf{L}_0) = \left(\tfrac{2\nu}{\kappa^2}\mathbf{I} + \mathbf{L}_0\right)^{\frac{\nu}{2}}, \tag{4.1}$$

where $\Phi(\mathbf{L}_0) = \mathbf{U}_0 \Phi(\mathbf{\Lambda}_0) \mathbf{U}_0^\top$ and $\Phi(\cdot)$ applies to $\mathbf{\Lambda}_0$ element-wise. Its solution gives the Matérn graph GP

$$\mathbf{f}_0 \sim \mathrm{GP}\left(\mathbf{0}, \left(\tfrac{2\nu}{\kappa^2}\mathbf{I} + \mathbf{L}_0\right)^{-\nu}\right) \tag{4.2}$$

with positive parameters κ, ν. When scaled properly, the Matérn kernel gives the graph diffusion kernel for $\nu \to \infty$, which in turn relates to the random walk kernel by Kondor & Lafferty [2002]. This SPDE framework can be extended to spatial-temporal data yielding respective graph kernels [Nikitin et al., 2022].

[1] Stocahstic partial differential equations generalize partial differential equations via random force terms and coefficients, in the same way ordinary stochastic differential equations generalize ordinary differential equations. For example, the stochastic heat equation may be written as $\partial_t x = \Delta x + \xi$ where Δ is the Laplacian (not to be confused with the graph Laplacian) and ξ denotes space-time random noise.



(a) SC_2 (b) $\mathbf{f}_1$ (c) $\mathbf{f}_G$ (d) $\mathbf{f}_C$ (e) $\mathbf{f}_H$

Figure 4.1: (a) A SC_2 where we shade (closed) triangles in green and denote reference orientations of edges/triangles by arrows. (b) An edge function $\mathbf{f}_1$ with its divergence (purple values on nodes) and curl (orange values in triangles). (c) Gradient flow $\mathbf{f}_G = \mathbf{B}_1^\top \mathbf{f}_0$ ($\mathbf{f}_0$ denoted in blue). (c-d) Hodge decomposition: (c) $\mathbf{f}_G$; (d) curl flow $\mathbf{f}_C = \mathbf{B}_2\mathbf{f}_2$ ($\mathbf{f}_2$ denoted in red); and (e) harmonic flow $\mathbf{f}_H$. All numbers are rounded to two decimal places.

4

4.2.2 Edge Functions on Simplicial Complexes

Simplicial 2-complexes represent discrete geometry more expressively than graphs. They are triples $\mathrm{SC}_2 = (\mathcal{V}, \mathcal{E}, \mathcal{T})$ where $\mathcal{V}, \mathcal{E}$ are the sets of nodes and edges, same as for graphs, and $\mathcal{T}$ is the set of triangular faces (shortened as triangles) such that if $\{i,j\}, \{j,k\}, \{i,k\}$ form a *closed* triangle, then $t = \{i,j,k\} \in \mathcal{T}$ [Munkres, 2018]. An example is shown in Fig. 4.1a. We assume a fixed *orientation* for each edge and each triangle by increasing labels of their nodes. An oriented edge, denoted as $e = [i,j]$, is an ordering of $\{i,j\}$. This is not a directed edge allowing flow only from i to j, but rather an assignment of the sign of the flow: from i to j is positive and the reverse is negative. Same goes for oriented triangles denoted as $t = [i,j,k]$.

Given a SC_2, the functions, $f_1 : \mathcal{E} \to \mathbb{R}$, on its edges $\mathcal{E}$ are required to be *alternating* [Lim, 2020], meaning that, we have $f_1(\bar{e}) = -f_1(e)$ if $\bar{e} = [j,i]$ is oriented opposite to reference $e = [i,j]$. For example, in Fig. 4.1b, $f_1(1,2) = -1.2$ means there is 1.2 unit of flow from 2 to 1. This property keeps the flow unchanged with respect to the edge orientation. We collect the edge functions on $\mathcal{E}$ into $\mathbf{f}_1 = (f_1(e_1), \ldots, f_1(e_{N_1}))^\top \in \mathbb{R}^{N_1}$, as in Fig. 4.1b, which we also call as an *edge flow*.

We can also define alternating functions on triangles in $\mathcal{T}$ where $f_2(\bar{t}) = -f_2(t)$ if $\bar{t}$ is an odd permutation of reference $t = [i,j,k]$ [Lim, 2020]. We collect them in $\mathbf{f}_2 \in \mathbb{R}^{N_2}$ where $N_2 = |\mathcal{T}|$. In topology, functions f_0, f_1, f_2 are called *0-, 1-, 2-cochains*, which are discrete analogs of differential forms on manifolds [Grady & Polimeni, 2010]. This motivates the use of subscripts $0, 1, 2$. Here we can view these functions as vectors of data on nodes, edges and triangles, respectively.

4.3 Edge Gaussian Processes

We now define GPs on edges of a SC_2, specifically, $\mathbf{f}_1 \sim \mathrm{GP}(\mathbf{0}, \mathbf{K}_1)$ with zero mean and edge kernel $\mathbf{K}_1$. Throughout this work, we refer to them as *edge GPs*, and call graph GPs in Section 4.2.1 as *node GPs* because they are both multivariate Gaussian but the former is indexed by $\mathcal{X} = \mathcal{E}$ and the latter by $\mathcal{X} = \mathcal{V}$. We start with deriving edge GPs from SPDEs on edges as a natural extension of (4.1). Then, by introducing basic notions from discrete calculus [Grady & Polimeni, 2010] and the Hodge decomposition theorem, we propose the divergence-free and curl-free GPs, combining them into Hodge-compositional GPs.

4.3.1 Edge GPs from SPDEs on Edges

The derivation of graph GPs in (4.2) as solutions of graph SPDE in (4.1) motivates the following SPDEs on edges, with edge Gaussian noise $\mathbf{w}_1 \sim \mathcal{N}(\mathbf{0}, \mathbf{I})$,

$$\Phi(\mathbf{L}_1)\mathbf{f}_1 = \mathbf{w}_1 \tag{4.3}$$

where $\Phi(\mathbf{L}_1) = \mathbf{U}_1 \Phi(\mathbf{\Lambda}_1)\mathbf{U}_1^\top$ is a differential operator defined through $\mathbf{L}_1$. When we consider the operators

$$\Phi(\mathbf{L}_1) = \left(\frac{2\nu}{\kappa^2}\mathbf{I} + \mathbf{L}_1\right)^{\frac{\nu}{2}}, \text{ and } \Phi(\mathbf{L}_1) = e^{\frac{\kappa^2}{4}\mathbf{L}_1}, \tag{4.4}$$

the solutions to (4.3) give two edge GPs

$$\mathbf{f}_{1,\text{Matérn}} \sim \mathrm{GP}\left(\mathbf{0}, \left(\frac{2\nu}{\kappa^2}\mathbf{I} + \mathbf{L}_1\right)^{-\nu}\right), \text{ and } \mathbf{f}_{1,\text{diffusion}} \sim \mathrm{GP}\left(\mathbf{0}, e^{-\frac{\kappa^2}{2}\mathbf{L}_1}\right), \tag{4.5}$$

which are the *edge Matérn* and *diffusion* GPs, respectively. These edge GPs impose structured prior covariance that encodes the dependence between edges. A related *Hodge Laplacian kernel* $(\mathbf{L}_1^\top \mathbf{L}_1)^\dagger$ can be obtained by setting $\Phi(\mathbf{L}_1) = \mathbf{L}_1$, i.e., $\mathbf{L}_1\mathbf{f}_1 = \mathbf{w}_1$. This kernel was used to penalize the smoothness of edge functions in Schaub et al. [2021]. The kernels of (4.5) are more flexible though and allow encoding non-local edge-to-edge adjacency while $\mathbf{L}_1$ instead encodes the local direct (one-hop) adjacency.

4.3.2 Div-free and Curl-free Edge GPs

The edge GPs in Section 4.3.1 define distributions over all edge functions. As opposed to this, here we seek to define GPs on the classes of divergence-free and curl-free edge functions. Recall the physics interpretations of incidence matrices in Chapter 2, which carry the appropriate notions of discrete derivatives.

Discrete Derivatives. The *gradient* is a linear operator from the space of node functions to that of edge functions. At edge $e = [i, j]$, it is defined as

$$(\mathrm{grad}\, f_0)(e) = (\mathbf{B}_1^\top \mathbf{f}_0)_e = f_0(j) - f_0(i), \tag{4.6}$$

which computes the difference between the values of a function on adjacent nodes, resulting in a flow on the connecting edge. We call $\mathbf{f}_G = \mathbf{B}_1^\top \mathbf{f}_0$ a gradient flow and $\mathbf{f}_0$ a node potential, as shown in Fig. 4.1c.

The *divergence*, the adjoint of gradient, is a linear operator from the space of edge functions to that of node functions. At node i, it is defined as

$$(\mathrm{div}\, f_1)(i) = (\mathbf{B}_1\mathbf{f}_1)_i = -\textstyle\sum_{j \in \mathcal{N}(i)} f_1(i, j) \tag{4.7}$$

with $\mathcal{N}(i)$ the neighbors of i. Physically, it computes the net-flow of edge functions passing through node i, i.e., the in-flow minus the out-flow, as shown in Fig. 4.1b. A *divergence-free* flow has a zero net-flow everywhere.

Lastly, the *curl* operator is a linear operator from the space of edge functions to that of triangle functions. At triangle $t = [i, j, k]$, it is defined as

$$(\operatorname{curl} f_1)(t) = (\mathbf{B}_2^\top \mathbf{f}_1)_t = f_1(i,j) + f_1(j,k) - f_1(i,k) \tag{4.8}$$

which computes the *net-circulation* of edge functions along the edges of t, as a rotational measure of $\mathbf{f}_1$, as shown in Fig. 4.1b. A *curl-free* flow has zero curl over each triangle. As in calculus, we have the identity $\operatorname{curl}\operatorname{grad} = \mathbf{B}_2^\top \mathbf{B}_1^\top = \mathbf{0}$, i.e., gradient flow is curl-free.

Analogous to their continuous vector field counterparts, div-free and curl-free edge functions are ubiquitous, e.g., the electric currents and the exchange rates later in Section 4.4.1. We refer to Grady & Polimeni [2010]; Lim [2020] for more examples. From this perspective, we can view the graph Laplacian as $\mathbf{L}_0 = \operatorname{div}\operatorname{grad} = \mathbf{B}_1\mathbf{B}_1^\top$, which is a graph-theoretic analog of the Laplace-Beltrami operator Δ_0 on manifolds. Also, the SPDE on graphs in (4.1) is a discrete counterpart of the continuous one for scalar functions on manifolds. Moreover, the Hodge Laplacian $\mathbf{L}_1$ can be viewed as $\mathbf{L}_1 = \operatorname{grad}\operatorname{div} + \operatorname{curl}^* \operatorname{curl} = \mathbf{B}_1^\top \mathbf{B}_1 + \mathbf{B}_2\mathbf{B}_2^\top$, which is a discrete analog of the vector Laplacian (or Helmholtzian) Δ_1 for vector fields.

Recall the *Hodge decomposition* in Theorem 2.4 that unfolds an edge function. It states that any edge function $\mathbf{f}_1$ is composed of three orthogonal parts: gradient, curl, harmonic functions

$$\mathbf{f}_1 = \mathbf{f}_G + \mathbf{f}_H + \mathbf{f}_C \tag{4.9}$$

where $\mathbf{f}_G = \mathbf{B}_1^\top \mathbf{f}_0$, being curl-free, is the gradient of some node function $\mathbf{f}_0$, and $\mathbf{f}_C = \mathbf{B}_2\mathbf{f}_2$, being div-free, is the curl-adjoint of some triangle function $\mathbf{f}_2$. Lastly, $\mathbf{f}_H$ is harmonic (both div- and curl-free, $\mathbf{L}_1\mathbf{f}_H = \mathbf{0}$). This decomposition is illustrated in Fig. 4.1. It provides a crucial tool for understanding edge functions, has been used in many applications as we discussed above, and will allow us to improve the edge GPs in (4.5).

Furthermore, Proposition 2.5 shows that the eigenspace $\mathbf{U}_1$ of $\mathbf{L}_1$ can be reorganized in terms of the three Hodge subspaces as

$$\mathbf{U}_1 = [\mathbf{U}_H \; \mathbf{U}_G \; \mathbf{U}_C] \tag{4.10}$$

where $\mathbf{U}_H$ is the eigenvector matrix associated to zero eigenvalues $\mathbf{\Lambda}_H = \mathbf{0}$ of $\mathbf{L}_1$, $\mathbf{U}_G$ is associated to the nonzero eigenvalues $\mathbf{\Lambda}_G$ of $\mathbf{L}_\mathrm{d}$, and $\mathbf{U}_C$ is associated to the nonzero eigenvalues $\mathbf{\Lambda}_C$ of $\mathbf{L}_\mathrm{u}$. Moreover, they span the Hodge subspaces:

$$\operatorname{span}(\mathbf{U}_H) = \ker(\mathbf{L}_1),\ \operatorname{span}(\mathbf{U}_G) = \operatorname{im}(\mathbf{B}_1^\top),\ \operatorname{span}(\mathbf{U}_C) = \operatorname{im}(\mathbf{B}_2), \tag{4.11}$$

where $\operatorname{span}(\bullet)$ denotes all possible linear combinations of columns of $\bullet$.

Div-free, Curl-free Edge GPs. Given the eigendecomposition in (4.10), we can obtain special classes of edge GPs by only using a certain type of eigenvectors when building edge kernels of (4.5). Specifically, we define *gradient* and *curl edge GPs* as follows

$$\mathbf{f}_G \sim \mathrm{GP}(\mathbf{0}, \mathbf{K}_G), \quad \mathbf{f}_C \sim \mathrm{GP}(\mathbf{0}, \mathbf{K}_C) \tag{4.12}$$

where the gradient kernel and the curl kernel are

$$\mathbf{K}_G = \mathbf{U}_G \Psi_G(\mathbf{\Lambda}_G)\mathbf{U}_G^\top,\ \mathbf{K}_C = \mathbf{U}_C \Psi_C(\mathbf{\Lambda}_C)\mathbf{U}_C^\top. \tag{4.13}$$

We also define the *harmonic GPs* $\mathbf{f}_H \sim \mathrm{GP}(\mathbf{0}, \mathbf{K}_H)$ with the harmonic kernel $\mathbf{K}_H = \mathbf{U}_H \Psi_H(\mathbf{\Lambda}_H)\mathbf{U}_H^\top$.

Proposition 4.1. *Let $\mathbf{f}_G$ and $\mathbf{f}_C$ be the gradient and curl Gaussian processes, respectively. Then, $\operatorname{curl}\mathbf{f}_G = \mathbf{0}$ and $\operatorname{div}\mathbf{f}_C = \mathbf{0}$ with probability one. Moreover, a harmonic Gaussian process $\mathbf{f}_H$ follows $\operatorname{curl}\mathbf{f}_H = \mathbf{0}$ and $\operatorname{div}\mathbf{f}_H = \mathbf{0}$ with probability one.*

See proof in Appendix 4.A.2. These Hodge GPs provide more targeted priors for special edge functions which are either div- or curl-free, capable of capturing these key properties. In the case of Matérn kernels, we set

$$\Psi_\square(\mathbf{\Lambda}_\square) = \sigma_\square^2 \Big(\frac{2\nu_\square}{\kappa_\square^2}\mathbf{I} + \mathbf{\Lambda}_\square \Big)^{-\nu_\square}, \tag{4.14}$$

for $\square \in \{H, G, C\}$, where $\sigma_\square^2$ controls the variance we assign to the function in the subspace, and $\nu_\square, \kappa_\square$ are the regular Matérn parameters. Note that since $\mathbf{\Lambda}_H = \mathbf{0}$, we consider a scaling function for $\mathbf{K}_H$ as $\Psi_H(\mathbf{0}) = \sigma_H^2$. We illustrate such a Matérn kernel function in Fig. 4.2 (left). These Hodge GPs can be derived from SPDEs on edges as well.

Proposition 4.2. *Given a scaled curl white noise $\mathbf{w}_C \sim \mathcal{N}(\mathbf{0}, \mathbf{W}_C)$ where $\mathbf{W}_C = \sigma_C^2 \mathbf{U}_C \mathbf{U}_C^\top$, consider the following SPDE on edges:*

$$\Phi_C(\mathbf{L}_\mathrm{u})\mathbf{f}_C = \mathbf{w}_C, \tag{4.15}$$

with differential operators

$$\Phi_C(\mathbf{L}_\mathrm{u}) = \Big(\frac{2\nu_C}{\kappa_C^2}\mathbf{I} + \mathbf{L}_\mathrm{u} \Big)^{\frac{\nu_C}{2}}, \; \Phi_C(\mathbf{L}_\mathrm{u}) = e^{\frac{\kappa_C^2}{4}\mathbf{L}_\mathrm{u}}. \tag{4.16}$$

The respective solutions give the curl edge GPs with Matérn kernel in (4.14) and diffusion kernel

$$\Psi_C(\mathbf{\Lambda}_C) = \sigma_C^2 e^{-\frac{\kappa_C^2}{2}\mathbf{\Lambda}_C}. \tag{4.17}$$

Likewise, we can derive the gradient Matérn and diffusion GPs from the SPDEs as (4.15) but with operators $\Phi_G(\mathbf{L}_\mathrm{d})$ and a scaled gradient white noise.

See proof in Appendix 4.A.3. We can draw the intuition of SPDE in (4.15) from the continuous analogy. In the case of $\mathbf{L}_\mathrm{u}\mathbf{f}_C = \mathbf{w}_C$, the equation $\operatorname{curl}^* \operatorname{curl} f_1(\mathbf{x}) = w_1(\mathbf{x})$ is a stochastic vector Laplace's equation of a div-free (solenoidal) vector field, where $w_1(\mathbf{x})$ the curl adjoint of some vector potential. In physics, this describes the static magnetic field from a magnetic vector potential, as well as an incompressible fluid.

4.3.3 Hodge-compositional Edge GPs

Many edge functions of interest are indeed div- or curl -free, but not all. In this section we combine the gradient, curl and harmonic GPs to define the Hodge-compositional (HC) edge GPs.

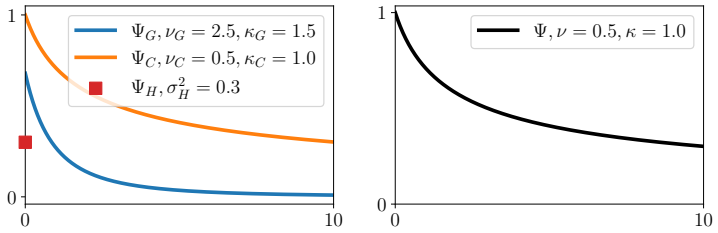


Figure 4.2: *(Left)* Matérn kernel functions $\Psi_\square(\lambda)$ for $\square = \{H, G, C\}$ in (4.14) of gradient, curl and harmonic GPs in the eigen-spectrum λ ranging in the min and man eigenvalues of $\mathbf{L}_1$. *(Right)* Matérn kernel function $\Psi(\lambda)$ of non-HC GP in (4.5).

4

Definition 4.3. A Hodge-compositional edge Gaussian process $\mathbf{f}_1 \sim \mathrm{GP}(\mathbf{0}, \mathbf{K}_1)$ is a sum of gradient, curl and harmonic GPs, i.e., $\mathbf{f}_1 = \mathbf{f}_G + \mathbf{f}_C + \mathbf{f}_H$ where

$$\mathbf{f}_\square \sim \mathrm{GP}(\mathbf{0}, \mathbf{K}_\square) \text{ with } \mathbf{K}_\square = \mathbf{U}_\square \Psi_\square(\mathbf{\Lambda}_\square)\mathbf{U}_\square^\top \tag{4.18}$$

for $\square = H, G, C$ where their kernels do not share hyperparameters.

Given this definition, we can obtain the following properties of HC edge GPs.

Lemma 4.4. *Let $\mathbf{f}_1 \sim \mathrm{GP}(\mathbf{0}, \mathbf{K}_1)$ be an edge GP in Definition 4.3. Its realizations then give all possible edge functions*[2]*. It further holds that $\mathbf{K}_1 = \mathbf{K}_H + \mathbf{K}_G + \mathbf{K}_C$, and the three Hodge GPs are mutually independent, i.e., $\mathrm{Cov}(\mathbf{f}_G, \mathbf{f}_C) = \mathrm{Cov}(\mathbf{f}_G, \mathbf{f}_H) = \mathrm{Cov}(\mathbf{f}_C, \mathbf{f}_H) = \mathbf{0}$.*

See proof in Appendix 4.A.4. Naturally, we can construct a Matérn HC GP as the sum of Matérn GPs in the three subspaces with their kernels given by (4.14), and likewise for the diffusion HC GP by (4.17). Compared to the GPs in (4.5), referred to as non-HC GPs henceforth, HC GPs are more flexible and expressive, having more degrees of freedom. We discuss their practical advantages below.

Inductive GP Prior. The HC GP encodes the prior covariance $\mathrm{Cov}(f_1(e), f_1(e'))$ between edge functions over two edges e, e' as follows: (i) the covariance is the sum of three covariances $\mathrm{Cov}_\square = \mathrm{Cov}(f_\square(e), f_\square(e'))$ for $\square = H, G, C$; (ii) each $\mathrm{Cov}_\square$ encodes the covariance between the corresponding Hodge parts of f_1 without affecting the others; and (iii) no covariance is imposed across different Hodge components, e.g., $\mathrm{Cov}(f_G(e), f_C(e')) = 0$.

In the spatial/edge domain, this is related to separating the down and up adjacencies encoded in the SPDE operators $\Phi(\cdot)$. From an eigen-spectrum perspective, the eigenvalues $\Psi_\square$ of HC GP's kernels associated to the three Hodge subspaces have individual parameters. This enables capturing the different Hodge components of edge functions, as well as their relevance during hyperparameter optimization, further allowing us to directly recover the Hodge components in predictions, which we detail in Appendix 4.A.5. Non-HC GPs instead require solving the Hodge decomposition in (4.9) (least squares problems) [Lim,

[2]Note that HC edge GPs do not represent all possible edge GPs. They are a particular GP family satisfying the independence hypothesis on the three Hodge GPs. This, however, does not contradict with that their realizations can represent all edge functions.

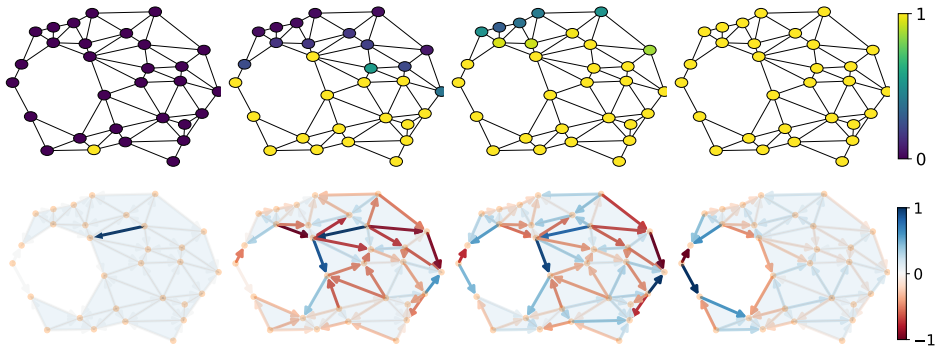


Figure 4.3: Node *(Top)* and edge *(Bottom)* diffusion processes (started at one random location *(Left)*, then two intermediate states *(Center)* and harmonic state *(Right)*).

2020]. Another implication is that, unlike for non-HC GPs, we do not require specific knowledge about the div or curl of the underlying function.

Comparison to non-HC GPs. When we view non-HC GPs in terms of the Hodge decomposition, we notice that they put priors on the three Hodge GPs in a way that shares hyperparameters. This enforces learning the same hyperparameters for different Hodge components, resulting in a single function covering the entire edge spectrum, as shown in Fig. 4.2 (right), as opposed to the three individual functions of the HC one.

This raises issues when separate learning, say, different lengthscales, is required for the gradient and curl components. Non-HC GPs are strictly incapable of this practical need when an eigenvalue is associated to both gradient and curl spaces. We also delve into this in terms of *edge Fourier features* in the following.

Connection to Diffusion on Edges.

The HC diffusion kernel, given by $\mathbf{K}_1 = \exp(-(\frac{\kappa_G^2}{2}\mathbf{L}_\mathrm{d} + \frac{\kappa_C^2}{2}\mathbf{L}_\mathrm{u}))$, when $\sigma_\square^2$s are one, is the Green's function for the edge diffusion of a function $\boldsymbol{\phi} : [0, \infty) \times \mathcal{E} \to \mathbb{R}$

$$\frac{\mathrm{d}\boldsymbol{\phi}(t)}{\mathrm{d}t} = -(\mu\mathbf{L}_\mathrm{d} + \gamma\mathbf{L}_\mathrm{u})\boldsymbol{\phi}(t), \text{ where } \mu, \gamma > 0 \tag{4.19}$$

with the solution $\boldsymbol{\phi}(t)|_{t=\tau} = e^{-(\mu\tau\mathbf{L}_\mathrm{d} + \gamma\tau\mathbf{L}_\mathrm{u})}\boldsymbol{\phi}(0)$. This equation describes the diffusion process on the edge space of SC_2 that was used for network analysis [DeVille, 2021; Muhammad & Egerstedt, 2006], often arising as the limit of random walks on edges [Schaub et al., 2020]. The covariance $\mathbf{K}_1$ within this context encodes the proportion of edge flow traveling from edge e to e' via down and up edge adjacencies. Its vector field counterpart was used for shape analysis [Sharp et al., 2019; Zobel et al., 2011]. Compared to the graph (node) diffusion converging ($t \to \infty$) to the state that is constant on all nodes as long as the graph is connected, the harmonic state of the edge diffusion can be non-constant, but lies in the span of $\mathbf{U}_H$. We refer to Fig. 4.3 for visualizations of the two harmonic states.

Complexity. The kernels of HC edge GPs can be constructed in a scalable way by consid-

ering the l largest eigenvalues with off-the-shelf eigen-solvers, e.g., Lanczos algorithm. We refer to Appendix 4.A.8 for more details on the complexity of implementing HC GPs, as well as sampling from them.

4.3.4 Edge Fourier Feature Perspective

From the edge eigen-feature perspective, any edge function can be viewed as a linear combination of eigenvectors in $\mathbf{U}$, that is,

$$\mathbf{f}_1 = \sum_{i=1}^{N_1} \tilde{f}_{1,i}\mathbf{u}_i = \mathbf{U}_1\tilde{\mathbf{f}}_1 \text{ with } \tilde{\mathbf{f}}_1 = \mathbf{U}^\top\mathbf{f}_1 \tag{4.20}$$

4

where $\tilde{\mathbf{f}}_1$ is known as the (edge) Fourier feature of $\mathbf{f}_1$ and $\tilde{f}_{1,i}$ is the i-th Fourier coefficient at eigenvalue λ_i. These eigenvalues carry the notion of frequency [cf. Section 2.4.2]. Based on the correspondence in Proposition 2.5 between the Hodge subspaces and the eigenbasis $\mathbf{U}_1$ [cf. (4.10)], we futher have the edge Fourier representation as

$$\tilde{\mathbf{f}}_1 = \mathbf{U}_1^\top\mathbf{f}_1 = [\tilde{\mathbf{f}}_H^\top, \tilde{\mathbf{f}}_G^\top, \tilde{\mathbf{f}}_C^\top]^\top \text{ with } \tilde{\mathbf{f}}_H = \mathbf{U}_H^\top\mathbf{f}_1,\ \tilde{\mathbf{f}}_G = \mathbf{U}_G^\top\mathbf{f}_1,\ \tilde{\mathbf{f}}_C = \mathbf{U}_C^\top\mathbf{f}_1. \tag{4.21}$$

This provides as a spectral tool to understand the edge functions. The harmonic Fourier feature $\tilde{\mathbf{f}}_H$ measures the extent of harmonic Fourier basis $\mathbf{U}_H$ in $\mathbf{f}$, reflecting how harmonic $\mathbf{f}_1$ is. The gradient Fourier feature $\tilde{\mathbf{f}}_G$ measures the extent of gradient Fourier basis $\mathbf{U}_G$ in $\mathbf{f}_1$, reflecting how divergent $\mathbf{f}_1$ is, where each basis in $\mathbf{U}_G$ has different total divergence. The curl Fourier feature $\tilde{\mathbf{f}}_C$ measures the extent of curl Fourier basis $\mathbf{U}_C$ in $\mathbf{f}_1$, reflecting how rotational $\mathbf{f}_1$ is, where each basis in $\mathbf{U}_C$ has different total curl.

Corollary 4.5 (Fourier feature perspective of edge GPs)**.** *Let $\mathbf{f}_1 \sim \mathrm{GP}(\mathbf{0}, \mathbf{K}_1)$ be an edge Gaussian process with kernel diagonalizable by $\mathbf{U}_1$. Then, given the edge Fourier transform $\tilde{\mathbf{f}}_1 = \mathbf{U}_1^\top\mathbf{f}_1$, its Fourier coefficients $\{\tilde{f}_{1,i}\}_{i=1}^N$ are independently distributed Gaussian variables*

$$\tilde{f}_{1,i} \sim \mathcal{N}(0, \mathbf{u}_i^\top\mathbf{K}_1\mathbf{u}_i),\ \textit{for } i = 1, \ldots, N. \tag{4.22}$$

This corollary indicates that an edge GP can be viewed as an affine transformation by $\mathbf{U}_1$ of a collection of independent Gaussian variables, $\tilde{\mathbf{f}}_1 = [\tilde{f}_{1,1}, \ldots, \tilde{f}_{1,N_1}]^\top$, which are the Fourier coefficients of $\mathbf{f}$. The prior distribution of certain Fourier coefficient is the prior imposed on the corresponding divergent or rotational part of the function $\mathbf{f}$. This allows us to compare HC and non-HC edge GPs from the following perspective.

Proposition 4.6. *Suppose the Hodge Laplacian $\mathbf{L}_1$ has eigenpairs $(\lambda, \mathbf{u}_G)$ and $(\lambda, \mathbf{u}_C)$, i.e., λ is associated to both gradient and curl subspaces. Let $\mathbf{f}_1 \sim \mathrm{GP}(\mathbf{0}, \mathbf{K}_1)$ be an edge Gaussian process. Denote the Fourier coefficients of $\mathbf{f}_1$ at λ_G and λ_C as $\tilde{f}_G$ and $\tilde{f}_C$, respectively. Then, a non-Hodge-compositional GP with $\mathbf{K}_1 = \Psi(\mathbf{L}_1)$ imposes the same prior variance on $\tilde{f}_G$ and $\tilde{f}_C$, i.e.,*

$$\mathrm{Var}[\tilde{f}_G] = \mathrm{Var}[\tilde{f}_C] = \Psi(\lambda). \tag{4.23}$$

Instead, a Hodge-compositional GP with $\mathbf{K}_1$ in (4.18) imposes different variances on two coefficients

$$\mathrm{Var}[\tilde{f}_G] = \Psi_G(\lambda) \textit{ and } \mathrm{Var}[\tilde{f}_C] = \Psi_C(\lambda). \tag{4.24}$$

This edge Fourier feature perspective directly shows that non-HC GPs impose the same prior on two Fourier coefficients, which are however associated with two different Hodge subspaces. This prohibits individual learning for the gradient and curl parts of edge functions particularly associated to the same eigenvalue. Instead, HC edge GPs do not have this limitation.

4.3.5 Node-Edge-Triangle GP Interactions

The gradient and curl components of edge functions are (co)derivatives of some node and triangle functions, specifically, $\mathbf{f}_G = \mathbf{B}_1^\top \mathbf{f}_0$ and $\mathbf{f}_C = \mathbf{B}_2 \mathbf{f}_2$ as in (4.9). Since the derivative of a GP is also a GP, we can then construct a gradient GP from node GPs.

Corollary 4.7. *Suppose a node function* $\mathbf{f}_0$ *is a GP* $\mathbf{f}_0 \sim \mathcal{GP}(\mathbf{0}, \mathbf{K}_0)$ *with* $\mathbf{K}_0 = \Psi_0(\mathbf{L}_0) = \mathbf{U}_0 \Psi_0(\mathbf{\Lambda}_0) \mathbf{U}_0^\top$. *Then, its gradient is an edge GP* $\mathbf{f}_G \sim \mathcal{GP}(\mathbf{0}, \mathbf{K}_G)$ *where* $\mathbf{K}_G = \mathbf{B}_1^\top \mathbf{K}_0 \mathbf{B}_1 = \mathbf{U}_G \Psi_G(\mathbf{\Lambda}_G) \mathbf{U}_G^\top$ *with*

$$\Psi_G(\mathbf{\Lambda}_G) = \mathbf{\Lambda}_G \Psi_0(\mathbf{\Lambda}_G). \tag{4.25}$$

The proof follows from (i) derivatives preserving Gaussianity, and (ii) $\mathbf{L}_0$ and $\mathbf{L}_\mathrm{d}$ having the same nonzero eigenvalues. We can also obtain a curl edge GP from a GP on triangles likewise. In turn, for an edge GP, its div is a node GP and its curl is a GP on triangles. We refer to Appendix 4.A.9 for the proof and more details.

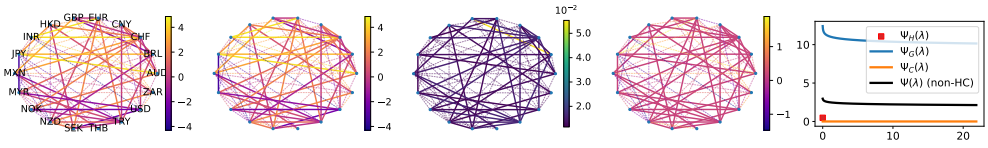
Exploiting this interaction between GPs on nodes, edges and triangles can lead to new useful GPs, especially when functions on nodes, edges and triangles are intrinsically related by physical laws. For example, in water networks, water flowrates in pipes are often related to the gradient of hydraulic heads on nodes, as we will show in Section 4.4.3. This implies that given an appropriate node GP, say, node Matérn GP in (4.2), a good edge GP prior can be imposed as its gradient, as in Corollary 4.7. Furthermore, by leveraging this interaction, we can construct HC edge GPs as follows.

Proposition 4.8. *Let* $\mathbf{f}_1$ *be an edge function defined in (4.9) with harmonic component* $\mathbf{f}_H$, *node function* $\mathbf{f}_0$ *and triangle function* $\mathbf{f}_2$. *If we model* $\mathbf{f}_0$ *as a GP on nodes* $\mathbf{f}_0 \sim \mathcal{GP}(\mathbf{0}, \mathbf{K}_0)$, *model* $\mathbf{f}_2$ *as a GP on triangles* $\mathbf{f}_2 \sim \mathcal{GP}(\mathbf{0}, \mathbf{K}_2)$, *and* $\mathbf{f}_H$ *as a harmonic GP* $\mathbf{f}_H \sim \mathcal{GP}(\mathbf{0}, \mathbf{K}_H)$, *then we have GP* $\mathbf{f}_1 \sim \mathcal{GP}(\mathbf{0}, \mathbf{K}_1)$ *with*

$$\mathbf{K}_1 = \mathbf{K}_H + \mathbf{B}_1^\top \mathbf{K}_0 \mathbf{B}_1 + \mathbf{B}_2 \mathbf{K}_2 \mathbf{B}_2^\top. \tag{4.26}$$

See proof in Appendix 4.A.10. This alternative HC GP incorporates the Hodge theorem prior in a way that directly relates the node potential and the triangle function. It can be applicable when GP priors of node or triangle functions are more discernible. Similar ideas for general cellular complexes are studied in the concurrent paper by Alain et al. [2024].

Continuous Counterparts. Edge functions can be viewed as discrete analogs of vector fields. Berlinghieri et al. [2023] studied the models similar to our HC edge GPs in (4.26) for Euclidean vector fields and the concurrent work by Robert-Nicoud et al. [2024] studies similar models for vector fields on manifolds.



(a) Ground truth (b) Mean, HC Matérn (c) Standard deviation (d) Mean, Non-HC (e) Learned kernels

Figure 4.4: (a-d) Interpolating a smaller forex market (for better visibility) with train ratio 50% where dashed (solid) edges are used for training (testing). (e) Learned Matérn kernels in the spectrum of the Hodge Laplacian, $\Psi(\lambda)$ for non-HC GP and $\Psi_\square(\lambda)$ with $\square = \{H, G, C\}$ for HC GPs.

4.4 Experiments

We apply HC GPs for edge-based inference tasks in three applications: foreign currency exchange (forex), ocean current and water supply networks (WSNs). We showcase the structured prior on edges in these tasks by comparing them to baselines: (i) Euclidean GPs with RBF and Matérn kernels, and (ii) Node GPs on the line-graph—built by exchanging the nodes with edges in the original graph [Godsil & Royle, 2001]. To highlight the prior of Hodge decomposition, we also compare with non-HC GPs. For each of them, we consider Matérn and diffusion kernels. We perform GP regression with Gaussian likelihood for model fitting using the `GPyTorch` framework [Gardner et al., 2018]. We use the root mean squared error (RMSE) to evaluate the predictive mean and the negative log predictive density (NLPD) for prediction uncertainty. We refer to Appendix 4.B for full experimental details.

4.4.1 Foreign Currency Exchange

A forex market can be modeled as a network where nodes represent currencies and edges the exchangeable pairs [Jiang et al., 2011]. Forex rates in a fair market ideally satisfy the *arbitrage-free* condition: for any currencies i, j, k, we have $r^{i/j} r^{j/k} = r^{i/k}$ with $r^{i/j}$ the rate between i and j. That is, the exchange path $i \to j \to k$ provides no gain or loss over a direct path $i \to k$. If we model forex rates as edge flows $f_1(i,j) = \log(r^{i/j})$, this condition can be translated into that $\mathbf{f}_1$ is a gradient flow, being curl-free, i.e., $f_1(i,j) + f_1(j,k) - f_1(i,k) = 0$. Here we consider real-world forex data on 2018/10/05 with 25 most traded currencies forming 210 exchangeable pairs and 710 triangles, formed by any three pairwise exchangeable currencies [Jia et al., 2019; Oanda, 2018]. We randomly sample 20% of edges for training and test on the rest.

From Table 4.1, we see that HC GPs achieve significantly lower RMSEs with high certainty (small NLPDs), as visualized in Fig. 4.4. This shows their ability to automatically capture the curl-free nature of the forex rates. As shown in Fig. 4.4e, the HC Matérn GP learns that harmonic and curl components should vanish. In contrast, the other three give poor predictions, due to: (i) Euclidean GPs being oblivious of the structure of edge functions; (ii) line-graph GPs imposing inappropriate structure through node priors; and (iii) non-HC GPs being unable to induce the curl-free prior without removing the gradient. This results from sharing parameters in their kernels for different Hodge components. As shown in

Table 4.1: Forex rates inference results.

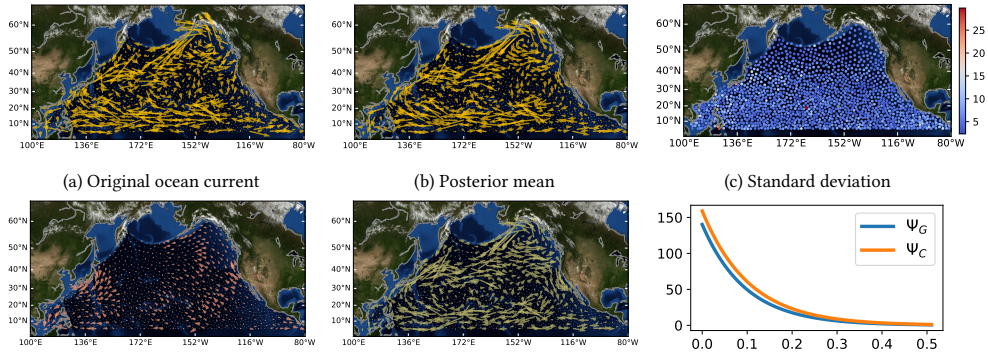
Method	RMSE		NLPD	
	Diffusion	Matérn	Diffusion	Matérn
Euclidean	2.17 ± 0.13	2.19 ± 0.12	2.12 ± 0.07	2.20 ± 0.18
Line-Graph	2.43 ± 0.07	2.46 ± 0.07	2.28 ± 0.04	2.32 ± 0.03
Non-HC	2.48 ± 0.07	2.47 ± 0.08	2.36 ± 0.07	2.34 ± 0.04
HC	0.08 ± 0.12	0.06 ± 0.12	-3.52 ± 0.02	-3.52 ± 0.02

Fig. 4.4e, the non-HC Matérn GP learns a nonzero kernel in the whole spectrum, unable to remove the non-arbitrage-free part.

4.4.2 Ocean Current Analysis

We consider the edge-based ocean current learning following the setup in Chen et al. [2021c]. The ocean current velocity fields were converted using the linear integration approximation to edge flows within a SC_2 whose nodes are 1500 buoys sampled from North Pacific ocean drifter records in 2010-2019 [Lumpkin & Centurioni, 2019]. We apply both non-HC and HC GPs to predict the converted edge flows. Given the large number of edges (∼20k), we consider a truncated approximation of kernels with eigenpairs associated with the 500 largest eigenvalues [Knyazev, 2001]. We randomly sample 20% of edges for training and test on the rest.

From Table 4.2, we see that HC and non-HC GPs exhibit similar performance. This arises from the comparable behavior of the gradient and curl components, as depicted in Fig. 4.5f, where the learned gradient and curl diffusion kernels display close patterns. In contrast, Euclidean and line-graph GPs give poor predictions emphasizing the importance of structured edge priors.



(a) Original ocean current (b) Posterior mean (c) Standard deviation

(d) Curl-free component (e) Div-free component (f) Learned HC kernel

Figure 4.5: (a-b) Ground truth and interpolated ocean current in the vector field domain. (c) Standard deviation approximated by sampling 50 edge flows from the predictive posterior distribution and converted to the vector field domain. (d-e) The curl-free and div-free components directly obtained from the learned kernels. (f) Learned diffusion kernels $\Psi_G(\lambda)$ and $\Psi_C(\lambda)$ of the HC GP in the spectrum of the Hodge Laplacian.

Table 4.2: Ocean current inference results.

Method	RMSE		NLPD	
	Diffusion	Matérn	Diffusion	Matérn
Euclidean	1.00 ± 0.01	1.00 ± 0.00	1.42 ± 0.01	1.42 ± 0.10
Line-Graph	0.99 ± 0.00	0.99 ± 0.00	1.41 ± 0.00	1.41 ± 0.00
Non-HC	0.35 ± 0.00	0.35 ± 0.00	0.33 ± 0.00	0.36 ± 0.03
HC	0.34 ± 0.00	0.35 ± 0.00	0.33 ± 0.01	0.37 ± 0.04

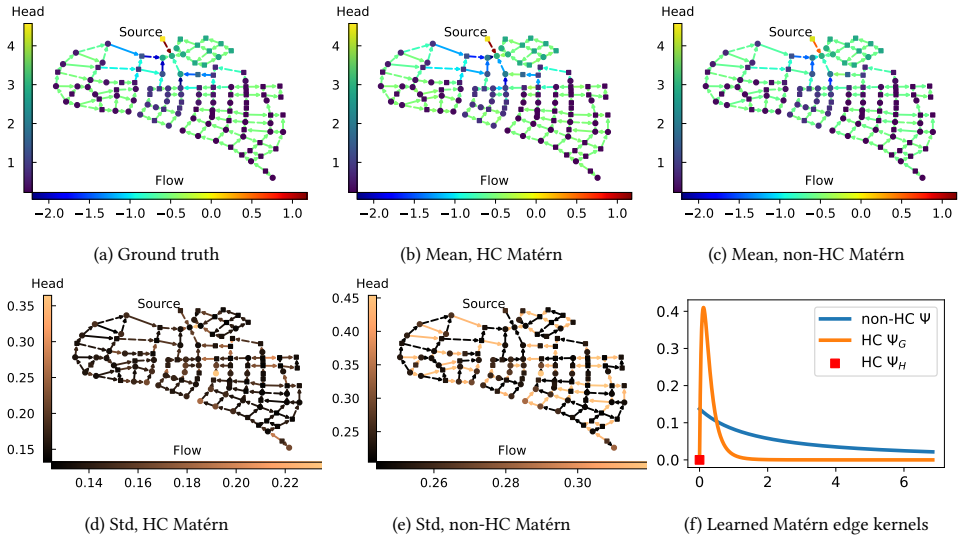
We further convert the predicted edge flows into the vector field domain, as shown in Fig. 4.5b, based on Chen et al. [2021c]. We see that the predictions capture the pattern of the original velocity field. We approximate the predicted velocity field uncertainty by computing the average ℓ_2 distance per location from 50 posterior samples to the mean in the vector field domain. As shown in Fig. 4.5c, the standard deviation estimated this way is small in most locations except for a few exceptions (small islands at the bottom left) where the original field is more discontinuous. Moreover, since HC GPs enable the direct recovery of gradient and curl components, we show their corresponding vector fields in Figs. 4.5d and 4.5e, giving better insights into how ocean currents behave, of particular interest in oceanography. For example, we can observe the well-known North Pacific gyres including the North Equatorial, Kuroshio and Alaska currents in Fig. 4.5e.

4.4.3 Water Supply Networks

Network-based methods have been used in WSNs where tanks or reservoirs are represented by nodes, and pipes by edges [Zhou et al., 2022]. By modeling the hydraulic heads as node functions $\mathbf{f}_0$ and the water flowrates as edge functions $\mathbf{f}_1$, the commonly used empirical equation connecting the two reads as $\mathbf{B}_1^\top \mathbf{f}_0 = \bar{\mathbf{f}}_1 := \operatorname{diag}(\mathbf{r}) \mathbf{f}_1^{1.852}$ where r_e is the resistance of pipe e and the exponentiation is applied element-wise [Dini & Tabesh, 2014].

We consider the Zhi Jiang WSN with 114 tanks (including one source) and 164 pipes (without triangles, Dandy [2016]) and simulate a scenario based on Klise et al. [2017]. We perform joint state estimation of heads $\mathbf{f}_0$ and the adjusted flowrates $\bar{\mathbf{f}}_1$, by modeling them as GPs on nodes and edges, respectively. To compare HC and non-HC edge GPs, for a node GP with kernel $\mathbf{K}_0$, we consider the HC GP as its gradient, as discussed in Corollary 4.7. For the non-HC one, we consider a kernel $\mathbf{K}_1$ of the same type as $\mathbf{K}_0$. We randomly sample 50% of nodes and edges for training and test on the rest.

From Table 4.3, we see that while the mean predictions of heads remain similar whether we use HC or non-HC edge GPs, the former perform better for edge flows, particularly in the pipes around the source, as shown in Figs. 4.6b and 4.6c. Moreover, HC GPs have better prediction uncertainty with smaller average NLPDs for both heads and flowrates, as illustrated in Figs. 4.6d and 4.6e. This is because HC GPs that we use share parameters with node GPs, helping to calibrate the uncertainty of head predictions. They also capture the physical prior of the pipe equation that assumes flowrates are a gradient flow. As shown in Fig. 4.6f, the HC Matérn GP learns a kernel with a trivial harmonic prior and



(a) Ground truth (b) Mean, HC Matérn (c) Mean, non-HC Matérn

(d) Std, HC Matérn (e) Std, non-HC Matérn (f) Learned Matérn edge kernels

Figure 4.6: (a-e) Posterior mean and standard deviation (std) based on the Matérn node GPs, and the HC and non-HC Matérn edge GPs. Squared (Circled) nodes represent the node samples for training (testing). Dashed (solid) edges denote the edge samples for training (testing). (f) The learned edge GP kernels in the spectrum, $\Psi(\lambda)$ for non-HC GP and $\Psi_H(0), \Psi(\lambda)$ for HC GPs.

Table 4.3: WSN inference results.

Method	Node Heads		Edge Flowrates	
	RMSE	NLPD	RMSE	NLPD
Diffusion, non-HC	0.16 ± 0.05	0.72 ± 2.06	0.32 ± 0.05	0.97 ± 1.80
Matérn, non-HC	0.16 ± 0.04	0.71 ± 2.39	0.26 ± 0.05	0.10 ± 0.13
Diffusion, HC	0.15 ± 0.04	-0.47 ± 0.14	0.22 ± 0.03	-0.20 ± 0.13
Matérn, HC	0.15 ± 0.04	-0.25 ± 0.48	0.23 ± 0.03	-0.45 ± 0.49

a nonzero gradient prior in small eigenvalues, reflecting the gradient nature of the pipe flowrates. Note that due to the randomness of training samples, the WSN, having small edge connectivity, may become disconnected, causing the significant variance in NLPDs.

4.5 Conclusion

We introduced Hodge-compositional (HC) Gaussian processes (GPs) for modeling functions on the edges of simplicial 2-complexes. These HC GPs are constructed by combining three individual GPs, each designed to capture the gradient, curl and harmonic components of edge functions through Hodge decomposition. This allows them to independently learn each component, providing increased expressiveness and interpretability when compared to directly extending edge GPs from graph GPs. They can also be constructed by leveraging the physical interactions between functions on nodes, edges and triangles. We demonstrated

their practical potential in learning real-world flow data. Finally, the HC GPs can be extended to higher-order simplicial complexes, and we focused on the edge flow modeling because of their relevance in real-world applications.

Appendix

4.A Edge Gaussian Processes

Here, we provide the additional details on Section 4.3 and the missing proofs.

4.A.1 Derivation of Edge GPs from SPDEs on Edges

Here we derive the edge Matérn and diffusion GPs in (4.5) from the two SPDEs in (4.3).

Proposition 4.9. *Given the SPDE with a general differential operator* $\Phi(\mathbf{L}_1) = \mathbf{U}_1\Phi(\mathbf{\Lambda}_1)\mathbf{U}_1^\top$ *and the stochastic Gaussian noise process* $\mathbf{w}_1 \sim \mathcal{N}(\mathbf{0}, \mathbf{I})$

$$\Phi(\mathbf{L}_1)\mathbf{f}_1 = \mathbf{w}_1, \tag{4.27}$$

its solution is an edge GP

$$\mathbf{f}_1 \sim \mathrm{GP}(\mathbf{0}, (\Phi^\top(\mathbf{L}_1)\Phi(\mathbf{L}_1))^\dagger) \tag{4.28}$$

Proof. By writing out its solution

$$\mathbf{f}_1 = \Phi^\dagger(\mathbf{L}_1)\mathbf{w}_1, \tag{4.29}$$

which is a random process, we can find its covariance as

$$\mathrm{Cov}[\mathbf{f}_1] = \Phi^\dagger(\mathbf{L}_1)\mathrm{Cov}[\mathbf{w}_1](\Phi^\dagger(\mathbf{L}_1))^\top = (\Phi^\top(\mathbf{L}_1)\Phi(\mathbf{L}_1))^\dagger \tag{4.30}$$

□

Corollary 4.10. *Matérn and diffusion edge kernels in (4.5) given as follows*

$$\mathbf{f}_1 \sim \mathrm{GP}\left(\mathbf{0}, \left(\frac{2\nu}{\kappa^2}\mathbf{I} + \mathbf{L}_1\right)^{-\nu}\right), \quad \mathbf{f}_1 \sim \mathrm{GP}\left(\mathbf{0}, e^{-\frac{\kappa^2}{2}\mathbf{L}_1}\right) \tag{4.31}$$

are the solutions of the following two SPDEs, respectively.

$$\left(\frac{2\nu}{\kappa^2}\mathbf{I} + \mathbf{L}_1\right)^{\frac{\nu}{2}}\mathbf{f}_1 = \mathbf{w}_1, \quad e^{\frac{\kappa^2}{4}\mathbf{L}_1}\mathbf{f}_1 = \mathbf{w}_1. \tag{4.32}$$

Proof. By following the procedure in Proposition 4.9, the proof completes. □

4.A.2 Samples of Gradient and Curl Edge GPs

Here, we discuss the div and curl properties of samples of gradient and curl GPs in (4.12), which completes the proof of Proposition 4.1.

Proposition 4.11. *Consider the gradient and curl GPs*

$$\mathbf{f}_G \sim \mathrm{GP}(\mathbf{0}, \mathbf{K}_G), \quad \mathbf{f}_C \sim \mathrm{GP}(\mathbf{0}, \mathbf{K}_C) \tag{4.33}$$

where the gradient kernel and the curl kernel are

$$\mathbf{K}_G = \mathbf{U}_G \Psi_G(\mathbf{\Lambda}_G) \mathbf{U}_G^\top, \quad \mathbf{K}_C = \mathbf{U}_C \Psi_C(\mathbf{\Lambda}_C) \mathbf{U}_C^\top. \tag{4.34}$$

Their prior samples are, respectively, curl-free and div-free.

4

Proof. We focus on the case of gradient GPs. First, we can decompose the gradient kernel in terms of $\mathbf{U}_1 = [\mathbf{U}_H \ \mathbf{U}_G \ \mathbf{U}_C]$ as

$$\mathbf{K}_G = \mathbf{U}_1 \begin{pmatrix} \mathbf{0} & & \\ & \Psi_G(\mathbf{\Lambda}_G) & \\ & & \mathbf{0} \end{pmatrix} \mathbf{U}_1^\top. \tag{4.35}$$

From a vector $\mathbf{v} = (v_1, \ldots, v_{N_1})^\top$ of variables following independent normal distribution, we can draw a random sample of gradient function as

$$\mathbf{f}_G = \mathbf{U}_1 \mathrm{diag}([\mathbf{0}, \Psi_G^{\frac{1}{2}}(\mathbf{\Lambda}_G), \mathbf{0}]) \mathbf{v} \tag{4.36}$$

where $\mathrm{diag}([\mathbf{a}, \mathbf{b}, \mathbf{c}])$ is the diagonal matrix with $(\mathbf{a}, \mathbf{b}, \mathbf{c})^\top$ on its diagonal.

Therefore, their curls are

$$\mathrm{curl}\, \mathbf{f}_G = \mathbf{B}_2^\top \mathbf{U}_1 \mathrm{diag}([\mathbf{0}, \Psi_G^{\frac{1}{2}}(\mathbf{\Lambda}_G), \mathbf{0}]) = \mathbf{B}_2^\top \mathbf{U}_G \Psi_G^{\frac{1}{2}}(\mathbf{\Lambda}_G) = \mathbf{0}. \tag{4.37}$$

Likewise, we can show the samples of a curl GP are div-free.

Remark 4.12. An alternative proof can follow by studying the curl of the gradient GP which is another GP on triangles as given later by Proposition 4.16. The kernel $\mathbf{B}_2^\top \mathbf{K}_G \mathbf{B}_2$ is zero, due to the orthogonality $\mathbf{B}_2^\top \mathbf{U}_G = \mathbf{0}$. Thus, the curl of a gradient GP is a zero GP on triangles, as well as its samples. Similarly, one can show the div of a curl GP is a zero GP on nodes, thus, its samples are zero.

□

4.A.3 Derivation of Gradient and Curl GPs from SPDEs

Here we provide proofs for Proposition 4.2, deriving Matérn and diffusion gradient/curl GPs from their SPDE representations.

Proposition 4.13. *Given a scaled curl white noise* $\mathbf{w}_C \sim \mathcal{N}(\mathbf{0}, \mathbf{W}_C)$ *where* $\mathbf{W}_C = \sigma_C^2 \mathbf{U}_C \mathbf{U}_C^\top$, *consider the following SPDE on edges:*

$$\Phi_C(\mathbf{L}_\mathrm{u})\mathbf{f}_C = \mathbf{w}_C, \tag{4.38}$$

with differential operators

$$\Phi_C(\mathbf{L}_\mathrm{u}) = \Big(\frac{2\nu_C}{\kappa_C^2}\mathbf{I} + \mathbf{L}_\mathrm{u}\Big)^{\frac{\nu_C}{2}}, \quad \Phi_C(\mathbf{L}_\mathrm{u}) = e^{\frac{\kappa_C^2}{4}\mathbf{L}_\mathrm{u}}. \tag{4.39}$$

The respective solutions give the curl edge GPs with Matérn kernel and diffusion kernel

$$\mathbf{f}_C \sim \mathrm{GP}\Big(\mathbf{0}, \sigma_C^2 \mathbf{U}_C \Big(\frac{2\nu_C}{\kappa_C^2}\mathbf{I} + \mathbf{L}_\mathrm{u}\Big)^{-\nu_C} \mathbf{U}_C^\top\Big), \quad \mathbf{f}_C \sim \mathrm{GP}\Big(\mathbf{0}, \sigma_C^2 \mathbf{U}_C e^{-\frac{\kappa_C^2}{2}} \mathbf{U}_C^\top\Big). \tag{4.40}$$

Proof. First, consider the Matérn curl GP case. The corresponding SPDE has the form

$$\Big(\frac{2\nu_C}{\kappa_C^2}\mathbf{I} + \mathbf{L}_\mathrm{u}\Big)^{\frac{\nu_C}{2}} \mathbf{f}_C = \mathbf{w}_C, \tag{4.41}$$

with a solution $\mathbf{f}_C = \Phi_C^\dagger(\mathbf{L}_\mathrm{u})\mathbf{w}_C$.

Given the scaled curl Gaussian noise process $\mathbf{w}_C \sim \mathcal{G}(\mathbf{0}, \mathbf{W}_C)$ with $\mathbf{W}_C = \sigma_C^2 \mathbf{U}_C \mathbf{U}_C^\top$, the solution $\mathbf{f}_C$ is an edge GP following $\mathbf{f}_C \sim \mathrm{GP}(\mathbf{0}, \mathrm{Cov}[\mathbf{f}_C])$ with the covariance of solution $\mathbf{f}_C$ as

$$\mathrm{Cov}[\mathbf{f}_C] = \Big(\frac{2\nu_C}{\kappa_C^2}\mathbf{I} + \mathbf{L}_\mathrm{u}\Big)^{-\frac{\nu_C}{2}} \mathbf{W}_C \Big(\frac{2\nu_C}{\kappa_C^2}\mathbf{I} + \mathbf{L}_\mathrm{u}\Big)^{-\frac{\nu_C}{2}}. \tag{4.42}$$

Note that we have

$$\mathbf{W}_C = \begin{pmatrix}\mathbf{U}_H & \mathbf{U}_G & \mathbf{U}_C\end{pmatrix} \begin{pmatrix}\mathbf{0} & & \\ & \mathbf{0} & \\ & & \sigma_C^2\mathbf{I}\end{pmatrix} \begin{pmatrix}\mathbf{U}_H & \mathbf{U}_G & \mathbf{U}_C\end{pmatrix}^\top. \tag{4.43}$$

Moreover, $\mathbf{L}_\mathrm{u}$ can be decomposed by $\mathbf{U}_1$ as follows

$$\mathbf{L}_\mathrm{u} = \begin{pmatrix}\mathbf{U}_H & \mathbf{U}_G & \mathbf{U}_C\end{pmatrix} \begin{pmatrix}\mathbf{0} & & \\ & \mathbf{0} & \\ & & \mathbf{\Lambda}_C\end{pmatrix} \begin{pmatrix}\mathbf{U}_H & \mathbf{U}_G & \mathbf{U}_C\end{pmatrix}^\top, \tag{4.44}$$

which follows that

$$\begin{aligned}\Big(\frac{2\nu_C}{\kappa_C^2}\mathbf{I} + \mathbf{L}_\mathrm{u}\Big)^{-\frac{\nu_C}{2}} &= \begin{pmatrix}\mathbf{U}_H & \mathbf{U}_G & \mathbf{U}_C\end{pmatrix} \\ &\begin{pmatrix}\Big(\frac{2\nu_C}{\kappa_C^2}\mathbf{I}\Big)^{-\frac{\nu_C}{2}} & & \\ & \Big(\frac{2\nu_C}{\kappa_C^2}\mathbf{I}\Big)^{-\frac{\nu_C}{2}} & \\ & & \Big(\frac{2\nu_C}{\kappa_C^2}\mathbf{I} + \mathbf{\Lambda}_C\Big)^{-\frac{\nu_C}{2}}\end{pmatrix} \begin{pmatrix}\mathbf{U}_H & \mathbf{U}_G & \mathbf{U}_C\end{pmatrix}^\top.\end{aligned} \tag{4.45}$$

By plugging (4.43) and (4.45) into (4.42), we can then express the covariance as

$$\begin{aligned}\mathrm{Cov}[\mathbf{f}_C] &= \begin{pmatrix}\mathbf{U}_H & \mathbf{U}_G & \mathbf{U}_C\end{pmatrix}\begin{pmatrix}\mathbf{0} & & \\ & \mathbf{0} & \\ & & \sigma_C^2\left(\frac{2\nu_C}{\kappa_C^2}\mathbf{I}+\mathbf{\Lambda}_C\right)^{-\nu_C}\end{pmatrix}\begin{pmatrix}\mathbf{U}_H & \mathbf{U}_G & \mathbf{U}_C\end{pmatrix}^\top \\ &= \mathbf{U}_C\sigma_C^2\left(\frac{2\nu_C}{\kappa_C^2}\mathbf{I}+\mathbf{\Lambda}_C\right)^{-\nu_C}\mathbf{U}_C^\top\end{aligned} \tag{4.46}$$

which returns the Matérn curl GP $\mathbf{f}_C \sim \mathrm{GP}\left(\mathbf{0}, \sigma_C^2\mathbf{U}_C\left(\frac{2\nu_C}{\kappa_C^2}\mathbf{I}+\mathbf{L}_\mathrm{u}\right)^{-\nu_C}\mathbf{U}_C^\top\right)$.

Second, consider the following SPDE

$$e^{\frac{\kappa_C^2}{4}\mathbf{L}_\mathrm{u}}\mathbf{f}_C = \mathbf{w}_C. \tag{4.47}$$

Following the same procedure as above, we have its solution as

$$\mathbf{f}_C \sim \mathrm{GP}\left(\mathbf{0}, \sigma_C^2\mathbf{U}_C e^{-\frac{\kappa_C^2}{2}}\mathbf{U}_C^\top\right) \tag{4.48}$$

which is the diffusion curl GP. □

Proposition 4.14. *Given a scaled gradient white noise $\mathbf{w}_G \sim \mathcal{N}(\mathbf{0}, \mathbf{W}_G)$ where $\mathbf{W}_G = \sigma_G^2\mathbf{U}_G\mathbf{U}_G^\top$, consider the following SPDE on edges:*

$$\Phi_G(\mathbf{L}_\mathrm{d})\mathbf{f}_G = \mathbf{w}_G, \tag{4.49}$$

with differential operators

$$\Phi_G(\mathbf{L}_\mathrm{d}) = \left(\frac{2\nu_G}{\kappa_G^2}\mathbf{I}+\mathbf{L}_\mathrm{d}\right)^{\frac{\nu_G}{2}}, \quad \Phi_G(\mathbf{L}_\mathrm{d}) = e^{\frac{\kappa_G^2}{4}\mathbf{L}_\mathrm{d}}. \tag{4.50}$$

The respective solutions give the curl edge GPs with Matérn kernel and diffusion kernel

$$\mathbf{f}_G \sim \mathrm{GP}\left(\mathbf{0}, \sigma_G^2\mathbf{U}_G\left(\frac{2\nu_G}{\kappa_G^2}\mathbf{I}+\mathbf{L}_\mathrm{d}\right)^{-\nu_G}\mathbf{U}_G^\top\right) \quad \mathbf{f}_G \sim \mathrm{GP}\left(\mathbf{0}, \sigma_G^2\mathbf{U}_G e^{-\frac{\kappa_G^2}{2}}\mathbf{U}_G^\top\right). \tag{4.51}$$

Proof. The proof follows Proposition 4.13 likewise. □

4.A.4 Proof of Properties of HC Edge GPs

Here we provide proofs for Lemma 4.4, which directly follow from Definition 4.3.

Proof. For an edge GP $\mathbf{f}_1$ with covariance kernel $\mathbf{K}_1$, due to the fact that the HC edge kernel $\mathbf{K}_1$ is built using all the orthonormal basis of the edge function space $\mathbf{U}_1$, its realizations give all possible edge functions. This is analogous to Karhunen-Loève theorem for GPs with Mercer kernels. For the second point that $\mathbf{K}_1 = \mathbf{K}_H + \mathbf{K}_G + \mathbf{K}_C$ and the three Hodge GPs mutually independent, this results from the construction of $\mathbf{f}_1$ and the orthogonality of the three Hodge GPs. □

4.A.5 Posterior Distributions of Hodge Components

Here we discuss the posterior distribution of the three Hodge components from the posterior prediction of the edge function. As the construction of our HC edge GPs is essentially a sum of three independent functions, we can follow Duvenaud [2014, Section 2.4] modeling the sums of Euclidean functions. Denote $\mathbf{f}_1(\mathbf{x})$ and $\mathbf{f}_1(\mathbf{x}^*)$ the function values, respectively, at training locations $\mathbf{x} = [x_1, \ldots, x_n]^\top$ and query locations $\mathbf{x}^* = [x_1^*, \ldots, x_n^*]^\top$. We first write down the joint prior distribution over the three Hodge components and the edge function.

$$\begin{bmatrix} \mathbf{f}_H(\mathbf{x}) \\ \mathbf{f}_H(\mathbf{x}^*) \\ \mathbf{f}_G(\mathbf{x}) \\ \mathbf{f}_G(\mathbf{x}^*) \\ \mathbf{f}_C(\mathbf{x}) \\ \mathbf{f}_C(\mathbf{x}^*) \\ \mathbf{f}_1(\mathbf{x}) \\ \mathbf{f}_1(\mathbf{x}^*) \end{bmatrix} \sim \mathcal{N}\left(\mathbf{0}, \begin{bmatrix} \mathbf{K}_H & \mathbf{K}_H^* & & & & & \mathbf{K}_H & \mathbf{K}_H^* \\ \mathbf{K}_H^{*\top} & \mathbf{K}_H^{**} & & & & & \mathbf{K}_H^* & \mathbf{K}_H^{**} \\ & & \mathbf{K}_G & \mathbf{K}_G^* & & & \mathbf{K}_G & \mathbf{K}_G^* \\ & & \mathbf{K}_G^\top & \mathbf{K}_G^{**} & & & \mathbf{K}_G^* & \mathbf{K}_G^{**} \\ & & & & \mathbf{K}_C & \mathbf{K}_C^* & \mathbf{K}_C & \mathbf{K}_C^* \\ & & & & \mathbf{K}_C^\top & \mathbf{K}_C^{**} & \mathbf{K}_C^* & \mathbf{K}_C^{**} \\ \mathbf{K}_H & \mathbf{K}_H^* & \mathbf{K}_G & \mathbf{K}_G^* & \mathbf{K}_C & \mathbf{K}_C^* & \mathbf{K}_1 & \mathbf{K}_1^* \\ \mathbf{K}_H^{*\top} & \mathbf{K}_H^{**} & \mathbf{K}_G^{*\top} & \mathbf{K}_G^{**} & \mathbf{K}_C^{*\top} & \mathbf{K}_C^{**} & \mathbf{K}_1^{*\top} & \mathbf{K}_1^{**} \end{bmatrix} \right) \tag{4.52}$$

where we represent the kernel matrices by $\mathbf{K}_1 = k_1(\mathbf{x}, \mathbf{x}), \mathbf{K}_1^* = k_1(\mathbf{x}, \mathbf{x}^*)$ and $\mathbf{K}_1^{**} = k_1(\mathbf{x}^*, \mathbf{x}^*)$, and likewise for the other kernel matrices. Given this joint distribution, we can obtain the posterior distributions of the three Hodge components as follows

$$\mathbf{f}_H(\mathbf{x}^*)|\mathbf{f}_1(\mathbf{x}) \sim \mathcal{N}\left(\mathbf{K}_H^{*\top}\mathbf{K}_1^{-1}\mathbf{f}_1(\mathbf{x}), \mathbf{K}_H^{**} - \mathbf{K}_H^{*\top}\mathbf{K}_1^{-1}\mathbf{K}_H^*\right) \tag{4.53a}$$

$$\mathbf{f}_G(\mathbf{x}^*)|\mathbf{f}_1(\mathbf{x}) \sim \mathcal{N}\left(\mathbf{K}_G^{*\top}\mathbf{K}_1^{-1}\mathbf{f}_1(\mathbf{x}), \mathbf{K}_G^{**} - \mathbf{K}_G^{*\top}\mathbf{K}_1^{-1}\mathbf{K}_G^*\right) \tag{4.53b}$$

$$\mathbf{f}_C(\mathbf{x}^*)|\mathbf{f}_1(\mathbf{x}) \sim \mathcal{N}\left(\mathbf{K}_C^{*\top}\mathbf{K}_1^{-1}\mathbf{f}_1(\mathbf{x}), \mathbf{K}_C^{**} - \mathbf{K}_C^{*\top}\mathbf{K}_1^{-1}\mathbf{K}_C^*\right) \tag{4.53c}$$

From these posterior distributions, we can directly obtain the means and the uncertainties of the Hodge components of the predicted edge function.

4.A.6 Edge Fourier Feature Perspective

Proof of Corollary 4.5

Proof. Using the affine transformation preserving Gaussian, we have

$$\tilde{\mathbf{f}}_1 \sim \mathrm{GP}(\mathbf{0}, \mathbf{U}_1^\top \mathbf{K}_1 \mathbf{U}_1). \tag{4.54}$$

Since the kernel $\mathbf{K}_1$ can be diagonalized by $\mathbf{U}_1$, the kernel $\mathbf{U}_1^\top \mathbf{K}_1 \mathbf{U}_1$ is a diagonal matrix, implying the independence between variables in $\tilde{\mathbf{f}}_1$. Thus, a variable $\tilde{f}_{1,i}$ follows normal distribution $\mathcal{N}(0, \mathbf{u}_i^\top \mathbf{K}_1 \mathbf{u}_i)$. □

Proof of Proposition 4.6

Proof. For a non-HC edge GP with kernel $\Psi(\mathbf{L}_1)$, its Fourier coefficients $\tilde{f}_G$ and $\tilde{f}_C$ at a common λ follows the normal distribution with a variance $\Psi(\lambda)$, which follows from the nature of kernel function Ψ mapping each λ to exactly one value $\Psi(\lambda)$. However, for a HC edge GP, we have

$$\tilde{f}_G \sim \mathcal{N}(0, \mathbf{u}_G^\top \mathbf{K}_1 \mathbf{u}_G), \quad \tilde{f}_C \sim \mathcal{N}(0, \mathbf{u}_C^\top \mathbf{K}_1 \mathbf{u}_C). \tag{4.55}$$

Using Definition 4.3 [cf. (4.18)], we have $\mathbf{u}_G^\top \mathbf{K}_1 \mathbf{u}_G = \Psi_G(\lambda)$ and $\mathbf{u}_C^\top \mathbf{K}_1 \mathbf{u}_C = \Psi_C(\lambda)$, which are two different values, arising from the individually parametrized kernels $\mathbf{K}_G$ and $\mathbf{K}_C$. □

4.A.7 Diffusion on Edges

Here we provide the details on the connection of diffusion HC edge GPs to edge diffusion equations, as well as an illustration of diffusion process on edges. Consider the diffusion equation on the edge space

$$\frac{\mathrm{d}\boldsymbol{\phi}(t)}{\mathrm{d}t} = -(\mu \mathbf{L}_\mathrm{d} + \gamma \mathbf{L}_\mathrm{u})\boldsymbol{\phi}(t) \tag{4.56}$$

where $\mu, \gamma > 0$. Given an initial value $\boldsymbol{\phi}(0)$, we obtain a solution

$$\boldsymbol{\phi}|_{t=\tau} = e^{-(\mu\tau \mathbf{L}_\mathrm{d} + \gamma\tau \mathbf{L}_\mathrm{u})}\boldsymbol{\phi}(0), \tag{4.57}$$

When $\sigma_G^2 = \sigma_C^2 = \sigma_H^2 = 1$, the diffusion kernel can be written as

$$\mathbf{K}_1 = e^{-(\frac{\kappa_G^2}{2}\mathbf{L}_\mathrm{d} + \frac{\kappa_C^2}{2}\mathbf{L}_\mathrm{u})} \tag{4.58}$$

which is the Green's function of above diffusion equation. In Fig. 4.3, we illustrate the diffusion processes on nodes and on edges, started at a random location. When the graph is connected, the node diffusion converges to the harmonic state where all nodes are constant. Instead, the harmonic state of the edge diffusion gives an edge flow which is div- and curl-free, cycling around the 1-dimensional "hole" of the SC_2 [Munkres, 2018].

4.A.8 Complexity of Edge GPs

Here we discuss their complexity when training, e.g., in Gaussian process regression, and the complexity of sampling from them. Note that the complexity of graph GPs naturally apply to edge GPs.

Complexity when Training The Matérn and diffusion kernels can be trained in a scalable way. Due to their decreasing eigenvalues, we can consider the l largest eigenvalues of the kernel matrices with off-the-shelf eigen-solvers, e.g., Lanczos algorithm. The recent work on Krylov subspace methods to accelerate graph kernels by Erb [2023] can be extended to edge kernels. Moreover, other computational techniques applicable for graph GPs in Borovitskiy et al. [2021, Section 3.1] can be adopted as well.

Complexity when Sampling from Edge GPs Given an edge GP, as well as the eigenpairs for constructing the edge kernel, we can follow the procedure in Appendix 4.A.2 to sample an edge function. That is, from a vector $\mathbf{v} = (v_1, \ldots, v_{N_1})^\top$ of variables following independent normal distribution, a sample of the edge function can be given by

$$\mathbf{f}_1 = [\mathbf{U}_H \, \mathbf{U}_G \, \mathbf{U}_C] \operatorname{diag}([\Psi_H^{\frac{1}{2}}(\mathbf{\Lambda}_H), \Psi_G^{\frac{1}{2}}(\mathbf{\Lambda}_G), \Psi_C^{\frac{1}{2}}(\mathbf{\Lambda}_C)])\mathbf{v} \tag{4.59}$$

which has a complexity of $\mathcal{O}(N_1^2)$ (matrix-vector multiplication). Furthermore, the discussion on improving sampling efficiency in graph GP models by Nikitin et al. [2022, Section 4.7] naturally applies to our proposed edge GPs as well.

4.A.9 Interaction between Node, Edge and Triangle GPs

Here we provide the proof for Corollary 4.7, showing the gradient of a node GP is an edge GP.

Proof. Given a node GP $\mathbf{f}_0 \sim \mathrm{GP}(\mathbf{0}, \mathbf{K}_0)$, using the derivative of a GP is also a GP, its gradient $\mathbf{f}_G = \mathbf{B}_1^\top \mathbf{f}_0$ is an edge GP whose kernel can be found as

$$\mathbf{K}_G = \operatorname{Cov}[\mathbf{f}_G] = \mathbf{B}_1^\top \operatorname{Cov}[\mathbf{f}_0]\mathbf{B}_1 = \mathbf{B}_1^\top \mathbf{K}_0 \mathbf{B}_1. \tag{4.60}$$

By definition, $\mathbf{L}_0 = \mathbf{B}_1\mathbf{B}_1^\top$ and $\mathbf{L}_\mathrm{d} = \mathbf{B}_1^\top\mathbf{B}_1$ are isospectral, having the same nonzero eigenvalues. Furthermore, using $\mathbf{K}_0 = \Psi_0(\mathbf{L}_0)$, we can write above covariance as

$$\mathbf{K}_G = \mathbf{B}_1^\top \Psi(\mathbf{B}_1\mathbf{B}_1^\top)\mathbf{B}_1 = \mathbf{B}_1^\top\mathbf{B}_1 \Psi_0(\mathbf{B}_1^\top\mathbf{B}_1) = \mathbf{L}_\mathrm{d}\Psi_0(\mathbf{L}_\mathrm{d}) \tag{4.61}$$

where the second equality can be shown by using the definition of analytic functions of matrix [Higham, 2008, Corollary 1.34]. Furthermore, relying on the eigendecomposition

$$\mathbf{L}_\mathrm{d} = \begin{pmatrix}\mathbf{U}_H & \mathbf{U}_G & \mathbf{U}_C\end{pmatrix} \begin{pmatrix}\mathbf{0} & & \\ & \mathbf{\Lambda}_G & \\ & & \mathbf{0}\end{pmatrix} \begin{pmatrix}\mathbf{U}_H & \mathbf{U}_G & \mathbf{U}_C\end{pmatrix}^\top, \tag{4.62}$$

we can obtain

$$\mathbf{K}_G = \mathbf{U}_G\mathbf{\Lambda}_G\Psi_0(\mathbf{\Lambda})\mathbf{U}_G^\top, \tag{4.63}$$

which gives the gradient kernel function $\Psi_G(\mathbf{\Lambda}_G) = \mathbf{\Lambda}_G\Psi_0(\mathbf{\Lambda}_G)$. □

In the following we provide the respective corollaries for other derivative operations of interest, where the proofs can directly follow from the fact that derivatives preserve Gaussianity.

Corollary 4.15 (Curl of a triangle GP). *Suppose a triangle function $\mathbf{f}_2$ is a GP $\mathbf{f}_2 \sim \mathrm{GP}(\mathbf{0}, \mathbf{K}_2)$ with $\mathbf{K}_2 = \Psi_2(\mathbf{L}_2) = \mathbf{U}_2\Psi_2(\mathbf{\Lambda}_2)\mathbf{U}_2^\top$ given the eigendecomposition $\mathbf{L}_2 = \mathbf{U}_2\mathbf{\Lambda}_2 \mathbf{U}_2^\top$. Then, its curl is an edge GP $\mathbf{f}_C \sim \mathrm{GP}(\mathbf{0}, \mathbf{K}_C)$ where $\mathbf{K}_C = \mathbf{U}_C\Psi_C(\mathbf{\Lambda}_C)\mathbf{U}_C^\top$ with*

$$\Psi_C(\mathbf{\Lambda}_C) = \mathbf{\Lambda}_C\Psi_2(\mathbf{\Lambda}_C). \tag{4.64}$$

Proposition 4.16 (Div and Curl of edge GPs). *Let* $\mathbf{f}_1 \sim \mathrm{GP}(\mathbf{0}, \mathbf{K})$ *be a Hodge-compositional edge Gaussian process in Definition 4.3. Then, its divergence and curl are Gaussian processes on nodes and triangles, respectively, as follows*

$$\mathbf{B}_1\mathbf{f} \sim \mathrm{GP}(\mathbf{0}, \mathbf{B}_1\mathbf{K}_G\mathbf{B}_1^\top), \quad \mathbf{B}_2^\top\mathbf{f} \sim \mathrm{GP}(\mathbf{0}, \mathbf{B}_2^\top\mathbf{K}_C\mathbf{B}_2). \tag{4.65}$$

Remark 4.17. These interactions between GPs on nodes, edges and triangles provide us alternative ways to construct gradient and curl edge GPs [cf. Corollaries 4.7 and 4.15], as well as construct appropriate node GPs and triangle GPs. They are more applicable when the underlying physical relationships exist between the corresponding functions and the GP priors on the original simplices are easier to construct.

4

4.A.10 Alternative Hodge-compositional Edge GPs

Here we provide the proof for Proposition 4.8 giving an alternative way to build HC edge GPs.

Proof. From the Hodge decomposition, we can write an edge function as

$$\mathbf{f}_1 = \mathbf{f}_H + \mathbf{B}_1^\top\mathbf{f}_0 + \mathbf{B}_2\mathbf{f}_2. \tag{4.66}$$

where $\mathbf{f}_0$ and $\mathbf{f}_2$ are some node and triangle functions. Then, the proof can be completed by using the results from Corollaries 4.7 and 4.15. □

4.A.11 Alternative HC Edge GPs from SPDEs on Edges

While gradient and curl edge GPs in Definition 4.3 can be linked to their SPDEs as discussed by Proposition 4.2, we can also obtain the alternatively constructed counterparts in Corollaries 4.7 and 4.15 from SPDEs. Again, we consider the Matérn family.

Corollary 4.18. *Suppose a node function* $\mathbf{f}_0$ *is a graph (node) Matérn GP* $\mathbf{f}_0 \sim \mathrm{GP}(\mathbf{0}, \mathbf{K}_0)$ *with*

$$\mathbf{K}_0 = \Psi_0(\mathbf{L}_0) = \Big(\frac{2\nu_0}{\kappa_0^2}\mathbf{I} + \mathbf{L}_0\Big)^{-\nu_0}. \tag{4.67}$$

Then, Corollary 4.7 gives us its gradient as a gradient edge GP $\mathbf{f}_G \sim \mathrm{GP}(\mathbf{0}, \mathbf{K}_G)$ *with*

$$\mathbf{K}_G = \mathbf{L}_\mathrm{d}\Big(\frac{2\nu_0}{\kappa_0^2}\mathbf{I} + \mathbf{L}_\mathrm{d}\Big)^{-\nu_0}. \tag{4.68}$$

Furthermore, the gradient GP $\mathbf{f}_G$ *is the solution of the following SPDE*

$$\Big(\frac{2\nu_0}{\kappa_0^2}\mathbf{I} + \mathbf{L}_\mathrm{d}\Big)^{\frac{\nu_0}{2}}\mathbf{f}_G = \mathbf{B}_1^\top\mathbf{w}_0 \tag{4.69}$$

where $\mathbf{w}_0$ *is a standard Gaussian noise on nodes following* $\mathbf{f}_0 \sim \mathcal{N}(\mathbf{0}, \mathbf{I})$.

Proof. First, we can solve the SPDE with the following solution

$$\mathbf{f}_G = \left(\frac{2\nu_0}{\kappa_0^2}\mathbf{I} + \mathbf{L}_\mathrm{d}\right)^{-\frac{\nu_0}{2}} \mathbf{B}_1^\top \mathbf{w}_0 = \mathbf{B}_1^\top \left(\frac{2\nu_0}{\kappa_0^2}\mathbf{I} + \mathbf{L}_0\right)^{-\frac{\nu_0}{2}} \mathbf{w}_0 \tag{4.70}$$

where the second equality follows from the definition of $\mathbf{L}_0$ and $\mathbf{L}_\mathrm{d}$. Given that $\mathbf{w}_0$ is a GP, so is $\mathbf{f}_G$ and we can study its covariance as

$$\begin{aligned}
\mathrm{Cov}[\mathbf{f}_G] &= \mathbf{B}_1^\top \left(\frac{2\nu_0}{\kappa_0^2}\mathbf{I} + \mathbf{L}_0\right)^{-\frac{\nu_0}{2}} \mathrm{Cov}[\mathbf{w}_0] \left(\frac{2\nu_0}{\kappa_0^2}\mathbf{I} + \mathbf{L}_0\right)^{-\frac{\nu_0}{2}} \mathbf{B}_1 \\
&= \mathbf{B}_1^\top \left(\frac{2\nu_0}{\kappa_0^2}\mathbf{I} + \mathbf{L}_0\right)^{-\nu_0} \mathbf{B}_1 \\
&= \mathbf{L}_\mathrm{d} \left(\frac{2\nu_0}{\kappa_0^2}\mathbf{I} + \mathbf{L}_\mathrm{d}\right)^{-\nu_0}
\end{aligned} \tag{4.71}$$

which completes the proof. □

For completeness, we give the corollary relating the curl Matérn edge GP obtained from some triangle GP to its SPDE representation.

Corollary 4.19. *Suppose a triangle function $\mathbf{f}_2$ is a triangle Matérn GP $\mathbf{f}_2 \sim \mathrm{GP}(\mathbf{0}, \mathbf{K}_2)$ with*

$$\mathbf{K}_2 = \Psi_2(\mathbf{L}_2) = \left(\frac{2\nu_2}{\kappa_2^2}\mathbf{I} + \mathbf{L}_2\right)^{-\nu_2}. \tag{4.72}$$

Then, Corollary 4.15 gives us its curl adjoint as a curl edge GP $\mathbf{f}_C \sim \mathrm{GP}(\mathbf{0}, \mathbf{K}_C)$ with

$$\mathbf{K}_C = \mathbf{L}_\mathrm{u} \left(\frac{2\nu_2}{\kappa_2^2}\mathbf{I} + \mathbf{L}_\mathrm{u}\right)^{-\nu_2}. \tag{4.73}$$

Furthermore, the curl GP $\mathbf{f}_C$ is the solution of the following SPDE

$$\left(\frac{2\nu_2}{\kappa_2^2}\mathbf{I} + \mathbf{L}_\mathrm{u}\right)^{\frac{\nu_2}{2}} \mathbf{f}_C = \mathbf{B}_2 \mathbf{w}_2 \tag{4.74}$$

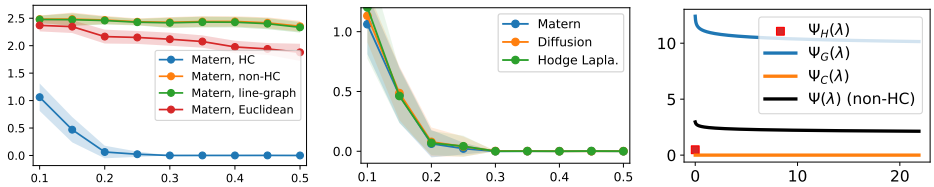
where $\mathbf{w}_2$ is a standard Gaussian noise on triangles following $\mathbf{f}_2 \sim \mathcal{N}(\mathbf{0}, \mathbf{I})$.

Proof. The proof can follow the same procedure as above for Corollary 4.18. □

4.B Experiments

Here we provide additional details on the three experiments presented in the main text.

Experimental Setup In our three experiments we consider the regression tasks and implement GP regression using the `GPyTorch` library [Gardner et al., 2018]. We optimize the marginal log likelihood loss for 1000 iterations with the ADAM optimizer where the learning rate is set to the default value of 0.001. We run each experiment 10 times with hyperparameters randomly initialized. We report evaluation metrics averaged over 10 experiments and the respective standard deviations. All experiments are run on a NVIDIA GeForce RTX 3080 GPU with 10GB of memory.



(a) RMSEs versus training ratios (b) RMSEs versus training ratios (c) Learned Matérn kernels

Figure 4.7: (a) Forex prediction RMSEs of different GPs using Matérn kernels with respect to training ratios. (b) Forex prediction RMSEs of HC GPs using different edge kernels with respect to training ratios. (c) Learned HC and non-HC Matérn kernels in the spectrum for a training ratio of 0.2.

4

Line-graph Construction Given the incidence matrix $\mathbf{B}_1$ of the original graph, the adjacency matrix and the corresponding graph Laplacian of the line-graph can be found as $\mathbf{A}_{lg} := |\mathbf{B}_1^\top \mathbf{B}_1 - 2\mathbf{I}|$ and $\mathbf{L}_{lg} = \mathrm{diag}(\mathbf{A}_{lg}\mathbf{1}) - \mathbf{A}_{lg}$.

4.B.1 Additional Details for the Forex Experiment

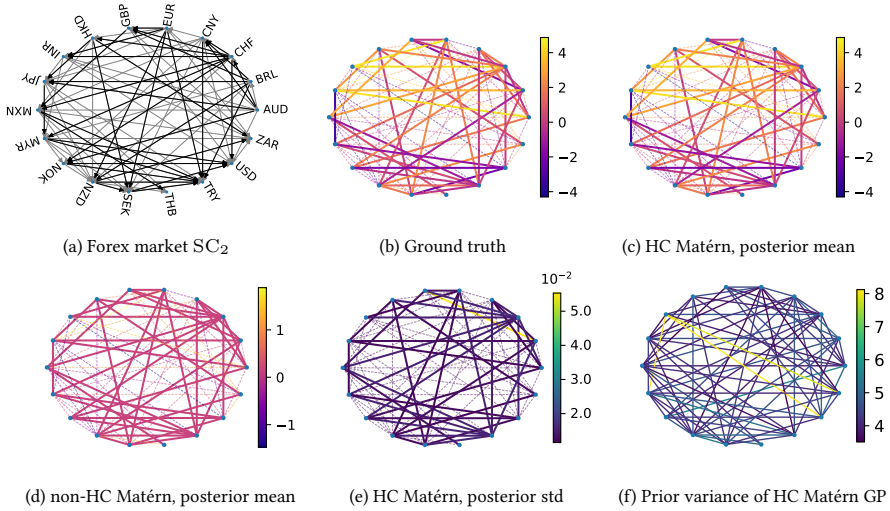
In the forex experiment, we obtain the data from *Foreign Exchange Data* by *Oanda Corporation*[3]. The data was collected at 2018/20/05 17:00 UTC by Jia et al. [2019]. It includes the pairwise exchange rates between the 25 most traded currencies, which form 210 exchangeable pairs. With them as nodes and edges, we then construct an unweighted SC_2 by including the triangles formed by any three pairwise exchangeable currencies. For an edge $\{i, j\}$ connecting currencies i, j, we assign its orientation based on an alphabetical order of their currency names, and likewise for a triangle. For each exchangeable pair, we consider the underlying edge flow as $f_1(i, j) = \log r^{i/j}$, translating the arbitrage-free condition to curl-free condition, where $r^{i/j}$ is the midpoint between ask and bid prices. Fig. 4.7 shows the prediction RMSEs using different GP models with respect to training ratios from 0.1 to 0.5 with a step 0.05, as well as the learned Matérn kernels.

For visualizing the predictions using different models, we consider a smaller market for better visibility where we first randomly removed seven currencies then half of the exchangeable pairs, resulting 18 currencies and 77 pairs, as shown in Fig. 4.8.

4.B.2 Additional Details for Ocean Current Analysis

In the second experiment, we consider the ocean drifter data, also known as *Global Lagrangian Drifter Data*, which was collected by NOAA Atlantic Oceanographic and Meteorological Laboratory[4]. Each point in the dataset is a buoy at a specific time, with buoy ID, location (in latitude and longitude), date/time, velocity and water temperature. We consider the buoys that were in the North Pacific ocean dated from 2010 to 2019 with a size of around three million. The dataset itself is a 3D point cloud after converting the location to the *earth-centered, earth-fixed* (ECEF) coordinate system. We follow the procedure in

[3] https://www.oanda.com/.
[4] http://www.aoml.noaa.gov/envids/gld/.



(a) Forex market SC_2 (b) Ground truth (c) HC Matérn, posterior mean

(d) non-HC Matérn, posterior mean (e) HC Matérn, posterior std (f) Prior variance of HC Matérn GP

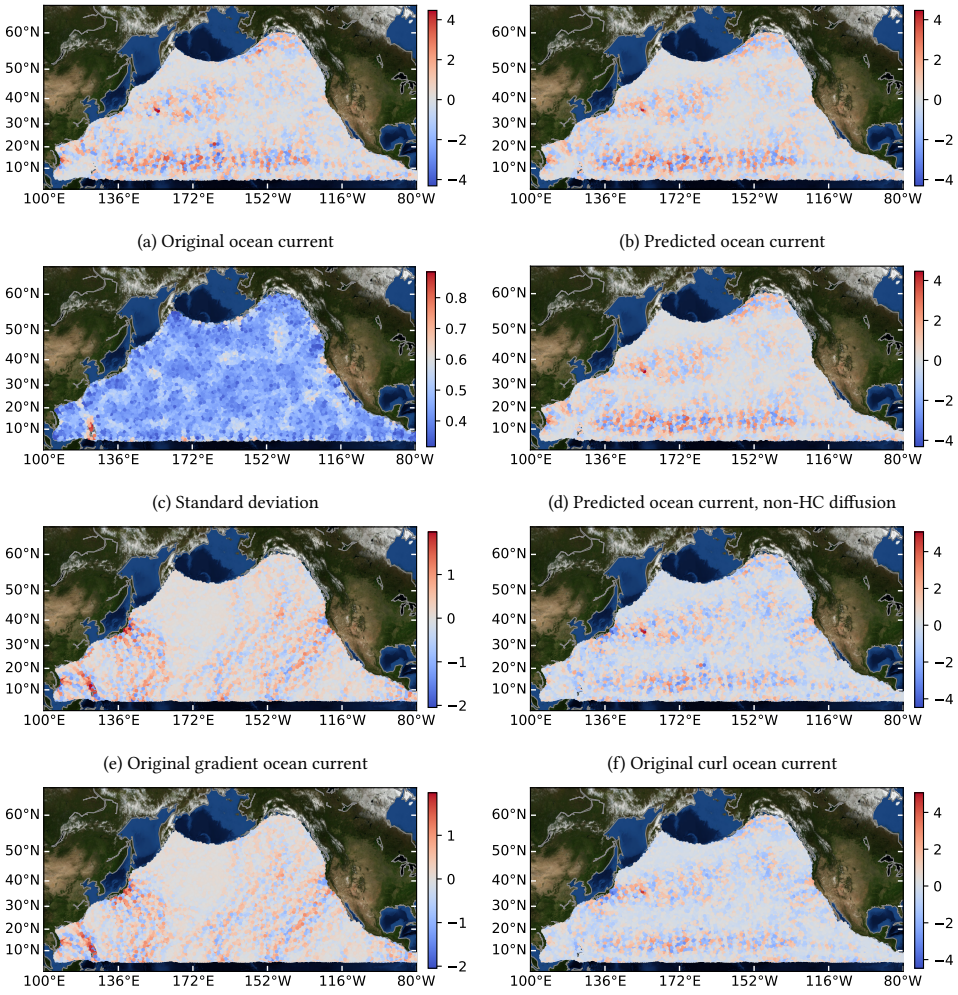
Figure 4.8: (a-e): Visualization of forex rates predictions in a smaller market. (f): Prior variance of the learned HC Matérn GP. Note that (b-d) and (f) are the same as the ones in the main content [cf. Fig. 4.4]. Here we show them with a better resolution.

Chen & Meila [2021] to first sample 1,500 buoys furthest from each other, then construct a weighted SC_2 as a Vietoris-Rips (VR) complex with N_1 around 20k and N_2 around 90k. We then convert the velocity field into flows on the edges of SC_2 by using the linear integration approximation [Chen et al., 2021c]. We randomly sample 20% of the edges for training and test on the rest. To efficiently construct the edge kernels, we use eigensolver in Knyazev [2001], implemented using the `megaman` library [McQueen et al., 2016], to compute the eigenpairs associated to the 500 largest eigenvalues. We evaluate the prediction mean and uncertainty in the edge flow domain, reported in Table 4.4. Furthermore, we obtain the gradient and curl components of the edge flow of the prediction as in Appendix 4.A.5. We visualize the predictions in the edge flow domain in Fig. 4.9. We see that both HC and non-HC edge diffusion GPs give close performance and they capture the general pattern of the edge flow. Moreover, the standard deviation is small in most of the locations except few locations (small islands around the lower left corner) where the edge flows (velocity fields) exhibit more discontinuities due to the boundary.

We further convert the edge flows back into vector fields, as shown in Fig. 4.10. We refer to Chen et al. [2021c] for this procedure. We also approximate the standard deviation of the velocity field prediction by sampling 50 edge flows from the posterior distribution and converting them to the vector field domain, followed by computing the average ℓ_2 distance between the samples and the mean per location, as shown in Fig. 4.10d.

4.B.3 Additional Details for Water Supply Networks

We obtain the Zhi Jiang WSN from Dandy [2016] which contains 114 nodes (113 tanks and 1 source reservoir) and 164 edges (water pipes), no triangles considered. We build



(a) Original ocean current

(b) Predicted ocean current

(c) Standard deviation

(d) Predicted ocean current, non-HC diffusion

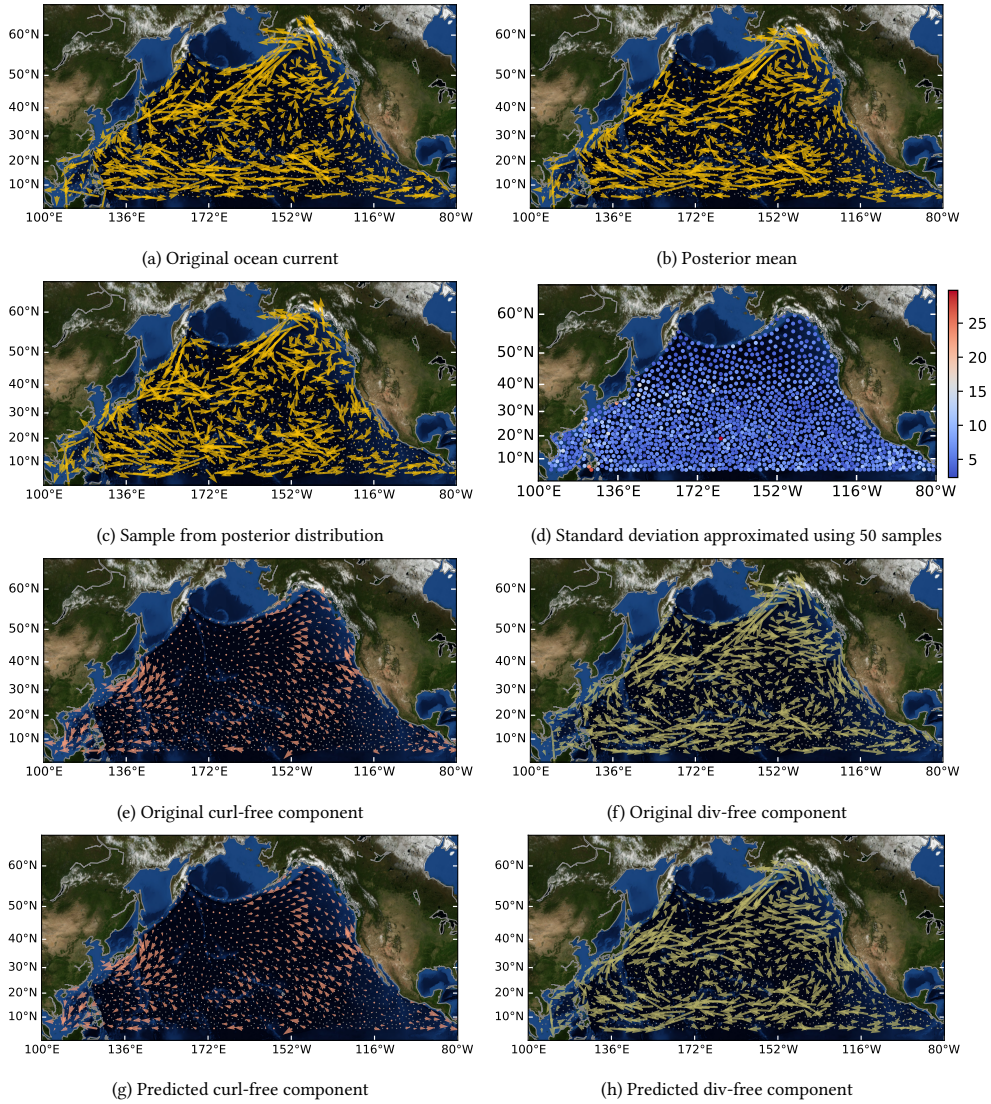
(e) Original gradient ocean current

(f) Original curl ocean current

(g) Predicted gradient flow

(h) Predicted curl flow

Figure 4.9: (a-h) Results for ocean current prediction with 20% training ratio in the edge flow domain. Note that we highlight the edge flow values on the middle points of the edges.



(a) Original ocean current

(b) Posterior mean

(c) Sample from posterior distribution

(d) Standard deviation approximated using 50 samples

(e) Original curl-free component

(f) Original div-free component

(g) Predicted curl-free component

(h) Predicted div-free component

Figure 4.10: (a-h) Results for ocean current prediction with 20% training ratio in the vector field domain. Note that (a-b) and (g-h) are the same as in Fig. 4.5. We show them here for reader's convenience.

Table 4.4: Ocean current inference results.

Method	RMSE			NLPD		
	Diffusion	Matérn	Hodge Laplacian	Diffusion	Matérn	Hodge Laplacian
Euclidean	1.00 ± 0.01	1.00 ± 0.00	–	1.42 ± 0.01	1.42 ± 0.10	–
Line-Graph	0.99 ± 0.00	0.99 ± 0.00	–	1.41 ± 0.00	1.41 ± 0.00	–
Non-HC	0.35 ± 0.00	0.35 ± 0.00	0.35 ± 0.00	0.33 ± 0.00	0.36 ± 0.03	0.33 ± 0.01
HC	0.34 ± 0.00	0.35 ± 0.00	0.35 ± 0.00	0.33 ± 0.01	0.37 ± 0.04	0.33 ± 0.01

an unweighted graph based on the topology of this WSN. We model the hydraulic heads as functions on nodes $\mathbf{f}_0$ and water flowrates as functions on edges $\mathbf{f}_1$. A WSN is often governed by the following equations

$$\begin{aligned} \text{mass conservation} &: \mathbf{B}_1 \mathbf{f}_1 = \mathbf{q}, \\ \text{Hazen-Williams equation} &: [\mathbf{B}_1^\top \mathbf{f}_0](e) = \bar{\mathbf{f}}_1(e) := r_e f_1(e)^{1.852} \end{aligned} \tag{4.75}$$

for a pipe e, where $\mathbf{q} \in \mathbb{R}^{N_0}$ is the demand on nodes, r_e is the roughness of pipe e [Dini & Tabesh, 2014]. We then use the `WNTR` library [Klise et al., 2017] to simulate a scenario generating the states of node heads and edge flowrates given the pipe roughnesses and the node demands. The latter are sampled uniformly from 0 to 10 (unit $\mathrm{liter/s}$), modeling the read-world demand.

We consider the joint state estimation of both heads (using node GPs) and the adjusted flowrates $\bar{\mathbf{f}}_1$ (using edge GPs). Specifically, our GP models are

$$\begin{pmatrix} \mathbf{f}_0 \\ \bar{\mathbf{f}}_1 \end{pmatrix} \sim \mathrm{GP}\left(\begin{pmatrix} \mathbf{0} \\ \mathbf{0} \end{pmatrix}, \begin{pmatrix} \mathbf{K}_0 & \\ & \mathbf{K}_1 \end{pmatrix} \right). \tag{4.76}$$

We choose the Matérn and diffusion node GPs [Borovitskiy et al., 2021]. For HC edge GPs, we leverage the physical prior to model $\mathbf{K}_1 = \mathbf{B}_1^\top \mathbf{K}_0 \mathbf{B}_1$ as discussed in Corollary 4.7, while for non-HC edge GPs, we choose them as in (4.5), of the same type as node GPs. We randomly sample 50% of the nodes and edges for training and use the rest for test. Note that the WSN has small edge connectivity. The randomness of the training set may disconnect the graph, which may deteriorate the performance, causing the large variance in the metrics.

5

Topological Schrödinger Bridge Matching

Gaussian processes on simplicial complexes in Chapter 4 provide us a modest statistical modeling of simplicial signals. In the probabilistic setting, a more involved question concerns the matching of two arbitrary distributions, which is fundamental in the modern machine learning tasks, especially in generative modeling. In this chapter, we investigate the problem of matching distributions of simplicial signals and intend to build models for generative learning and matching on simplicial complexes. The classical formulation of Schrödinger bridge (SB) problem originates from Schrödinger's Gedankenexperiment where he aimed to seek the "most likely process" that describes the random evolution between two marginals, i.e., two point coulds of diffusive particles, relative to a Wiener reference process. This was later on shown to be closely related to optimal transport theory. Here, instead of Wiener process, we formulate the topological SBP to match two distributions of simplicial signals with respect to a topology-aware reference driven by the simplicial convolution in Chapter 2, e.g., stochastic diffusion on graphs/simplicial complexes. Inspired by the recent results on SBPs, we try to find the solutions of topological SBP in both Gaussian and general cases. While a closed form solution for the general case is intractable, we leverage the power of neural networks (e.g., simplicial CNNs in Chapter 3) to approximate the involved unknowns. Upon the recent likelihood training framework that unifies the diffusion and flow matching methods, we build the topological SB models for generative modeling and matching on simplicial complexes. This chapter is based on the work published in Yang [2025].

5.1 Introduction

As a fundamental problem in statistics and optimization, *matching distributions* aims to find a map that transforms one distribution to another. It has found numerous applications in machine learning tasks, particularly in generative modeling, which often involves learning a transformation from a data distribution to a simple one (often Gaussian) for efficient

sampling and inference. While various methods have been proposed, including score-based [Ho et al., 2020; Song et al., 2020b] and flow-based [Lipman et al., 2022] generative models, among others [Albergo et al., 2024; Neklyudov et al., 2023; Tong et al., 2024a], the *Schrödinger Bridge* (SB)-based methods [Chen et al., 2022b; De Bortoli et al., 2021; Liu et al., 2024] provide a principled framework for matching *arbitrary* distributions.

Inspired by Schrödinger [1931; 1932], the classical SB problem (SBP) aims to find an optimal *stochastic process* that evolves from an initial distribution to a final distribution, while minimizing the *relative entropy* (Kullback-Leibler divergence) between the measures of the optimal process and the Wiener process [Léonard, 2014]. Alternatively, the SBP can be cast as a *stochastic optimal control* (SOC) problem which minimizes the kinetic energy while matching the distributions through a nonlinear stochastic process [Dai Pra, 1991; Pavon & Wakolbinger, 1991]. The optimal solution to this problem satisfies a *Schrödinger system* of coupled *forward-backward* (FB) stochastic differential equations (SDEs) [Léonard, 2014]. Traditionally, SB problems have been solved by addressing the unknowns in this system using purely numerical methods [Föllmer, 1988; Fortet, 1940]. More recently, machine learning approaches have been proposed [Chen et al., 2022b; De Bortoli et al., 2021; Pavon et al., 2021; Vargas et al., 2021; Wang et al., 2021] where the unknowns are approximated by learnable models (e.g., Gaussian processes, neural networks) trained on data-driven objectives, such as the *likelihood* training [Chen et al., 2022b].

However, SB-based methods have primarily focused on solving tasks in Euclidean spaces, such as time series, images [Deng et al., 2024] and point clouds. Modern learning tasks often involve data supported on irregular topological domains such as graphs, simplicial and cell complexes. Arising from applications like chemical reaction networks, biological networks, power systems, social networks [Bick et al., 2023; Faskowitz et al., 2022; Wang et al., 2022], the emerging field of *topological machine learning* [Papamarkou et al., 2024] centers on signals supported on topological objects such as nodes and edges, which can represent sensor data or flow type data over network entities. A direct application of existing SB models to such topological data may fail due to their inability to account for the underlying topology. Thus, in this work, we investigate the SBP for topological signals, with a focus on node and edge signals over networks modeled as graphs and simplicial complexes. To this end, we propose the *Topological Schrödinger Bridge problem* ($\mathcal{T}$SBP) between two distributions of signals defined on a topological domain, with a focus on the distributions of node signals and edge flows on graphs and simplicial complexes. To match distributions of such topological signals, our contributions are threefold.

(i) We propose the *Topological Schrödinger Bridge problem* ($\mathcal{T}$SBP), which seeks an optimal *topological stochastic process* that minimizes the relative entropy with respect to a reference process, while respecting the initial and final distributions. To incorporate the domain knowledge, we define the reference process to follow *topology-aware* SDEs ($\mathcal{T}$SDEs) with a linear *topological convolution* drift term, admitting tractable Gaussian transition kernels. This subsumes the commonly-used stochastic heat diffusions on graphs and simplicial complexes for networked dynamics modeling.

(ii) Focusing on the case where the end distributions are Gaussian, we find the closed-form optimal Gaussian $\mathcal{T}$SB and characterize it in terms of a *stochastic interpolant* time marginal, as well as its Itô differential. This generalizes the results of Bunne et al. [2023] where the

reference process is limited to SDEs *scalar*-valued linear coefficients. For the general case, we show that, upon existing results, the optimal $\mathcal{T}$SB adheres a pair of FB-$\mathcal{T}$SDEs governed by some unknown terms (also called *policies*), which in turn satisfy a *system* driven by topological dynamics.

(iii) We propose the $\mathcal{T}$SB-based model for topological signal generative modeling and matching. Specifically, we parameterize the *hard-to-solve* policies by some (topology-aware) learnable models (e.g., graph/simplicial neural networks), and train them by maximizing the likelihood of the model based on Chen et al. [2022b]. We show that $\mathcal{T}$SB-based models unify the extensions of score-based and diffusion bridge-based models [Song et al., 2020b; Zhou et al., 2024] for topological signals.

We validate the theoretical results and demonstrate the practical implications of $\mathcal{T}$SB-models on synthetic and real-world networks involved with brain signals, single-cell data, ocean currents, seismic events and traffic flows. Before concluding the chapter, we provide an extensive discussion on future directions in generative modeling for topological data. Overall, our work lies in the intersection of SB theory, stochastic dynamics on topology, machine learning and generative modeling for topological signals.

Notations. For $x \in \mathbb{R}^n$, the gradient, divergence and Hession of a function $f(x)$ are denoted by $\nabla f(x) \in \mathbb{R}^n$, $\nabla \cdot f(x) \in \mathbb{R}$ and $\Delta f(x) \in \mathbb{R}^{n\times n}$, respectively. We denote by X a *stochastic process* $(X_t)_{0\le t\le 1}$ as a map $X : [0,1] \times \mathcal{X} \to \mathbb{R}^n$ from the unit time interval $[0,1]$ (i.e., *index space*) and sample space $\mathcal{X}$ (e.g., Euclidean space) to $\mathbb{R}^n$ (state space). Here X_t is a random variable representing the state at t. The standard n-dim *Wiener process* (Brownian motion) is denoted by W. Let $\Omega = \mathcal{C}([0,1],\mathbb{R}^n)$ denote the space of all continuous $\mathbb{R}^n$-valued paths on $[0,1]$, and let $\mathcal{P}(\Omega)$ denote the space of probability measures on Ω. For a *path measure* $\mathbb{P} \in \mathcal{P}(\Omega)$ describing the law of the process X, we denote by $\mathbb{P}_t$ its *time marginal* that describes the distribution of X_t, i.e., if $X \sim \mathbb{P}$, then $X_t \sim \mathbb{P}_t$.

In this chapter, we make the following assumptions on the stochastic processes and measures. We assume distributions of random variables are associated with measures that have a *density* with respect to the Lebesgue measure. We may then use the terms *measure* and *density* interchangeably, unless otherwise specified.

5.2 Background

5.2.1 Schrödinger Bridge Problem

Let $\mathbb{Q}_W$ be the path measure of a Wiener process $\mathrm{d}Y_t = \sigma\,\mathrm{d}W_t$ with variance σ^2. The *classical* SBP [Léonard, 2014] seeks an optimal path measure $\mathbb{P}$ on Ω by minimizing its relative entropy D_{KL} with respect to $\mathbb{Q}_W$

$$\min D_{\mathsf{KL}}(\mathbb{P}\|\mathbb{Q}_W), \quad \text{s.t. } \mathbb{P} \in \mathcal{P}(\Omega), \mathbb{P}_0 = \rho_0, \mathbb{P}_1 = \rho_1, \tag{SBP}$$

where ρ_0 and ρ_1 are the *prescribed* initial and final time marginals on $\mathbb{R}^n$. Intuitively, the SBP aims to find a stochastic process evolving from ρ_0 to ρ_1 that are "most likely" to a *reference* (a *prior*) process, here the Wiener process. This is in fact a *dynamic* formulation

of the *entropic-regularized optimal transport* (OT) with a quadratic transport cost [Villani, 2009]. The *static* formulation reads

$$\min_{\pi \in \Pi(\rho_0, \rho_1)} \int_{\mathbb{R}^n \times \mathbb{R}^n} \frac{1}{2} \|x_0 - x_1\|^2 \, \mathrm{d}\pi(x_0, x_1) + \sigma^2 D_{\mathrm{KL}}(\pi \| \rho_0 \otimes \rho_1) \tag{E-OT}$$

where $\Pi(\rho_0, \rho_1)$ is the set of couplings (or transport plans) between ρ_0 and ρ_1, and $\rho_0 \otimes \rho_1$ denotes their product measure (i.e., independent coupling). The first term is the quadratic cost of transporting mass from ρ_0 to ρ_1, and the second term is the relative entropy of the coupling π with respect to the independent coupling $\rho_0 \otimes \rho_1$, weighted by the variance σ^2 of the Wiener process.

In this *static* formulation, the optimization is over the coupling of ρ_0 and ρ_1, as opposed to the full path measure in the *dynamic* formulation (SBP). As $\sigma \to 0$, E-OT reduces to the typical 2-Wasserstein OT, and the associated dynamic problem is given by Benamou & Brenier [2000]. The entropy regularization makes the OT strongly convex, ensuring unique solutions, and allows for efficient algorithms like Sinkhorn algorithm [Villani, 2009].

5

5.2.2 Topological Signals

In this work, we are interested in signals defined on a graph, a simplicial complex or a cell complex such a topological domain, denoted by $\mathcal{T}$. If $\mathcal{T}$ is a graph with a node set and an edge set, we may define a *node signal* $x \in \mathbb{R}^n$ as a collection of values associated to the nodes, where n denotes the number of nodes. Such signals often arise from sensor measurements in sensor networks, user properties in social networks, etc [Shuman et al., 2013]. Similarly, if $\mathcal{T}$ is a simplicial 2-complex SC_2 with the sets of nodes, edges, as well as triangular faces (or triangles), we can define an *edge flow* by associating a real value to each *oriented* edge. Here, for an edge $e = \{i, j\}$, if we choose $[i, j]$ as its positive orientation, then $[j, i]$ represents the opposite [Godsil & Royle, 2001]. The sign of the signal thus indicates the flow orientation relative to the chosen one. Such edge signals often represent flows of information or energy, such as water flows, power flows, or transaction flows [Bick et al., 2023]. Moreover, we may consider signals on general topological objects such as higher-order simplices (or cells). If n is the number of simplices, we refer to $x \in \mathbb{R}^n$ as a *topological signal* where the i-th entry represents the signal value on the i-th simplex. In topology, these are called *cochains*, which are the discrete analogues to *differential forms* [Lim, 2020].

The emerging field of learning on graphs and topology [Barbarossa & Sardellitti, 2020; Papamarkou et al., 2024] concerns such topological signals, where the central idea is to leverage the underlying topological structure in $\mathcal{T}$. For example, the *graph Laplacian* or adjacency matrix (or their variants) can be used to encode the graph's structure, acting as a spectral operator for node signals [Chung, 1997]. Similarly, in a SC_2, the *Hodge Laplacian* can be defined as the operator for edge flows, composed of the *down* and *up* parts, which encode the edge adjacency through a common node or triangle, respectively. Other variants of Hodge Laplacians can also be defined [Grady & Polimeni, 2010; Schaub et al., 2020]. Thus, for a topological signal $x \in \mathbb{R}^n$, we assume a *Laplacian-type*, positive semidefinite, *topological operator* $L \in \mathbb{R}^{n \times n}$ on $\mathcal{T}$ which encodes the topological structure.

In a probabilistic setting, a *topological signal* can be considered random, following some high-dim distribution on $\mathbb{R}^n$ associated with the topology $\mathcal{T}$. This allows for the application of probabilistic methods to topological signals, similar to Euclidean cases. Recent works [Alain et al., 2024; Borovitskiy et al., 2021; Yang et al., 2024] have modeled node signals and edge flows using *Gaussian processes* (GPs) on graphs and simplicial complexes. These GPs encode the topological structure by building their covariance matrix (kernel) Σ as a *matrix function* of the associated Laplacian L. For example, a diffusion node GP uses the kernel $\Sigma = \exp(-\frac{\kappa^2}{2}L)$ with a hyperparameter κ and the graph Laplacian L. Other GPs can be defined as well to model signals in specific subspaces with certain properties [Yang et al., 2024] or to jointly model the node-edge signals [Alain et al., 2024].

5.3 Topological Schrödinger Bridge Problem

In a topological domain $\mathcal{T}$, we consider a *topological stochastic process* X where the index space is instead the product space of $[0,1]$ and the set of topological objects (e.g., nodes, edges) in $\mathcal{T}$, and the state space $\mathbb{R}^n$ is the space of topological signals with n the cardinality of the set. When we consider the node set of a graph (the edge set of a SC_2), X is a stochastic process of node signals (edge flows). We assume that X follows some *unknown* dynamics with its law described by the path measure $\mathbb{P}$. For some prescribed initial and final time-marginals, i.e., $X_0 \sim \mathbb{P}_0 = \nu_0$ and $X_1 \sim \mathbb{P}_1 = \nu_1$, we then aim to obtain the optimal $\mathbb{P}$ by solving the *Topological Schrödinger Bridge Problem* ($\mathcal{T}$SBP):

$$\min D_{\mathsf{KL}}(\mathbb{P} \,\|\, \mathbb{Q}_{\mathcal{T}}), \qquad \text{s.t. } \mathbb{P} \in \mathcal{P}(\Omega), \mathbb{P}_0 = \nu_0, \mathbb{P}_1 = \nu_1. \tag{$\mathcal{T}$SBP}$$

Here, $\mathbb{Q}_{\mathcal{T}}$ is the path measure of a reference process Y which follows some prior *topology-aware* stochastic dynamics on $\mathcal{T}$. Effectively, the solution $\mathbb{P}$ to $\mathcal{T}$SBP describes the "most likely" process X that conforms to the prior Y in the sense of minimizing relative entropy with respect to $\mathbb{Q}_{\mathcal{T}}$, while respecting the initial and final distributions ν_0 and ν_1.

Topological stochastic dynamics. For the reference process Y, given an initial topological signal condition $Y_0 = y_0$, we assume it follows a general class of topological SDEs:

$$\mathrm{d}Y_t = f(t, Y_t; L)\,\mathrm{d}t + g_t\,\mathrm{d}W_t, \tag{$\mathcal{T}$SDE}$$

where $f_t \equiv f(t,\cdot;L) : \mathbb{R}^n \to \mathbb{R}^n$ is a time-varying *drift* that depends on the topological structure $\mathcal{T}$ through the operator L, and $g_t \equiv g(t) \in \mathbb{R}$ is a scalar *diffusion* coefficient. We assume that g_t is uniformly lower-bounded, i.e., $g_t^2 \geq c_1$ for some $c_1 > 0$, and that f_t is Lipschitz continuous in the sense that there exists a constant $c_2 > 0$ such that $\|f(t,y;L) - f(t,y';L)\| \leq c_2\|y - y'\|$ for all $y, y' \in \mathbb{R}^n$ and $t \in [0,1]$.

For tractability, we consider a class of *linear dynamics* on $\mathcal{T}$ with the following drift term:

$$f(t, Y_t; L) = H_t(L)Y_t + \alpha_t, \quad \text{with } H_t(L) = \textstyle\sum_{k=0}^{K} h_k(t) L^k \tag{5.1}$$

and $\alpha_t \in \mathbb{R}^n$ a bias term. Here, $H_t(L)$, denoted simply as H_t, is a *matrix polynomial* of L with time-varying coefficients $h_{k,t} \equiv h_k(t)$, which is able to approximate any *analytic function* of L for an appropriate K by the *Cayley-Hamilton theorem*. The drift f_t is also

referred to as a *topological convolution* of the topological signal in the literature. With a graph Laplacian L, this returns the *graph convolution* of a node signal [Sandryhaila & Moura, 2013; 2014], and with the Hodge Laplacian for edges or general simplices, it yields the *simplicial convolution* [Yang et al., 2022b]. Various topological machine learning methods have been developed based on such convolutions for their expressivity and efficiency. We provide a few examples of linear $\mathcal{T}$SDE, which will be used later.

Topological stochastic heat diffusion: $\mathcal{T}$SDE gives the stochastic variant of heat equation on $\mathcal{T}$

$$\mathrm{d}Y_t = -cLY_t\,\mathrm{d}t + g_t\,\mathrm{d}W_t, \tag{$\mathcal{T}$SHeat}$$

by setting $H_t = -cL$, with $c > 0$. When $\mathcal{T}$ is a graph with the graph Laplacian L, this dynamics enables modeling graph-time GPs [Nikitin et al., 2022], networked dynamic system [Delvenne et al., 2015; Pereira et al., 2010; Santos et al., 2024], and social opinion dynamics [Gaitonde et al., 2021]. More importantly, in the deterministic case of $g_t = 0$, it has a harmonic steady-state, revealing the *topological features* of $\mathcal{T}$. Specifically, node diffusion converges to a state that can identify the connected components (0-dim *holes*), while edge diffusion in a SC_2 based on the Hodge Laplacian has a converging state of circulating around cycles (1-dim *holes*). We refer to Fig. 4.3 for such illustrations [cf. diffusion over graphs and SC_2 in Chapter 4]. In line with our goal of distribution matching for topological signals, we present three examples of $\mathcal{T}$SHeat, inspired by diffusion models for generative modeling [Song et al., 2020b].

Example 5.1 ($\mathcal{T}$SHeat$_{\mathrm{BM}}$). Consider a constant $g_t = g$ in $\mathcal{T}$SHeat. This results in a mixture of a topological heat diffusion and BM with variance g^2, which we refer to as $\mathcal{T}$SHeat$_{\mathrm{BM}}$.

Example 5.2 ($\mathcal{T}$SHeat$_{\mathrm{VE}}$). For some noise scales $0 < \sigma_{\min} < \sigma_{\max}$, consider a time-increasing $g_t = \sqrt{\mathrm{d}\sigma^2(t)/\mathrm{d}t}$ with $\sigma(t) = \sigma_{\min}(\sigma_{\max}/\sigma_{\min})^t$, which drives the well-known *variance exploding* (VE) noising process [Song & Ermon, 2020; Song et al., 2020b]. The resulting form of $\mathcal{T}$SHeat is

$$\mathrm{d}Y_t = -cLY_t\,\mathrm{d}t + \sqrt{\mathrm{d}\sigma^2(t)/\mathrm{d}t}\,\mathrm{d}W_t. \tag{$\mathcal{T}$SHeat$_{\mathrm{VE}}$}$$

Example 5.3 ($\mathcal{T}$SHeat$_{\mathrm{VP}}$). When combined with another noising process, known as the *variance preserving* (VP) process [Ho et al., 2020; Sohl-Dickstein et al., 2015; Song et al., 2020b], we obtain

$$\mathrm{d}Y_t = -\big(\tfrac{1}{2}\beta(t)I + cL\big)Y_t\,\mathrm{d}t + \sqrt{\beta(t)}\,\mathrm{d}W_t, \tag{$\mathcal{T}$SHeat$_{\mathrm{VP}}$}$$

where $\beta(t) = \beta_{\min} + t(\beta_{\max} - \beta_{\min})$ with scales $0 < \beta_{\min} < \beta_{\max}$. The drift here can be considered as an instantiation of the topological convolution $H_t = -\big(\frac{1}{2}\beta(t)I + cL\big)$ in (5.1).

Gaussian transition kernels. The $\mathcal{T}$SDE, as an Itô process, is fully characterized by its *transition kernel* in a probabilistic sense. As a result of the linear drift (5.1) of $\mathcal{T}$SDE, the associated transition kernel $p_{t|s}(y_t|y_s)$ (i.e., conditional distribution of $Y_t|Y_s$) is Gaussian. Its mean and covariance can be computed according to Särkkä & Solin [2019, Eq. 6.7]. Let the *transition matrix* of the ODE $\mathrm{d}Y_t = H_t(L)Y_t\,\mathrm{d}t$ be denoted by $\Psi_{ts} \equiv \Psi(t,s)$, which is given by $\Psi_{ts} = \exp(\int_s^t H_\tau\,\mathrm{d}\tau)$ [cf. Lemma 5.18]. For brevity, we denote Ψ_{t0} as simply Ψ_t. We then have the following lemma.

Lemma 5.4 (Statistics of transition kernels). *For the $\mathcal{T}$SDE with the linear drift (5.1), its Gaussian transition kernel $p_{t|0}(y_t|y_0)$ has the mean m_t and the cross covariance $K_{t_1t_2}$, at t_1 and t_2:*

$$m_t = \Psi_t y_0 + \Psi_t \int_0^t \Psi_\tau^{-1} \alpha_\tau \,\mathrm{d}\tau =: \Psi_t y_0 + \xi_t, \qquad \text{(cond. mean)}$$

$$K_{t_1t_2} = \Psi_{t_1} \left(\int_0^{\min\{t_1,t_2\}} g_\tau^2 \Psi_\tau^{-2} \,\mathrm{d}\tau \right) \Psi_{t_2}^\top. \qquad \text{(cond. cross cov)}$$

More importantly, we may characterize them for $\mathcal{T}$SHeat$_{\text{BM}}$ and $\mathcal{T}$SHeat$_{\text{VE}}$ in closed-forms. Both have the same mean $m_t = \Psi_t y_0$ with $\Psi_t = \exp(-cLt)$, and their covariances are given by:

$$K_{t_1t_2} = \begin{cases} \frac{g^2}{2c} \big[\exp(-cL|t_1 - t_2|) - \exp(-cL(t_1 + t_2)) \big] L^{-1}, & \text{for } \mathcal{T}\text{SHeat}_{\text{BM}} \\ \sigma_{\min}^2 \ln\left(\frac{\sigma_{\max}}{\sigma_{\min}}\right) \exp(-cL(t_1+t_2)) \big[\exp(2A\min\{t_1,t_2\}) - I \big] A^{-1}, & \text{for } \mathcal{T}\text{SHeat}_{\text{VE}} \end{cases} \tag{5.2}$$

with $A = \ln\left(\frac{\sigma_{\max}}{\sigma_{\min}}\right) I + cL$. If L is singular, we use a perturbed $L + \epsilon I$ for a small $\epsilon > 0$ to compute $K_{t_1t_2}$ or numerically model the $\mathcal{T}$SHeat. We detail the derivations in Appendix 5.B. These expressions allow for tractable solutions for the $\mathcal{T}$SBP and, more importantly, facilitate the construction of $\mathcal{T}$SB-based learning models, as will be discussed later.

5.3.1 Towards an Optimal Solution of $\mathcal{T}$SBP

To solve the classical SBP, early mathematical treatments [Beurling, 1960; Föllmer, 1988; Fortet, 1940; Jamison, 1975] lead to a *Schrödinger system* characterizing the SB optimality. Similarly, by Disintegration of Measures, we can convert the $\mathcal{T}$SBP to a static problem over the joint measure $\mathbb{P}_{01}$ of the initial and final states, instead of the full path measure $\mathbb{P}$

$$\min D_{\mathsf{KL}}(\mathbb{P}_{01} \,\|\, \mathbb{Q}_{\mathcal{T}01}), \quad \text{s.t. } \mathbb{P}_{01} \in \mathcal{P}(\mathbb{R}^n \times \mathbb{R}^n), \mathbb{P}_0 = \nu_0, \mathbb{P}_1 = \nu_1 \qquad (\mathcal{T}\text{SBP}_{\text{static}})$$

where $\mathbb{Q}_{\mathcal{T}01}$ is the joint measure of the reference process Y at $t = 0$ and 1. The $\mathcal{T}$SBP$_{\text{static}}$ only concerns at the boundary times, unlike the (*dynamic*) $\mathcal{T}$SBP. Using Lagrange multipliers for the linear constraints above, we can arrive a *Schrödinger System* that is instead *driven by topological dynamics* (see Appendix 5.C), differing from the classical case [Jamison, 1975; Léonard, 2014]. This can be also interpreted through the equivalent E-OT formulation of $\mathcal{T}$SBP$_{\text{static}}$:

$$\min_{\mathbb{P}_{01}} \int_{\mathbb{R}^n \times \mathbb{R}^n} \frac{1}{2} \|y_1 - \Psi_1 y_0 - \xi_1\|^2_{K_{11}^{-1}} \,\mathrm{d}\mathbb{P}_{01}(y_0, y_1) + \int_{\mathbb{R}^n \times \mathbb{R}^n} \log(\mathbb{P}_{01}) \,\mathrm{d}\mathbb{P}_{01} \qquad (\mathcal{T}\text{E-OT})$$

where the transport cost is linked to the $\mathcal{T}$SDE as a K_{11}^{-1}-weighted norm of the difference $y_1 - m_1$.

On the other hand, this system could *also* be derived from the SOC view which makes more apparent connections to machine learning methods. Initiated by the variational formulation of the classical SBP by Dai Pra [1991]; Pavon & Wakolbinger [1991], Caluya &

Halder [2021] extended the analysis to the case with a general nonlinear reference process and derived the optimality condition. Since it is convenient to arrive an SOC formulation for the $\mathcal{T}$SBP (see Appendix 5.C), we readily obtain the following optimality.

Proposition 5.5 ($\mathcal{T}$SBP optimality; Caluya & Halder [2021]; Chen et al. [2022b]). *The optimal solution $\mathbb{P}$ of $\mathcal{T}$SBP can be expressed as the path measure of the following forward (5.3a), or equivalently, backward (5.3b), $\mathcal{T}$SDE:*

$$\mathrm{d}X_t = [f_t + g_t Z_t]\,\mathrm{d}t + g_t\,\mathrm{d}W_t, \quad X_0 \sim \nu_0, Z_t \equiv g_t \nabla \log \varphi_t(X_t) \tag{5.3a}$$

$$\mathrm{d}X_t = [f_t - g_t \hat{Z}_t]\,\mathrm{d}t + g_t\,\mathrm{d}W_t, \quad X_1 \sim \nu_1, \hat{Z}_t \equiv g_t \nabla \log \hat{\varphi}_t(X_t) \tag{5.3b}$$

where (5.3a) runs forward and (5.3b) runs backward with a backward Wiener process. Here, $\varphi_t \equiv \varphi_t(X_t)$ and $\hat{\varphi}_t \equiv \hat{\varphi}_t(X_t)$ satisfy a pair of PDEs system (forward-backward Kolmogorov equations). Using nonlinear Feynman-Kac formula (or applying Itô's formula on $\log \varphi_t$ and $\log \hat{\varphi}_t$), this PDE system admits the SDEs

$$\begin{aligned} \mathrm{d}\log \varphi_t &= \frac{1}{2}\|Z_t\|^2\,\mathrm{d}t + Z_t^\top\,\mathrm{d}W_t, \\ \mathrm{d}\log \hat{\varphi}_t &= \left(\frac{1}{2}\|\hat{Z}_t\|^2 + \nabla\cdot(g_t\hat{Z}_t - f_t) + \hat{Z}_t^\top Z_t\right)\mathrm{d}t + \hat{Z}_t^\top\,\mathrm{d}W_t. \end{aligned} \tag{5.4}$$

Then, the optimal path measure has the time-marginal $\mathbb{P}_t = \varphi_t(X_t)\hat{\varphi}_t(X_t) = \mathbb{P}_t^{(5.3a)} = \mathbb{P}_t^{(5.3b)}$. That is, the paths given by (5.3a) and (5.3b) are equivalent in the sense that they have the same time-marginal distributions.

This optimality condition adapts the result from Chen et al. [2022b] for $\mathcal{T}$SBP. From the forward-backward $\mathcal{T}$SDEs (FB-$\mathcal{T}$SDEs in (5.3)), we see that the optimal Z_t guides the forward $\mathcal{T}$SDE to the final ν_1, and likewise $\hat{Z}_t$ adjusts the reverse $\mathcal{T}$SDE to return to the initial ν_0. While solving the system (5.4) is still highly nontrivial, we **highlight** that the FB-$\mathcal{T}$SDEs (5.3) and (5.4) pave a way for constructing generative models and efficient training algorithms, as demonstrated by the recent works, to name a few, [Chen et al., 2022b; De Bortoli et al., 2021; Pavon et al., 2021; Vargas et al., 2021]. We further discuss in detail how to build such models for topological signals in Section 5.5.

5.4 Gaussian Topological SBP

In this section, we consider the special case of $\mathcal{T}$SBP where the initial and final measures are Gaussians, to which we refer as the *Gaussian topological SBP* (G$\mathcal{T}$SBP). We show that there exists a closed-form G$\mathcal{T}$SB by following the idea in Bunne et al. [2023], which focuses on a limited class of reference SDEs with a scalar coefficient in the drift, instead of a convolution operator $H_t(L)$. We establish the first closed-form expression on the G$\mathcal{T}$SB in the following theorem.

Theorem 5.6. *Denote by $\mathbb{P}$ the solution to G$\mathcal{T}$SBP with $\nu_0 = \mathcal{N}(\mu_0, \Sigma_0)$ and $\nu_1 = \mathcal{N}(\mu_1, \Sigma_1)$. Then, $\mathbb{P}$ is the path measure of a* Markov Gaussian process *whose marginal $X_t \sim \mathcal{N}(\mu_t, \Sigma_t)$ admits an expression in terms of the initial and final variables, X_0, X_1, as follows*

$$X_t = \bar{R}_t X_0 + R_t X_1 + \xi_t - R_t \xi_1 + \Gamma_t Z \tag{5.5}$$

where $Z \sim \mathcal{N}(0, I)$ is standard Gaussian, independent of (X_0, X_1), and we have

$$\begin{aligned} R_t &= K_{t1} K_{11}^{-1}, \quad \bar{R}_t = \Psi_t - R_t \Psi_1, \\ \Gamma_t &:= \mathrm{Cov}[Y_t | (Y_0, Y_1)] = K_{tt} - K_{t1} K_{11}^{-1} K_{1t}. \end{aligned} \tag{5.6}$$

Proof. We provide a sketch of the proof here, with the full derivations presented in Appendix 5.D.

1. By Disintegration of Measures, we first solve the reduced *static* Gaussian $\mathcal{T}$SBP$_{\text{static}}$ (i.e., $\mathcal{T}$E-OT). We can then convert the problem into a Gaussian E-OT via a change-of-variables. The closed-form formula for the latter has been recently found by Janati et al. [2020, Theorem 1]. Via an inverse transform, we can then obtain the optimal coupling $\mathbb{P}_{01}$ [i.e., the optimal G$\mathcal{T}$SB$_{\text{static}}$].

2. In the disintegration of G$\mathcal{T}$SBP to its static problem, the optimum is achieved when $\mathbb{P}$ shares the *bridge* with the reference $\mathbb{Q}_{\mathcal{T}}$ (i.e., $\mathbb{P}$ is in the *reciprocal class* of $\mathbb{Q}_{\mathcal{T}}$) [Föllmer, 1988; Léonard, 2014, Proposition 1]. The $\mathbb{Q}_{\mathcal{T}}$*-bridge*, $\mathbb{Q}_{\mathcal{T}}^{xy} = \mathbb{Q}_{\mathcal{T}}[\cdot | Y_0 = x, Y_1 = y]$, can be constructed using the conditional Gaussian formula and Lemma 5.4. Upon this, together with the optimal $\mathbb{P}_{01}$, we can construct the optimal X_t and the marginal $\mathbb{P}_t$. □

At the first sight, the construction of optimal process X in (5.5) meets the recently proposed *stochastic interpolant* framework by Albergo et al. [2024, Definition 1], in that $X_{t=0} = X_0$ and $X_{t=1} = X_1$, and $\Gamma_0 = \Gamma_1 = 0$. Moreover, from (5.5), we can compute the marginal statistics $\mathbb{P}_t$ in terms of its mean μ_t and covariance Σ_t in closed-form as well, detailed in Corollary 5.26. In the following, we characterize the process X under the optimal $\mathbb{P}$ in terms of its Itô differential.

Theorem 5.7 (SDE representation)**.** *Under the optimal $\mathbb{P}$, the process X admits the SDE dynamics:*

$$\begin{aligned} \mathrm{d}X_t &= f_{\mathcal{T}}(t, X_t; L)\,\mathrm{d}t + g_t\,\mathrm{d}W_t, \text{ where} \\ f_{\mathcal{T}}(t, x; L) &= S_t^\top \Sigma_t^{-1}(x - \mu_t) + \dot{\mu}_t \end{aligned} \tag{5.7}$$

with μ_t, Σ_t the mean and covariance of X_t [cf. Corollary 5.26] and we have

$$S_t = P_t - Q_t^\top + H_t K_{tt} - K_{t1} K_{11}^{-1} \Upsilon_t^\top, \tag{5.8}$$

with $P_t = (R_t \Sigma_1 + \bar{R}_t C)\dot{R}_t^\top, Q_t = -\dot{\bar{R}}_t (C R_t^\top + \Sigma_0 \bar{R}_t^\top), \Upsilon_t = H_t K_{t1} + g_t^2 \Psi_t^{-1} \Psi_1^\top$, where C is the covariance of X_0, X_1 in the optimal $\mathbb{P}_{01}$, and $\dot{R}_t, \dot{\bar{R}}_t$ are the time-derivatives of $R_t, \bar{R}_t$.

Proof. We detail the proof in Appendix 5.D and outline a sketch here. From Caluya & Halder [2021]; Léonard [2014] [cf. Theorem 5.24], the optimal $\mathbb{P}$ is the law of an SDE in the class of (5.7). To determine the drift, we first compute the associated *infinitesimal generator* by definition for some test function. Since the generator for an Itô SDE is *known* (dependent on the drift) [Särkkä & Solin, 2019, Eq. 5.9], we can then match the two expressions and find a closed-form for the drift term. □

Theorems 5.6 and 5.7 characterize the optimal $\mathbb{P}$ of the G$\mathcal{T}$SBP from different views. While the stochastic interpolant formula is intuitive and straightforward, it is natural to look for

the associated SDE for a Markov measure. Despite the packed variables, both results [cf. Eqs. (5.5) and (5.7)] fundamentally depend on the transition kernel, defined by the transition matrix Ψ_t and $\xi_t, K_{t_1 t_2}$ [cf. Lemma 5.4], which has closed-forms (5.2) for $\mathcal{T}$SHeat$_{\text{BM}}$ and $\mathcal{T}$SHeat$_{\text{VE}}$, and is related to the topological convolution H_t. From a broader perspective, Theorems 5.6 and 5.7 extended the existing results of Bunne et al. [2023], where the reference process has a limited drift $cY_t + \alpha_t$ for some *scalar* c. While Chen et al. [2016] aimed to solve for a linear drift with a matrix coefficient, their results lead to the solution of a matrix Riccati equation, which is computationally expensive.

Solution complexity. While the variables involved in Eqs. (5.5) and (5.7) involve many matrix operations, we remark that (i) the *underlying* Ψ_t is a matrix function of L and can be computed efficiently [Higham, 2008]. Given the eigen-decomposition $L = U\Lambda U^\top$, which scales at $\mathcal{O}(n^3)$, denote by $\tilde{h}_k^{t,s} = \int_s^t h_{k,\tau}\,\mathrm{d}\tau$ the integral of the scalar coefficients in H_t [cf. (5.1)], we then have $\Psi_{ts} = U \exp\big(\sum_{k=0}^{K} \tilde{h}_k^{t,s} \Lambda^k\big) U^\top$, where the matrix exponential can be directly computed elementwise on each diagonal element of Λ. We may also consider the Chebyshev polynomial approximation based on Huguet et al. [2023], which is feasible thanks to the closed-form Chebyshev coefficients for matrix exponentials [Marcotte et al., 2022, Eq. 5], despite the need for all $t \in [0,1]$. (ii) The other terms depending on Ψ_t can be computed similarly in the eigenspectrum of L.

5.5 Topological Signal Generative Models

The recent SB-based generative modeling framework primarily relies on the *learnable* parameterizations of the $(Z_t, \hat{Z}_t)$ pair (also viewed as the FB policies) in the FB-SDEs and a trainable objective that approximates the SBP. Specifically, Vargas et al. [2021] and De Bortoli et al. [2021] use GPs and neural networks, respectively, to parameterize the policies, and alternatively train them using iterative proportional fitting (IPF) to solve the *half*-bridge problem. On the other hand, Chen et al. [2022b] derived a *likelihood* based on the SB optimality condition, generalizing the *score matching* framework [Song et al., 2020b]. Upon the proposed $\mathcal{T}$SBP, along with the above theoretical results, we now discuss how to build generative models for topological signals using the existing framework designed for Euclidean domains.

$\mathcal{T}$SB-based model. Consider the matching task: *In some topological domain $\mathcal{T}$, given two sets of signal samples following initial and final distributions ν_0, ν_1 on $\mathcal{T}$, we aim to learn a Topological Schrödinger Bridge between the two distributions.* From Proposition 5.5, the optimal $\mathcal{T}$SB follows the FB-$\mathcal{T}$SDEs in (5.3). Moreover, given a path sampled from the forward SDE (5.3a) with an initial signal x_0, one can obtain an unbiased estimation of the *log-likelihood* $\mathcal{L}(x_0)$ of the $\mathcal{T}$SB model driven by the optimal policies by using (5.4) [Chen et al., 2022b, Theorem 4]. Similarly, the log-likelihood $\mathcal{L}(x_1)$, given a final sample x_1, can be found. This allows us to build a $\mathcal{T}$SB-based model for topological signals, following the ideas of Chen et al. [2022b]; De Bortoli et al. [2021].

We first parameterize the policies, Z_t and $\hat{Z}_t$, by two learnable models $Z_t^\theta \equiv Z(t, x; \theta)$ and $\hat{Z}_t^{\hat{\theta}} \equiv \hat{Z}(t, x; \hat{\theta})$ with parameters θ and $\hat{\theta}$, resulting in the parameterized FB-$\mathcal{T}$SDEs. Then, we can perform a *likelihood training* by minimizing the following loss functions in

an alternative fashion at initial and final signal samples x_0 and x_1

$$l(x_0;\hat{\theta}) = \int_0^1 \mathbb{E}_{X_t\sim(5.3a)}\left[\frac{1}{2}\|\hat{Z}_t^{\hat{\theta}}\|^2 + g_t\nabla\cdot\hat{Z}_t^{\hat{\theta}} + Z_t^{\theta^\top}\hat{Z}_t^{\hat{\theta}}\middle| X_0 = x_0\right]\mathrm{d}t, \tag{5.9a}$$

$$l(x_1;\theta) = \int_0^1 \mathbb{E}_{X_t\sim(5.3b)}\left[\frac{1}{2}\|Z_t^{\theta}\|^2 + g_t\nabla\cdot Z_t^{\theta} + \hat{Z}_t^{\hat{\theta}^\top} Z_t^{\theta}\middle| X_1 = x_1\right]\mathrm{d}t, \tag{5.9b}$$

which are, respectively, the upper bounds of the negative log-likelihoods (after dropping the unrelated terms) of the signal samples x_0 and x_1 given paths sampled from the FB-$\mathcal{T}$SDEs.

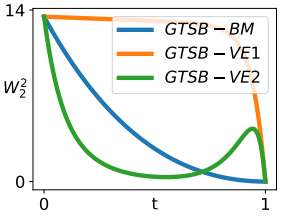
Other choice of reference $\mathcal{T}$SDE. In this work, we mainly consider reference dynamics following $\mathcal{T}$SHeat$_{\text{BM}}$, $\mathcal{T}$SHeat$_{\text{VE}}$ and $\mathcal{T}$SHeat$_{\text{VP}}$. For the dynamics involved with Hodge Laplacians in a SC_2, we may further allow *heterogeneous* diffusion based on the down and up parts of the Laplacian. Bunne et al. [2023] proposed to better initialize the SB model using the closed-form Gaussian SB. Likewise, we can consider the G$\mathcal{T}$SB in (5.7) as a stronger prior process, which yet requires a GP approximation from signal samples. We also consider *fractional Laplacian* for some cases to enable a more efficient exploration of the network [Riascos & Mateos, 2014] due to its non-local nature.

Topological neural networks (TNNs). While Chen et al. [2022b]; De Bortoli et al. [2021] applied convolutional neural networks for the Euclidean SBP, we naturally consider parameterizing the policies using the emergent TNNs. For node signals, we could consider graph convolution networks (GCNs) [Kipf & Welling, 2017]; and likewise, for edge flows in a SC_2, simplicial neural networks (SNNs) [Roddenberry et al., 2021]. These *topology-aware* models perform convolutional learning upon the topological structure, more efficient with less parameters and better in performance.

Complexity. Like standard SB models, $\mathcal{T}$SB-based models also require simulations of the FB-$\mathcal{T}$SDEs. The key difference is that these models operate over topological networks where the drift (5.1) involves a matrix-vector multiplication H_tY_t. However, this is essentially a *recursive* iteration of LY_t, which is efficient due to the *typically sparse* structure of L, reflecting the underlying topological sparsity. Moreover, our TNN-parameterized policies are also efficient for the same reason.

Connection to other models. As discussed in Chen et al. [2022b], in the special case of $Z_t \equiv 0$ and $\hat{Z}_t$ as the *score function* (scaled by g_t), the likelihood of SB models reduces to that of the score-based models [Song et al., 2020b] when ν_1 is a simple Gaussian and the forward process is designed to reach ν_1. Furthermore, if the reference process is poorly designed. SB models can still guide the process to the target distribution through these learnable policies, thus generalizing score-based models. On the other hand, for FB-$\mathcal{T}$SDEs, we can also obtain *probability flow ODEs* [Chen et al., 2018; Song et al., 2020b] which share the same time-marginals and likelihoods, allowing for exact likelihood evaluation of the model. Training through the likelihood of these flow ODEs naturally links to flow-based models. While there are no direct score-based or flow-based models for topological signals, the above discussions apply to $\mathcal{T}$SB-based models. We refer to Appendix 5.E for more details where we show how the variants of these models including the *diffusion bridge* models [Zhou et al., 2024] for topological processes can be constructed.

5.6 Experiments



We first validate the theoretical results on G$\mathcal{T}$SB using the synthetic graph in Fig. 4.3. Here, we aim to bridge a zero-mean graph Matérn GP ν_0 with $\Sigma_0 = (I + L)^{-1.5}$ and a diffusion GP ν_1 with $\Sigma_1 = \exp(-20L)$. Using the $\mathcal{T}$SHeat$_{\text{BM}}$ and $\mathcal{T}$SHeat$_{\text{VE}}$ reference dynamics, we obtain the closed-form X_t in (5.5), from which we further compute the covariance Σ_t [cf. Corollary 5.26]. We measure the *Bures-Wasserstein* distance between Σ_t and Σ_1. From the *right-hand-side* figure, we see that both bridges reach the target distribution. The bridges exhibit distinct behaviors depending on the reference dynamics, as demonstrated by the disparate curves for $\mathcal{T}$SHeat$_{\text{VE}}$ with $c = 0.01$ and 10. This highlights the flexibility of $\mathcal{T}$SB-models in exploring a large space of topological bridges.

We then focus on evaluating $\mathcal{T}$SB-based models for topological signal *generation* (ν_1 is a Gaussian noise) and *matching* (general ν_1) in different applications, with the goal of investigating the question: ***whether $\mathcal{T}$SB-based models are beneficial for these tasks compared to the standard SB-based models?*** For this goal, we consider as the baseline SB-based models in Euclidean domains which use BM, VE and VP as reference dynamics [Chen et al., 2022b], labeled as SB-BM, SB-VE and SB-VP, respectively. We consider $\mathcal{T}$SB-based models using $\mathcal{T}$SHeat$_{\text{BM}}$, $\mathcal{T}$SHeat$_{\text{VE}}$ and $\mathcal{T}$SHeat$_{\text{VP}}$ as references, labeled as TSB-BM, TSB-VE and TSB-VP, respectively. We refer to Appendix 5.F for the experimental details and additional results, as well as complexity analyses in Appendix 5.F.2.

Topological signal matching. We first consider matching two sets of fMRI brain signals from the Human Connectome Project [Van Essen et al., 2013], which represent the liberal (with high energy, as the initial) and aligned (with low energy, as the final) brain activities, respectively. We use the recommended brain graph [Glasser et al., 2016] that connects 360 parcelled brain regions with edge weights denoting the connection strength. From Fig. 5.1, we see that a TSB-VE model learns to reach at a final state with low energy indicating the aligned activity, whereas SB-VE fails.

We then consider the single-cell embryoid body data that describes cell differentiation over 5 timepoints [Moon et al., 2019]. We follow the preprocessing from Tong [2023]; Tong et al. [2024a;b]. We aim to transport the initial observations to the final state. Our method relies on the affinity graph constructed from the *entire set of observations* (∼18k). We define two normalized indicator functions as the boundary distributions, which specify the nodes corresponding to the data observed at the first and last timepoints. Fig. 5.2 shows the two-dim `phate` embeddings of the groundtruth and predicted data points using TSB-BM and SB-BM models. Here, SB-BM gives very noisy predictions, especially for intermediate ones, even when trained on the full dataset (see Table 5.8).

Edge flows have been used to model vector fields upon a discrete Hodge Laplacian estimate of the manifold Helmholtzian [Chen et al., 2021c]. Following the setup there, we consider the edge-based ocean current matching in a SC_2 (∼20k edges). With an edge GP, learned by Yang et al. [2024] from drifter data, as the initial distribution modeling the currents, we

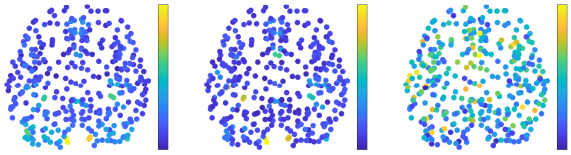


Figure 5.1: Energies of true *(Left)* final state and the predictions obtained from TSB-VE *(Center)* and SB-VE *(Right)* models.

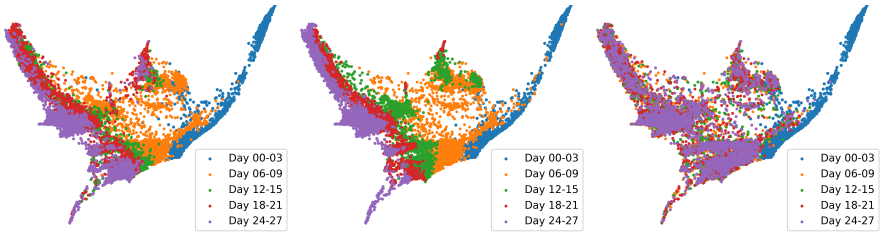


Figure 5.2: `Phate` embeddings of the single-cell data observations *(Left)* and predictions based on TSB-BM *(Center)* and SB-BM *(Right)* models.

synthetize a curl-free edge GP as the final one, modeling different behaviors of currents. From Fig. 5.3, we see that SB-BM fails to reach the final curl-free state, while TSB-BM becomes more divergent, ultimately closer to the target.

For these matching tasks, we evaluate the forward final predictions using 1-Wasserstein distances in Table 5.1, showing the consistent superiority of $\mathcal{T}$SB-based models over SB ones. We reasonably argue that this difference is due to the improper reference in SB-based models, which overlooks the underlying topology. This highlights the role of topology using $\mathcal{T}$SB-based models in these tasks.

Generative modeling. We model the magnitudes of yearly seismic events from IRIS as node signals on a mesh graph of 576 nodes based on the geodesic distance between the vertices of an icosahedral triangulated earth surface [Moresi & Mather, 2019]. We also consider the traffic flow from PeMSD4 dataset modeled as edge flows on a SC_2 with 340 edges [Chen et al., 2022c]. From Table 5.1, we see that $\mathcal{T}$SB-based models consistently outperform SB-based models also for signal generation tasks, highlighting the importance of topology-aware reference processes.

Effect of policy models. From the training curves of TSB-BM/VE for ResBlock and GCN as policy models on the right, we see that the training converges much faster and better using GCN compared to the former. This underlines the positive effect of TNNs on topological signal generative modeling. We refer to Tables 5.3 and 5.4 for the performance metrics of other bridge models with the two policy parameterizations on both seismic and traffic datasets.

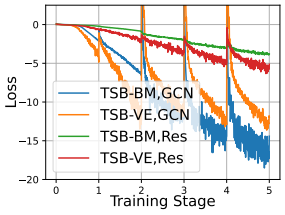


Table 5.1: 1-Wasserstein distances for generating and matching tasks across datasets over five runs, where ⋆ indicates using GSB-VE and GTSB-VE for ocean currents.

Method	Seismic magnitudes	Traffic flows	Brain signals	Single-cell data	Ocean currents
SB-BM	$11.73_{\pm 0.05}$	$18.69_{\pm 0.02}$	$12.08_{\pm 0.08}$	$0.33_{\pm 0.01}$	$7.21_{\pm 0.00}$
SB-VE	$11.49_{\pm 0.04}$	$19.04_{\pm 0.02}$	$17.46_{\pm 0.14}$	$0.33_{\pm 0.01}$	$7.17_{\pm 0.02}$
SB-VP	$12.61_{\pm 0.06}$	$18.22_{\pm 0.03}$	$13.41_{\pm 0.05}$	$0.33_{\pm 0.01}$	$0.83_{\pm 0.01}^{\star}$
TSB-BM	$9.01_{\pm 0.03}$	$10.57_{\pm 0.02}$	$7.51_{\pm 0.08}$	$0.14_{\pm 0.03}$	$6.94_{\pm 0.01}$
TSB-VE	$7.69_{\pm 0.04}$	$10.51_{\pm 0.02}$	$7.59_{\pm 0.05}$	$0.14_{\pm 0.02}$	$6.89_{\pm 0.00}$
TSB-VP	$8.40_{\pm 0.04}$	$9.92_{\pm 0.02}$	$7.67_{\pm 0.11}$	$0.14_{\pm 0.01}$	$0.53_{\pm 0.00}^{\star}$

Effect of G$\mathcal{T}$SB prior. Instead of using $\mathcal{T}$SHeat$_{\text{BM}}$ or $\mathcal{T}$SHeat$_{\text{VE}}$ as the reference, we here consider their corresponding closed-form SDEs (5.7) as the reference, imposing on the bridges a stronger prior carrying the moment information of the data samples [Bunne et al., 2023]. For ocean current matching, we show the samples from the learned FB-$\mathcal{T}$SDEs using GTSB-BM in Fig. 5.3, which arrives at a more faithful final state compared to TSB-BM, as also evaluated in Table 5.1.

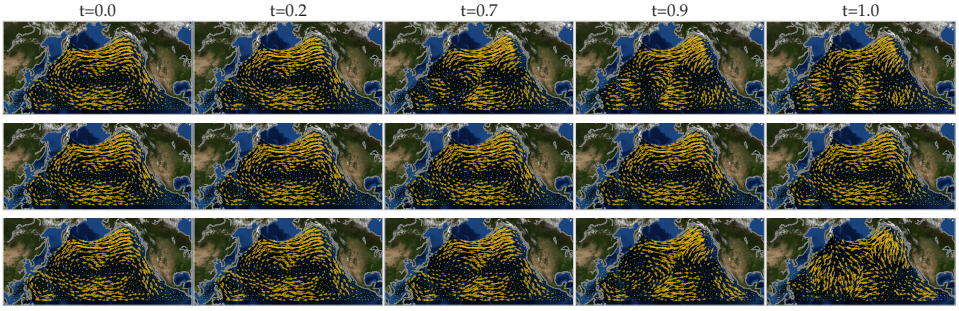


Figure 5.3: Forward sampled currents using TSB-BM *(Top)*, SB-BM *(Center)* and GTSB-BM *(Bottom)*.

5.7 Discussion and Conclusion

In this work, we demonstrated how to construct $\mathcal{T}$SB-based topological signal matching models within the likelihood training framework [Chen et al., 2022b]. We here discuss a few promising future directions based on emerging work and unexplored theoretical results.

On model training. Peluchetti [2023]; Shi et al. [2023] applied iterative Markovian fitting (IMF), as an alternative of IPF, to the classical SBP. This algorithm, trained via score matching, extends to $\mathcal{T}$SB-models with $\mathcal{T}$SHeat$_{\text{BM}}$ or $\mathcal{T}$SHeat$_{\text{VE}}$ as the reference, thanks to their closed-form transition kernels in (5.2). Recent work proposed (partially) *simulation-free* training of SB models. Tong et al. [2024b] learns the optimal SB by flow and score matching the forward SDE upon a heuristic E-OT. Gushchin et al. [2024]; Korotin et al. [2024] modeled Schrödinger potentials using Gaussian mixtures, enabling light training for the optimal drift, and Deng et al. [2024] linearized the forward policy. However,

these methods require the reference dynamics to be either Wiener process or have scalar drifts. While training $\mathcal{T}$SB-based models remain scalable w.r.t. the topology size (see Appendix 5.F.2), extending these approaches to our models is worthwhile but nontrivial.

On model improving. We focused on the reference dynamics driven by a topological convolution H_t up to order one. It is however worthwhile to consider more involved (potentially learnable) convolutions to impose more general priors or incorporate physics knowledge of the process. The scalar diffusion coefficient g_t could be extended as matrix-valued, enabling spatially correlated noising processes over the topology. On the other hand, SB models perform a kinetic energy minimization from the SOC view. [Liu et al., 2024] considered *generalized* SBP by adding a cost term which can model other knowledge of the process. This broadens the applicability of SB models and $\mathcal{T}$SB models could benefit from this, when there are external interactions with the topological process or prior knowledge on the process, such as enforcing curl-free edge flows.

On other models. While we showed the connections of $\mathcal{T}$SB to stochastic interpolants, flow- and score-based models, as well as diffusion bridges, we notice that the $\mathcal{T}$SB optimality can be interpreted as a *Wasserstein gradient flow* (WGF) (see Appendix 5.C.2). For example, the $\mathcal{T}$SHeat$_{\text{BM}}$-driven $\mathcal{T}$SB is the WGF of a functional $\mathcal{F}(\nu)$ of some measure ν with $\mathcal{F}(\nu) = c\int \frac{1}{2}x^\top Lx \cdot \nu(x)\,\mathrm{d}x + \frac{1}{2}g^2 \int \nu \log \nu \,\mathrm{d}x$ where $\mathcal{D}(x) := \frac{1}{2}x^\top Lx$ is the *Dirichlet energy* of x and the second term is the negative entropy. Thus, the $\mathcal{T}$SHeat$_{\text{BM}}$-driven forward $\mathcal{T}$SB essentially reduces the Dirichlet energy. This may not always align with the needs of real-world applications, which in turn motivates developing topological dynamics learning models via the JKO flow [Jordan et al., 1998] of a parametrized functional $\mathcal{D}(x)$ on topology, akin to approaches used in Euclidean domains [Bunne et al., 2022].

We focused on a fixed topological domain, but it is also of interest to study the case where $\mathcal{T}$ itself evolves over time. The $\mathcal{T}$SBP in this scenario may rely on a time-varying operator L_t to guide the reference process. This is relevant for recent generative models for graphs, to name a few [Jo et al., 2022; Liu et al., 2023; Niu et al., 2020], where the graph structure, together with node features, are learned in the latent space based on diffusion models [Song et al., 2020b]. Lastly, we remark that discrete distributions on topological domains may be defined. For instance, nodes of a graph can represent discrete states where node i is associated with a discrete probability P_i. This motivates the emerging geneartive models for discrete data [Austin et al., 2021; Campbell et al., 2024; Haefeli et al., 2023; Ye et al., 2022]. For a formal treatment of matching such discrete distributions on graphs, we refer to Chow et al. [2022]; Léonard [2013]; Maas [2011]; Solomon [2018].

Conclusion. With the goal of matching topological signal distributions beyond Euclidean domains, we introduced the $\mathcal{T}$SBP (topological Schrödinger bridge problem). We defined the reference process using an SDE driven by a topological convolution linear operator, which is tractable and includes the commonly used heat diffusion on topological domains. When the end distributions are Gaussians, we derived a closed-form $\mathcal{T}$SB, generalizing the existing results by Bunne et al. [2023]. In general cases, we showed that the optimal process satisfies a pair of FB-$\mathcal{T}$SDEs governed by the some optimal policies. Building upon these results, we developed $\mathcal{T}$SB-based models where we parameterize the policies as (topological) neural networks and learn them from likelihood training, extending the

framework of Chen et al. [2022b]; De Bortoli et al. [2021] to topological domains. We applied $\mathcal{T}$SB-based models for both topological signal generation and matching in various applications, demonstrating their improved performance compared to standard SB-based models. Overall, our work lies at the intersection of the SB-based distribution matching and topological machine learning, and we hope it inspires further research in this direction.

Appendix

5.A Preliminaries on Schrödinger Bridges

5.A.1 On Optimal Transport

For the static E-OT where the end distributions are Gaussian, when $\sigma = 0$, it reduces to the classical Gaussian optimal transport, whose solution is classical [Villani, 2009]. When $\sigma > 0$, ana analytical solution was found recently by Bojilov & Galichon [2016]; del Barrio & Loubes [2020]; Janati et al. [2020]; Mallasto et al. [2022] as follows.

Theorem 5.8 (Static Gaussian OT; [Janati et al., 2020]). *Let Σ_0, Σ_1 be positive definite. Given two Gaussian measures $\rho_0 \sim \mathcal{N}(\mu_0, \Sigma_0)$ and $\rho_1 \sim \mathcal{N}(\mu_1, \Sigma_1)$, the entropic-regularized optimal transport*

$$\min_{\pi \in \Pi(\mu_0, \mu_1)} \int_{\mathbb{R}^n \times \mathbb{R}^n} \frac{1}{2} \|x_0 - x_1\|^2 \, \mathrm{d}\pi(x_0, x_1) + \sigma^2 D_{KL}(\pi \| \mu_0 \otimes \mu_1) \tag{E-OT}$$

admits a closed-form solution $\pi^\star$

$$\pi^\star \sim \mathcal{N}\left(\begin{bmatrix} \mu_0 \\ \mu_1 \end{bmatrix}, \begin{bmatrix} \Sigma_0 & C_\sigma \\ C_\sigma^\top & \Sigma_1 \end{bmatrix} \right) \tag{5.10}$$

where

$$C_\sigma = \frac{1}{2}(\Sigma_0^{\frac{1}{2}} D_\sigma \Sigma_0^{-\frac{1}{2}} - \sigma^2 I), \quad D_\sigma = (4\Sigma_0^{\frac{1}{2}} \Sigma_1 \Sigma_0^{\frac{1}{2}} + \sigma^4 I)^{\frac{1}{2}}. \tag{5.11}$$

Remark 5.9. Note that while the above results are stated for positive definite covariance matrices (in order for ρ_0 and ρ_1 to have a Lebesgue density), the closed-form solution remains well-defined for positive semi-definite covariance matrices.

5.A.2 On Schrödinger Bridge

Léonard [2014] provides a survey on the connections between SBP and E-OT, showing how SBP can be solved by reducing it to E-OT. Chen et al. [2021b] offers a stochastic control perspective of SBP, which reformulates SBP as a variational optimization problem. Both works provide a comprehensive overview of SBP. Here, we highlight key results necessary for solving $\mathcal{T}$SBP.

Lemma 5.10 (Léonard [2014]). *For a given measure $\mathbb{P}$ over the path space Ω, let $\mathbb{P}^{xy}$ represent the conditioning of $\mathbb{P}$ on paths that take values x and y at $t = 0$ and 1, respectively. That is, $\mathbb{P}^{xy} = \mathbb{P}[\cdot | X_0 = x, X_1 = y]$. Let $\mathbb{P}_{01}$ denote the joint probability for the values of paths at the two ends $t = 0, 1$. Then, $\mathbb{P}$ can be disintegrated into*

$$\mathbb{P}(\cdot) = \int_{\mathbb{R}^n \times \mathbb{R}^n} \mathbb{P}^{xy}(\cdot) \mathbb{P}_{01}(\mathrm{d}x \, \mathrm{d}y). \tag{Disintegration of Measures}$$

Static SBP By Disintegration of Measures, for all $\mathbb{P} \in \mathcal{P}(\Omega)$, the relative entropy can be factorized as

$$D_{\mathrm{KL}}(\mathbb{P}\|\mathbb{Q}) = D_{\mathrm{KL}}(\mathbb{P}_{01}\|\mathbb{Q}_{01}) + \int_{\mathbb{R}^n\times\mathbb{R}^n} D_{\mathrm{KL}}(\mathbb{P}^{xy}\|\mathbb{Q}^{xy})\mathbb{P}_{01}(\mathrm{d}x\,\mathrm{d}y), \tag{5.12}$$

which implies that $D_{\mathrm{KL}}(\mathbb{P}_{01}\|\mathbb{Q}_{01}) \le D_{\mathrm{KL}}(\mathbb{P}\|\mathbb{Q})$ with equality if and only if $\mathbb{P}^{xy} = \mathbb{Q}^{xy}$ for each $(x, y) \in \mathbb{R}^n \times \mathbb{R}^n$. This allows us to reduce the (*dynamic*) SBP to the *static* one.

$$\min D_{\mathrm{KL}}(\mathbb{P}_{01}\|\mathbb{Q}_{01}), \quad \text{s.t. } \mathbb{P} \in \mathcal{P}(\mathbb{R}^n \times \mathbb{R}^n), \mathbb{P}_0 = \rho_0, \mathbb{P}_1 = \rho_1. \tag{SBP$_{\text{static}}$}$$

The above results allow us to reduce the dynamic SBP, which focuses on the entire path of the process, to a static one, which focues instead on the joint distribution of the end points. Furthermore, it readily gives the following important theorem.

Theorem 5.11 (Föllmer [1988]; Léonard [2014]). *The SBP and SBP$_{\text{static}}$ admit, respectively, at most one solution. If SBP has the solution $\mathbb{P}$, then its joint-marginal at the end times $\mathbb{P}_{01}$ is the solution of SBP$_{\text{static}}$. Conversely, if $\mathbb{P}_{01}$ solves SBP$_{\text{static}}$, then the solution of SBP can be expressed as*

$$\mathbb{P}(\,) = \int_{\mathbb{R}^n\times\mathbb{R}^n} \mathbb{Q}^{xy}(\,)\mathbb{P}_{01}(\mathrm{d}x\,\mathrm{d}y), \tag{5.13}$$

which means that $\mathbb{P}$ shares its bridges with $\mathbb{Q}$ (i.e., $\mathbb{P}$ is in the reciprocal class of $\mathbb{Q}$):

$$\mathbb{P}^{xy} = \mathbb{Q}^{xy} \quad \forall (x, y) \in \mathbb{R}^n \times \mathbb{R}^n. \tag{5.14}$$

Proposition 5.12 (Léonard [2014]). *If the reference measure $\mathbb{Q}$ is Markov, then the solution $\mathbb{P}$ of SBP is also Markov.*

These two results show that the optimal solution shares the bridge with the reference measure. That is, the optimal bridge can be constructed by composing the solution $\mathbb{P}_{01}$ of the static problem with the reference measure $\mathbb{Q}^{xy}$.

Schrödinger System

Theorem 5.13 (Chen et al. [2021a]; Jamison [1975]; Léonard [2014]). *Given two probability measures ρ_0, ρ_1 on $\mathbb{R}^n$ and the continuous, everywhere positive Markov kernel $p_{t|s}(y|x)$ (not necessarily associated to a scaled Brownian motion), there exists a unique pair of (up to scaling) of functions $\hat{\varphi}_0, \varphi_1$ on $\mathbb{R}^n$ such that the measure $\mathbb{P}_{01}$ on $\mathbb{R}^n \times \mathbb{R}^n$ defined by*

$$\mathbb{P}_{01} = \int_{\mathbb{R}^n\times\mathbb{R}^n} p_{1|0}(y|x)\hat{\varphi}_0(\mathrm{d}x)\varphi_1(\mathrm{d}y) \tag{5.15}$$

has marginals ρ_0 and ρ_1. Moreover, the Schrödinger bridge from ρ_0 to ρ_1 induces the distribution flow

$$\mathbb{P}_t = \varphi_t\hat{\varphi}_t \text{ with } \varphi_t(x) = \int p_{1|t}(y|x)\varphi_1(\mathrm{d}y), \hat{\varphi}_t(x) = \int p_{t|0}(x|y)\hat{\varphi}_0(\mathrm{d}y). \tag{5.16}$$

This theorem is stated for a general Markov kernel that is not necessarily associated with a Brownian motion. Thus, it can be readily applied to the topological case where the kernel is governed by $\mathcal{T}$SDE.

SOC Formulation Benamou & Brenier [2000]; Dai Pra [1991]; Pavon & Wakolbinger [1991] showed an equivalent SOC formulation of the SBP that aims to minimize the kinetic energy as follows

$$\min_{u\in\mathcal{U}} \mathbb{E}\left[\int_0^1 \frac{1}{2}\|u(t,X_t)\|^2\right], \quad \text{s.t.} \begin{cases} \mathrm{d}X_t = u(t,X_t)\,\mathrm{d}t + \sigma\,\mathrm{d}W_t \\ X_0 \sim \rho_0, X_1 \sim \rho_1, \end{cases} \tag{SBP$_{\text{soc}}$}$$

where $\mathcal{U}$ is the set of finite control functions $u_t \equiv u(t,x)$. This derivation builds upon the fact that the square of the Euclidean distance can be expressed as the infimum of an action integral. We refer to the derivations in Chen et al. [2021b, Section 3.3]. Given the SDE constraint in SBP$_{\text{soc}}$, the associated marginal density $\rho_t \equiv \rho(t,x)$ evolves according to the Fokker-Planck-Kolmogorov equation (FPK, Risken [1996]). This allows to arrive an equivalent variational formulation

$$\min_{(\rho_t,u_t)} \int_{\mathbb{R}^n}\int_0^1 \frac{1}{2}\|u(t,x)\rho(t,x)\|^2\,\mathrm{d}t\,\mathrm{d}x \quad \text{s.t.} \begin{cases} \partial_t\rho_t + \nabla\cdot(\rho_t u_t) = \frac{\sigma^2}{2}\Delta\rho_t, \\ \rho_0 = \rho_0, \rho_1 = \rho_1. \end{cases} \tag{SBP$_{\text{var}}$}$$

Given $\rho_t = \varphi_t\hat{\varphi}_t$, the optimal control in SBP$_{\text{soc}}$ can be obtained by $u_t = \sigma^2\nabla\log\varphi_t$.

5

5.B Stochastic Dynamics on Topological Domains

Compared to the Euclidean domain, the dynamics on topological domains are less studied. Here we provide some existing work on the dynamics on graphs and simplicial complexes. Note that our choice of the linear topological drift f_t in (5.1) is analogous to the ideas in [Archambeau et al., 2007; Verma et al., 2024] which considered linear SDEs to approximate nonlinear dynamics, enabling approximations of more complex topological dynamics.

5.B.1 Preliminaries on (Stochastic) Differential Equations

We are involved with differential equations in this work. In the following, we review some results on ordinary differential equations (ODEs) and SDEs which are required later.

on ODEs

Given an initial solution $x_0 \in \mathbb{R}^n$, consider a linear differential system of the form

$$\mathrm{d}x_t = A_t x_t\,\mathrm{d}t \tag{linear ODE}$$

with A_t a time-varying matrix. To solve this system in closed-form, we require an expression for the state *transition matrix* $\Psi(t,s)$ which transforms the solution at s to t, $x_t = \Psi(t,s)x_s$. For the general case, the closed-form of $\Psi(t,s)$ is not possible. In the following, we introduce an important class of matrices A_t for which a closed-form solution is possible [Antsaklis & Michel, 1997].

Lemma 5.14 (Closed-form of the transition matrix of a linear ODE). *Given a linear ODE, if for every $s,t \geq 0$, we have*

$$A_t\left[\int_s^t A_\tau\,\mathrm{d}\tau\right] = \left[\int_s^t A_\tau\,\mathrm{d}\tau\right]A_t, \tag{5.17}$$

then the transition matrix is given by

$$\Psi(t,s) = \exp\left(\int_s^t A_\tau \,\mathrm{d}\tau\right) := I + \int_s^t A_\tau \,\mathrm{d}\tau + \frac{1}{2!}\left(\int_s^t A_\tau \,\mathrm{d}\tau\right)^2 + \dots. \tag{5.18}$$

For the scalar case, or when A_t is diagonal, or for $A_t = A$, (5.17) is always true.

Lemma 5.15. *For $A_t \in \mathbb{C}[\mathbb{R}, \mathbb{R}^{n\times n}]$, (5.17) is true if and only if $A_t A_s = A_s A_t$ for all s, t.*

Lemma 5.16 (Integration of matrix exponential)**.** *For a nonsingular matrix A, we have*

$$\int_s^t \exp(A\tau)\,\mathrm{d}\tau = \big[\exp(At) - \exp(As)\big]A^{-1}. \tag{5.19}$$

On SDEs

Lemma 5.17 (Itô isometry, Oksendal [2013])**.** *Let $W : [0,1]\times\mathcal{X} \to \mathbb{R}$ denote the canonical real-valued Wiener process defined up to time 1, and let $X : [0,1]\times\mathcal{X} \to \mathbb{R}$ be a stochastic process that is adapted to the filtration generated by W*[1]*. Then*

$$\mathbb{E}\left[\left(\int_s^t X_t \,\mathrm{d}W_t\right)^2\right] = \mathbb{E}\left[\int_s^t X_t^2 \,\mathrm{d}t\right], \tag{5.20}$$

and

$$\mathbb{E}\left[\left(\int_0^t X_t \,\mathrm{d}W_t\right)\left(\int_0^t Y_t \,\mathrm{d}W_t\right)\right] = \mathbb{E}\left[\int_0^t X_t Y_t \,\mathrm{d}t\right]. \tag{5.21}$$

This corollary allows us to compute the covariance of two stochastic processes X_t and Y_t that are adapted to the same filtration.

Transition densities of SDEs All Itô processes, that is, solutions to Itô SDEs, are *Markov processes*. This means that all Itô processes are, in a probabilistic sense, completely characterized by the transition densities (from x_s at time s to x_t at time t, denoted by $p_{t|s}(x_t|x_s) \equiv p(x_s, s; x_t, t)$). The transition density is also a solution to the FPK equation with a degenerate (Dirac delta) initial density concentrated on x_s at time s. We refer to Särkkä & Solin [2019, Thm 5.10].

5.B.2 Transition Densities of $\mathcal{T}$SDE

$\mathcal{T}$SDE First, we can find the transition matrix of the associated ODE to $\mathcal{T}$SDE.

Lemma 5.18. *For an ODE $\mathrm{d}y_t = H_t(L)y_t\,\mathrm{d}t$, the transition matrix is given by*

$$\Psi(t,s) =: \Psi_{ts} = \exp\left(\int_s^t H_\tau \,\mathrm{d}\tau\right) = I + \sum_{k=0}^{\infty} \frac{1}{k!}\left(\int_s^t H_\tau \,\mathrm{d}\tau\right)^k \qquad \text{(transition matrix)}$$

with $y_t = \Psi_{ts} y_s$. Note that Ψ_{ts} is symmetric since H_t is a function of the symmetric L.

[1] An adapted process with respect to a Brownian motion is a stochastic process that only uses information from the past and present, never the future, of the Brownian motion.

This is a direct result from Lemmas 5.14 and 5.15 since $H_t H_\tau = H_\tau H_t$ for all t, τ. By the definition of matrix integral, the computation of Ψ_{ts} is given by

$$\Psi(t,\tau) = \exp(\tilde{H}_{t,\tau}(L)) = \exp\left(\sum_{k=0}^{K} \tilde{h}_k^{t,\tau} L^k\right) \tag{5.22}$$

where $\tilde{h}_k^{t,\tau} = \int_\tau^t h_{k,s}\,\mathrm{d}s$ are the integral of the scalar coefficients in H_t. In the following, we characterize the transition densities of the $\mathcal{T}$SDE, as well as the three concrete examples in Examples 5.1 to 5.3. Then, using the formulas in Särkkä & Solin [2019, Eq. 6.7], we can compute the statistics of the transition kernel.

The following two lemmas compose Lemma 5.4.

Lemma 5.19. *The transition density $p_{t|s}(y_t|y_s)$ of the $\mathcal{T}$SDE conditioned on $Y_s = y_s$ is Gaussian*

$$p_{t|s}(y_t|y_s) \sim \mathcal{N}(y_t; m_{t|s}, K_{t|s}) \tag{5.23}$$

with the mean and covariance, for $t \geq s$, as follows

$$m_{t|s} = \Psi_{ts} y_s + \Psi_t \int_s^t \Psi_\tau^{-1} \alpha_\tau\,\mathrm{d}\tau, \quad K_{t|s} = \Psi_t \left(\int_s^t g_\tau^2 \Psi_\tau^{-2}\,\mathrm{d}\tau\right) \Psi_t^\top.$$

Proof. Given the transition matrix, using the transition kernel formula in Särkkä & Solin [2019, Eq. 6.7], we have

$$m_{t|s} = \Psi_{ts} y_s + \int_s^t \Psi_{t\tau} \alpha_\tau\,\mathrm{d}\tau = \Psi_{ts} y_s + \Psi_t \int_s^t \Psi_\tau^{-1} \alpha_\tau\,\mathrm{d}\tau,$$

where we use the property $\Psi_{t\tau} = \Psi_t \Psi_\tau^{-1}$. Likewise, we have

$$K_{t|s} = \int_s^t g_\tau^2 \Psi_{t\tau} \Psi_{t\tau}^\top\,\mathrm{d}\tau = \Psi_t \left(\int_s^t g_\tau^2 \Psi_\tau^{-2}\,\mathrm{d}\tau\right) \Psi_t^\top.$$

□

Lemma 5.20. *Conditioned on $Y_0 = y_0$, the cross covariance $K(t_1, t_2)$ of $\mathcal{T}$SDE at t_1, t_2 is given by*

$$K_{t_1,t_2} = \Psi_{t_1} \left(\int_0^{\min\{t_1,t_2\}} g_\tau^2 \Psi_\tau^{-2}\,\mathrm{d}\tau\right) \Psi_{t_2}^\top.$$

Proof. This can be obtained by applying the cross covariance function in Särkkä & Solin [2019, Sec 6.4].

$$\begin{aligned}
K_{t_1,t_2} &= \mathrm{Cov}[Y_{t_1}, Y_{t_2}|Y_0] = \mathbb{E}\left[\left(\int_0^{t_1} g_{\tau_1} \Psi_{t_1,\tau_1}\,\mathrm{d}W_{\tau_1}\right)\left(\int_0^{t_2} g_{\tau_2} \Psi_{t_2,\tau_2}\,\mathrm{d}W_{\tau_2}\right)^\top\right] \\
&= \int_0^{\min\{t_1,t_2\}} g_\tau^2 \Psi_{t_1,\tau} \Psi_{t_2,\tau}^\top\,\mathrm{d}\tau && \text{(by Lemma 5.17 (Ito's isometry))} \\
&= \Psi_{t_1} \left(\int_0^{\min\{t_1,t_2\}} g_\tau^2 \Psi_\tau^{-2}\,\mathrm{d}\tau\right) \Psi_{t_2}^\top. && \text{(by } \Psi_{t,\tau} = \Psi_t \Psi_\tau^{-1}\text{)}
\end{aligned}$$

□

$\mathcal{T}$SHeat$_{\mathbf{BM}}$ Given an initial sample y_0 of the random topological signal Y_0, consider the SDE:

$$\mathrm{d}Y_t = -cLY_t\,\mathrm{d}t + g\,\mathrm{d}W_t. \tag{$\mathcal{T}$SHeat$_{\mathrm{BM}}$}$$

Its steady-state (by setting $t \to \infty$) distribution has zero mean and covariance matrix $\Sigma = \frac{g^2}{2c}L^{-1}$. Note that when L is singular, we consider a perturbed version $L + \epsilon I$ with a small constant $\epsilon > 0$. The transition matrix of the associated ODE $\mathrm{d}Y_t = -cLY_t\,\mathrm{d}t$ is given by

$$\Psi_{ts} = \exp\left(-c(t-s)L\right). \tag{5.24}$$

Lemma 5.21. *The transition density $p_{t|s}(y_t|y_s)$ of the $\mathcal{T}$SHeat$_{BM}$ conditioned on $Y_s = y_s$ is Gaussian with the mean and covariance, for $t \geq s$, as follows*

$$m_{t|s} = \Psi_{ts}y_s, \quad K_{t|s} = \frac{g^2}{2c}\left[I - \exp(-2cL(t-s))\right]L^{-1}.$$

Moreover, the conditional dynamics $Y_t|Y_0 = y_0$ has the covariance process at t_1, t_2 as

$$K_{t_1,t_2} = \frac{g^2}{2c}\Big[\exp(-cL|t_2 - t_1|) - \exp(-cL(t_1+t_2))\Big]L^{-1}. \tag{5.25}$$

Proof. For the conditional mean, we can obtain it directly from the transition matrix of the associated ODE. For the conditional covariance, we have

$$\begin{aligned}
K_{t|s} &= \int_s^t \Psi(t,\tau)g^2\Psi(t,\tau)^\top\,\mathrm{d}\tau = \Psi_t\left(\int_s^t \Psi_\tau^{-2}g^2\,\mathrm{d}\tau\right)\Psi_t^\top \\
&= g^2\Psi_t\left(\int_s^t \exp(2cL\tau)\,\mathrm{d}\tau\right)\Psi_t^\top && (\text{by } \Psi_\tau^{-2} = \exp(2cL\tau)) \\
&= \frac{g^2}{2c}\Psi_t\left[\exp(2cLt) - \exp(2cLs)\right]L^{-1}\Psi_t^\top && (\text{by Lemma 5.16}) \\
&= \frac{g^2}{2c}\left[I - \exp(-2cL(t-s))\right]L^{-1}. && (\Psi_t = \exp(-cLt))
\end{aligned}$$

To compute the covariance process of the conditional dynamics $Y_t|Y_0 = y_0$, by definition, we have, for $t_1 \leq t_2$

$$\begin{aligned}
K_{t_1,t_2} &= \mathrm{Cov}[Y_{t_1}, Y_{t_2}|Y_0] = \Psi_{t_1}\left(\int_0^{t_1} g^2\Psi_\tau^{-2}\,\mathrm{d}\tau\right)\Psi_{t_2}^\top && (\text{by Lemma 5.4}) \\
&= g^2\Psi_{t_1}\left[\int_0^{t_1}\exp(2cL\tau)\,\mathrm{d}\tau\right]\Psi_{t_2}^\top && (\text{by } \Psi_\tau^{-2} = \exp(2cL\tau)) \\
&= g^2\Psi_{t_1}\left[\frac{1}{2c}\left[\exp(2cLt_1) - I\right]\right]L^{-1}\Psi_{t_2}^\top && (\text{by Lemma 5.16}) \\
&= \frac{g^2}{2c}\left[\exp(-cL(t_2 - t_1)) - \exp(-cL(t_1+t_2))\right]L^{-1}.
\end{aligned}$$

The case of $t_1 > t_2$ can be similarly derived, which completes the proof. □

$\mathcal{T}$SHeat$_{\text{VE}}$

Lemma 5.22. *The Gaussian transition kernel $p(t|s)$ of $\mathcal{T}$SHeat$_{VE}$ has the mean and covariance*

$$m_{t|s} = \Psi_{ts} y_s, \quad K_{t|s} = \sigma_{\min}^2 \ln\left(\frac{\sigma_{\max}}{\sigma_{\min}}\right) \exp(-2cLt)\left[\exp(2At) - \exp(2As)\right] A^{-1} \tag{5.26}$$

where Ψ_{ts} is the same as (5.24) and $A = \ln\left(\frac{\sigma_{\max}}{\sigma_{\min}}\right) I + cL$. The cross covariance between Y_t and Y_s, conditioned on Y_0, is given by

$$K_{t_1,t_2} = \sigma_{\min}^2 \ln\left(\frac{\sigma_{\max}}{\sigma_{\min}}\right) \exp(-cL(t_1+t_2))[\exp(2A\min\{t_1,t_2\}) - I]A^{-1}. \tag{5.27}$$

Proof. As the associated ODE of $\mathcal{T}$SHeat$_{\text{VE}}$ is also a topological heat diffusion, the transition matrix is the same as $\Psi_{ts} = \exp(-cL(t-s))$ for $\mathcal{T}$SHeat$_{\text{BM}}$. By substituting $\sigma(t)$ into g_t, we can find that

$$g_t = \sigma_{\min}\left(\frac{\sigma_{\max}}{\sigma_{\min}}\right)^t \sqrt{2\ln\left(\frac{\sigma_{\max}}{\sigma_{\min}}\right)}. \tag{5.28}$$

For the covariance of the transition density, we have

$$K_{t|s} = \Psi_t \left(\int_s^t g_\tau^2 \Psi_\tau^{-2}\, \mathrm{d}\tau\right) \Psi_t^\top$$

for which, we need to compute the integral

$$\begin{aligned}
&\int_s^t g_\tau^2 \Psi_\tau^{-2}\, \mathrm{d}\tau = \int_s^t \sigma_{\min}^2 \left(\frac{\sigma_{\max}}{\sigma_{\min}}\right)^{2\tau} 2\ln\left(\frac{\sigma_{\max}}{\sigma_{\min}}\right) \exp(2cL\tau)\, \mathrm{d}\tau \\
&= 2\sigma_{\min}^2 \ln\left(\frac{\sigma_{\max}}{\sigma_{\min}}\right) \left[\int_s^t \left(\frac{\sigma_{\max}}{\sigma_{\min}}\right)^{2\tau} \exp(2cL\tau)\, \mathrm{d}\tau\right] && \text{(factor out the constant)} \\
&= 2\sigma_{\min}^2 \ln\left(\frac{\sigma_{\max}}{\sigma_{\min}}\right) \int_s^t \exp\left[2\tau \ln\left(\frac{\sigma_{\max}}{\sigma_{\min}}\right)\right] \exp(2cL\tau)\, \mathrm{d}\tau \\
& && \text{(by the identity } \exp(\ln x) = x\text{)} \\
&= 2\sigma_{\min}^2 \ln\left(\frac{\sigma_{\max}}{\sigma_{\min}}\right) \int_s^t \exp(2\tau A)\, \mathrm{d}\tau && \text{(by } A = \ln(\sigma_{\max}/\sigma_{\min})I + cL\text{)} \\
&= \sigma_{\min}^2 \ln\left(\frac{\sigma_{\max}}{\sigma_{\min}}\right) (\exp(2At) - \exp(2As))A^{-1}. && \text{(by Lemma 5.16)}
\end{aligned}$$

Thus, we have

$$K_{t|s} = \sigma_{\min}^2 \ln\left(\frac{\sigma_{\max}}{\sigma_{\min}}\right) \exp(-2cLt)\left[\exp(2At) - \exp(2As)\right] A^{-1}.$$

For the cross covariance, assuming $t_1 \le t_2$, then we can find the covariance kernel as

$$K_{t_1,t_2} = \mathrm{Cov}[Y_{t_1}, Y_{t_2}|Y_0] = \Psi_{t_1}\left[\int_0^{t_1} g_\tau^2 \Psi_\tau^{-2}\, \mathrm{d}\tau\right] \Psi_{t_2}^\top \qquad \text{(by Lemma 5.4)}$$

$$= \Psi_{t_1}\left[\int_0^{t_1} \sigma_{\min}^2 \left(\frac{\sigma_{\max}}{\sigma_{\min}}\right)^{2\tau} 2\ln\left(\frac{\sigma_{\max}}{\sigma_{\min}}\right) \exp(2cL\tau)\,\mathrm{d}\tau\right]\Psi_{t_2}^\top$$

(since $\Psi_\tau^{-2} = \exp(2cL\tau)$)

$$= \sigma_{\min}^2 \ln\left(\frac{\sigma_{\max}}{\sigma_{\min}}\right)\Psi_{t_1}\left[\exp(2At_1) - I\right]A^{-1}\Psi_{t_2}^\top \qquad \text{(using the same steps as above)}$$

$$= \sigma_{\min}^2 \ln\left(\frac{\sigma_{\max}}{\sigma_{\min}}\right)\exp(-cL(t_1+t_2))[\exp(2At_1) - I]A^{-1}.$$

The similar steps can be followed for $t_1 > t_2$, which completes the proof. □

$\mathcal{T}$SHeat$_{\text{VP}}$ For this stochastic process, a closed-form transition kernel cannot be found. Yet, we could proceed the following for numerical computations. First, we can find the closed-form transition matrix of the associated ODE as

$$\Psi_{ts} = \exp\left(\int_s^t -\left(\frac{1}{2}\beta(\tau)I + cL\right)\mathrm{d}\tau\right) = \exp\left(-cL(t-s) - \frac{1}{2}\int_s^t \beta(\tau)\,\mathrm{d}\tau\right) \tag{5.29}$$

where the integral can be easily obtained as

$$\int_s^t \beta(\tau)\,\mathrm{d}\tau = \left[\frac{1}{2}\tau^2(\beta_{\max} - \beta_{\min}) + \tau\beta_{\min}\right]_s^t =: \tilde{\beta}_{ts}. \tag{5.30}$$

This allows to compute the mean of the transition kernel $m_{t|s}$ given an initial solution y_s. For the covariance kernel, we have

$$K_{t|s} = \Psi_t\left(\int_s^t \beta(\tau)\Psi_\tau^{-2}\,\mathrm{d}\tau\right)\Psi_t^\top$$

where the integral can be expressed as

$$\int_s^t \beta(\tau)\Psi_\tau^{-2}\,\mathrm{d}\tau = \int_s^t \left(\tau(\beta_{\max} - \beta_{\min}) + \beta_{\min}\right)\exp\left(2cL\tau + \tilde{\beta}_{\tau 0}\right)\mathrm{d}\tau$$

$$= \left[\left(\tau(\beta_{\max} - \beta_{\min}) + \beta_{\min}\right)v(\tau)\right]\Big|_s^t - (\beta_{\max} - \beta_{\min})\int_0^s v(\tau)\,\mathrm{d}\tau$$

(integration by parts)

Here, we denote $v(\tau) = \int \exp\big(2cL\tau + \tilde{\beta}_{\tau 0}\big)\,\mathrm{d}\tau$, thus $v'(\tau) := \exp\big(2cL\tau + \tilde{\beta}_{\tau 0}\big)$, which does not have a simple closed-form, we need to compute it numerically. This gives the covariance kernel. Following the similar procedures, we can compute the cross covariance K_{t_1,t_2} of the conditional process $Y_t|Y_0 = y_0$.

5.B.3 Other Topological Dynamics

We may consider *fractional Laplacian* in $\mathcal{T}$SHeat which allows for a more efficient exploration of the network [Riascos & Mateos, 2014] due to its non-local nature.

For $\mathcal{T}$SHeat, we can further allow *heterogeneous heat diffusion* on the edge space as follows

$$\mathrm{d}Y_t = -(c_1 L_{\mathrm{d}} + c_2 L_{\mathrm{u}}) Y_t \,\mathrm{d}t + g_t \,\mathrm{d}W_t, \tag{5.31}$$

by setting $H_t = -(c_1 L_{\mathrm{d}} + c_2 L_{\mathrm{u}})$, with $c_1, c_2 > 0$. Here, the diffusion rates are different for the different edge-adajcency types encoded in L_{d} and L_{u}. This in fact can be generalized to using a more general topological convolution operator, if $L := L_{\mathrm{d}} + L_{\mathrm{u}}$ consists of the down and up parts,

$$H_t = \sum_{k=0}^{K_1} h_k^1(t) L_{\mathrm{d}}^k + \sum_{k=0}^{K_2} h_k^2(t) L_{\mathrm{u}}^k \tag{5.32}$$

where $h_k^1(t), h_k^2(t)$ are the coefficients of the topological convolution. We refer to Yang et al. [2022b] for more details on its expressive power compared to $\mathcal{T}$SDE.

Beyond $\mathcal{T}$SDE: Instead of first-order dynamics, we can use higher-order dynamics such as wave equations. *Graph wave equations* [Chung et al., 2007] have been used for building more expressive graph neural networks [Poli et al., 2021] and its stochastic variant for modeling graph-time GPs [Nikitin et al., 2022]. Moreover, we may allow the interactions between node and edge signals in which the dynamics is defined over the direct sum of the two spaces. While not considered in this work, we refer to [Alain et al., 2024] for such cases to define topological GPs on SCs.

5.C Towards the Optimality of Topological SBP

Proposition 5.23 ($\mathcal{T}$-Schrödinger System; [Chen et al., 2016; Jamison, 1975]). *The optimal solution of $\mathcal{T}$SBP$_{static}$ has the form $\mathbb{P}_{01} = \int_{\mathbb{R}^n \times \mathbb{R}^n} \hat{\varphi}_0(x_0) p_{1|0}(x_1|x_0) \varphi_1(x_1) \,\mathrm{d}x_0 \,\mathrm{d}x_1$ with φ and $\hat{\varphi}$ satisfying the system*

$$\varphi_t(x_t) = \int_{\mathbb{R}^n} p_{1|t}(x_1|x_t) \varphi_1(x_1) \,\mathrm{d}x_1, \quad \hat{\varphi}_t(x_t) = \int_{\mathbb{R}^n} p_{t|0}(x_t|x_0) \hat{\varphi}_0(x_0) \,\mathrm{d}x_0 \tag{5.33}$$

where $p_{t|s}(y|x) = \mathcal{N}(y; \mu_{t|s}, K_{t|s})$ is the Gaussian transition density [cf. Lemma 5.19] of $\mathcal{T}$SDE with drift in (5.1). Moreover, the time-marginal at t can be factored as $\mathbb{P}_t(x) = \varphi_t(x)\hat{\varphi}_t(x)$.

Proof. This is a direct result of the Schrödinger system in Theorem 5.13 by replacing the Markov kernel by that [cf. Lemma 5.19] of the $\mathcal{T}$SDE. □

From this system, we see that the optimal path measure has its marginal $\mathbb{P}_t$ factorized into two time-marginals φ_t and $\hat{\varphi}_t$, which are both governed by the $\mathcal{T}$SDE.

5.C.1 Variational Formulations of $\mathcal{T}$SBP

By Girsanov's theorem, the $\mathcal{T}$SBP can be formulated as the minimum energy SOC problem:

$$\min_{b_t} \mathbb{E}\left[\frac{1}{2}\int_0^1 \|b(t, X_t)\|^2 \,\mathrm{d}t\right], \quad \text{s.t.} \begin{cases} \mathrm{d}X_t = [f_t + g_t b(t, X_t)] \,\mathrm{d}t + g_t \,\mathrm{d}W_t \\ X_0 \sim \nu_0, X_0 \sim \nu_1 \end{cases} \tag{$\mathcal{T}$SBP$_{\mathrm{soc}}$}$$

where $b_t \equiv b(t, X_t)$ is the control function. The SDE constraint in $\mathcal{T}$SBP$_{\text{soc}}$ is also known as the *controlled* SDE, in comparison to the *uncontrolled* reference $\mathcal{T}$SDE. It further leads to the variational problem

$$\min_{(b_t,\nu_t)} \frac{1}{2}\int_0^1 \int_{\mathbb{R}^n} \|b_t\|^2 \nu(t,x)\,\mathrm{d}x\,\mathrm{d}t, \text{ s.t.} \begin{cases} \partial_t \nu_t + \nabla\cdot[\nu_t(f_t + g_t b_t)] = \frac{1}{2} g_t^2 \Delta \nu_t \\ \nu(0,x) = \nu_0, \nu(1,x) = \nu_1 \end{cases} \tag{$\mathcal{T}$SBP$_{\text{var}}$}$$

where $\nu_t \equiv \nu(t,x) \equiv \mathbb{P}_t$ is the time-marginal of $\mathbb{P}$ and follows some PDE constraint, which is the FPK equation of the SDE constraint in $\mathcal{T}$SBP$_{\text{soc}}$.

Theorem 5.24 ($\mathcal{T}$SBP Optimality; Caluya & Halder [2021]; Léonard [2014]). *Assume*[2] *that $g(t)$ is uniformly lower-bounded and $f(t,x;L)$ satisfies Lipschitz conditions with at most linear growth in x. Let $\varphi_t \equiv \varphi(t,x)$ and $\hat{\varphi}_t \equiv \hat{\varphi}(t,x)$ be the solutions to the pair of PDEs*

$$\begin{cases} \partial_t \varphi_t = -(\nabla\varphi_t)^\top f_t - \frac{1}{2} g_t^2 \Delta \varphi_t \\ \partial_t \hat{\varphi}_t = -\nabla\cdot(\hat{\varphi}_t f_t) + \frac{1}{2} g_t^2 \Delta \hat{\varphi}_t, \end{cases} \quad \text{s.t. } \varphi(0,\cdot)\hat{\varphi}(0,\cdot) = \nu_0, \varphi(1,\cdot)\hat{\varphi}(1,\cdot) = \nu_1. \tag{5.34}$$

*Then, the optimal control in $\mathcal{T}$SBP$_{\text{var}}$ is $b_t^\star = g_t^2 \nabla \log \varphi_t$ and the optimal path measure is $\nu_t = \mathbb{P}_t = \varphi_t \hat{\varphi}_t$. Moreover, the solution to $\mathcal{T}$SBP can be represented by the path measure of the following coupled (*forward-backward*) $\mathcal{T}$SDEs*

$$\mathrm{d}X_t = [f_t + g_t^2 \nabla \log \varphi(t, X_t)]\,\mathrm{d}t + g_t\,\mathrm{d}W_t, \quad X_0 \sim \nu_0, \tag{5.35a}$$

$$\mathrm{d}X_t = [f_t - g_t^2 \nabla \log \hat{\varphi}(t, X_t)]\,\mathrm{d}t + g_t\,\mathrm{d}W_t, \quad X_1 \sim \nu_1, \tag{5.35b}$$

where $\nabla \log \varphi(t, X_t)$ and $\nabla \log \hat{\varphi}(t, X_t)$ are the forward and backward optimal drifts, respectively.

Proof. The proof is an adaption of Caluya & Halder [2021] to the topological setting by specifying the topology-aware drift in 5.1. From the first oprder optimality conditions for the SOC formulation $\mathcal{T}$SBP$_{\text{var}}$, we can obtain a coupled system of nonlinear PDEs for ψ_t (the potential function of b_t, i.e., $b_t = \nabla\psi_t$) and ν_t, which are known as the Hamilton-Jacobi-Bellman (HJB) and FPK equations, respectively, as well as the optimal control $b_t^\star = g_t^2 \nabla \log \varphi_t$. Via the Hopf-Cole transform, this system returns (5.34) [Caluya & Halder, 2021, Thm 2]. Then, by substituting the optimal control into the constraint in $\mathcal{T}$SBP$_{\text{soc}}$, one can obtain the forward SDE, and the backward SDE can be derived from the time-reversal of the forward SDE [Anderson, 1982; Nelson, 2020]. □

Using nonlinear Feynman-Kac formula (or applying Itô's formula on $\log \varphi_t$ and $\log \hat{\varphi}_t$), the PDE system (5.34) admits the SDEs [Chen et al., 2022b]

$$\mathrm{d}\log \varphi_t = \frac{1}{2}\|Z_t\|^2\,\mathrm{d}t + Z_t^\top\,\mathrm{d}W_t, \tag{5.36}$$

$$\mathrm{d}\log \hat{\varphi}_t = \left(\frac{1}{2}\|\hat{Z}_t\|^2 + \nabla\cdot(g_t\hat{Z}_t - f_t) + \hat{Z}_t^\top Z_t\right)\mathrm{d}t + \hat{Z}_t^\top\,\mathrm{d}W_t \tag{5.37}$$

where $Z_t \equiv g_t \nabla \log \varphi_t(X_t)$ and $\hat{Z}_t \equiv g_t \nabla \log \hat{\varphi}_t(X_t)$. This results in Proposition 5.5.

[2] The nonexplosive Lipschitz condition on f_t rules out the finite-time blow up of the sample paths of the SDE, and ensures the existance and uniqueness of the solutions. It, together with the uniformly lower-bounded diffusion g_t, guarantees the transition kernel $p_{t|s}$ is positive and everwhere continuous [Särkkä & Solin, 2019].

5.C.2 Wasserstein Gradient Flow Interpretation

The gradient flow of a funcitional over the space of probability measures with Wasserstein metric, i.e., the Wasserstein gradient flow (WGF), is fundamentally linked to FPK equations [Ambrosio et al., 2008; Otto, 2001]. In the following, we show that solving the $\mathcal{T}$SBP with $\mathcal{T}$SHeat$_{\text{BM}}$ reference amounts to solving the WGF of some functional on a probability measure ν.

Theorem 5.25. *Consider the $\mathcal{T}$SBP$_{var}$ with the reference process $\mathcal{T}$SHeat. The SB optimality [cf. (5.34)] respects a pair of FPK equations of the form*

$$\partial_t \hat{\varphi}_t(x) = \nabla \cdot (cLx\hat{\varphi}_t(x)) + \frac{1}{2}g^2 \Delta \hat{\varphi}_t(x), \quad \hat{\varphi}_0(x) = \hat{\varphi}_0(x) \tag{5.38a}$$

$$\partial_t \rho_t(x) = \nabla \cdot (cLx\rho_t(x)) + \frac{1}{2}g^2 \Delta \rho_t(x), \quad \rho_0(x) = \varphi_1(x)\exp(2cx^\top Lx/g^2) \tag{5.38b}$$

Therefore, the Wasserstein gradient flow of $\mathcal{F}(v)$ recovers the paired PDE in solving $\mathcal{T}$SBP$_{var}$

$$\mathcal{F}(\nu) = c\int_{\mathbb{R}^n} \frac{1}{2}x^\top Lx \cdot \nu(x)\,\mathrm{d}x + \frac{1}{2}g^2 \int_{\mathbb{R}^n} \nu \log \nu \,\mathrm{d}x := c\mathbb{E}_\nu[\mathcal{D}(x)] + \frac{1}{2}g^2\mathcal{S}(\nu) \tag{5.39}$$

where $\mathcal{D}(x) = \frac{1}{2}x^\top Lx$ is the Dirichlet energy of x and $\mathcal{S}(\nu)$ is the negative differential entropy.

Proof. First, from Theorem 5.24, we have the PDE system in (5.34) which can be rewritten as the pair of PDEs in (5.38) by applying Caluya & Halder [2021, Thm 3]. Both PDEs are of the following FPK form with $V(x) := \frac{1}{2}cx^\top Lx$ on some density p_t

$$\partial_t p_t(x) = \nabla \cdot (p_t(x)\nabla V(x)) + \frac{1}{2}g^2 \Delta p_t(x), \tag{5.40}$$

for some initial condition. We can view $V(x)$ as the potential energy of some function $f_t(x)$. Here, we have $f_t(x) = -cLx_t = -\nabla V(x)$. Then, from the seminal work Jordan et al. [1998], the flows generated by the PDEs in (5.38) (both of the FPK form) can be seen as the gradient descent of the Lyapunov functional $\mathcal{F}(\cdot)$ in the following form

$$\mathcal{F}(\cdot) = c\int_{\mathbb{R}^n} \frac{1}{2}x^\top Lx \cdot (\cdot)\,\mathrm{d}x + \frac{1}{2}g^2 \int_{\mathbb{R}^n} (\cdot)\log(\cdot)\,\mathrm{d}x := c\mathbb{E}_{(\cdot)}[\mathcal{D}(x)] + \frac{1}{2}g^2\mathcal{S}(\cdot) \tag{5.41}$$

with respect to the 2-Wasserstein distance in the space $\mathcal{P}_2(\mathbb{R}^n)$ of probability measures on $\mathbb{R}^n$ with finite second moments. Here, $(\cdot)$ can be $\hat{\varphi}_t$ or p_t. □

We note that it is also possible to obtain the associated functional for the $\mathcal{T}$SBP with more general reference $\mathcal{T}$SDE based on the similar argument, but it would be more involved and lead to a time-dependent functional [Ferreira & Valencia-Guevara, 2018].

5.D The closed-form of Gaussian Topological Schrödinger Bridges [Theorems 5.6 and 5.7] (Proofs and others)

For convenience, we state the Gaussian $\mathcal{T}$SBP

$$\min D_{\mathsf{KL}}(\mathbb{P}\,\|\,\mathbb{Q}_{\mathcal{T}}), \quad \text{s.t. } \mathbb{P}\in\mathcal{P}(\Omega), \nu_0=\mathcal{N}(\mu_0,\Sigma_0), \nu_1=\mathcal{N}(\mu_1,\Sigma_1) \tag{G$\mathcal{T}$SBP}$$

and its *static* problem

$$\min D_{\mathsf{KL}}(\mathbb{P}_{01}\,\|\,\mathbb{Q}_{\mathcal{T}01}), \quad \text{s.t. } \mathbb{P}_{01}\in\mathcal{P}(\mathbb{R}^n\times\mathbb{R}^n), \mathbb{P}_{0\cdot}=\nu_0, \mathbb{P}_{\cdot 1}=\nu_1. \tag{G$\mathcal{T}$SBP$_{\text{static}}$}$$

We also restate the main results of the Gaussian $\mathcal{T}$SBP.

Theorem 5.6. *Denote by $\mathbb{P}$ the solution to G$\mathcal{T}$SBP with $\nu_0=\mathcal{N}(\mu_0,\Sigma_0)$ and $\nu_1=\mathcal{N}(\mu_1,\Sigma_1)$. Then, $\mathbb{P}$ is the path measure of a* Markov Gaussian process *whose marginal $X_t\sim\mathcal{N}(\mu_t,\Sigma_t)$ admits an expression in terms of the initial and final variables, X_0, X_1, as follows*

$$X_t=\bar{R}_tX_0+R_tX_1+\xi_t-R_t\xi_1+\Gamma_tZ \tag{5.5}$$

where $Z\sim\mathcal{N}(0,I)$ is standard Gaussian, independent of (X_0,X_1), and we have

$$\begin{aligned} &R_t=K_{t1}K_{11}^{-1}, \quad \bar{R}_t=\Psi_t-R_t\Psi_1,\\ &\Gamma_t:=\mathrm{Cov}[Y_t|(Y_0,Y_1)]=K_{tt}-K_{t1}K_{11}^{-1}K_{1t}. \end{aligned} \tag{5.6}$$

Corollary 5.26 (Marginal Statistics). *The time marginal variable X_t in (5.5) of the optimal solution to G$\mathcal{T}$SBP has the mean and covariance as follows*

$$\mu_t=\bar{R}_t\mu_0+R_t\mu_1+\xi_t-R_t\xi_1, \tag{5.42a}$$

$$\Sigma_t=\bar{R}_t\Sigma_0\bar{R}_t^\top+R_t\Sigma_1R_t^\top+\bar{R}_tCR_t^\top+R_tC^\top\bar{R}_t^\top+\Gamma_t, \tag{5.42b}$$

where $C=\Psi_1^{-1}K_{11}^{1/2}\tilde{C}K_{11}^{1/2}$ with

$$\begin{aligned} &\tilde{C}=\frac{1}{2}(\tilde{\Sigma}_0^{1/2}\tilde{D}\tilde{\Sigma}_0^{-1/2}-I), \qquad \tilde{D}=(4\tilde{\Sigma}_0^{1/2}\tilde{\Sigma}_1\tilde{\Sigma}_0^{1/2}+I)^{1/2},\\ &\tilde{\Sigma}_0=K_{11}^{-1/2}\Psi_1\Sigma_0\Psi_1^\top K_{11}^{-1/2}, \quad \tilde{\Sigma}_1=K_{11}^{-1/2}\Sigma_1K_{11}^{-1/2}. \end{aligned} \tag{5.43}$$

Theorem 5.7 (SDE representation). *Under the optimal $\mathbb{P}$, the process X admits the SDE dynamics:*

$$\begin{aligned} &\mathrm{d}X_t=f_{\mathcal{T}}(t,X_t;L)\,\mathrm{d}t+g_t\,\mathrm{d}W_t, \text{ where}\\ &f_{\mathcal{T}}(t,x;L)=S_t^\top\Sigma_t^{-1}(x-\mu_t)+\dot{\mu}_t \end{aligned} \tag{5.7}$$

with μ_t,Σ_t the mean and covariance of X_t [cf. Corollary 5.26] and we have

$$S_t=P_t-Q_t^\top+H_tK_{tt}-K_{t1}K_{11}^{-1}\Upsilon_t^\top, \tag{5.8}$$

with $P_t=(R_t\Sigma_1+\bar{R}_tC)\dot{R}_t^\top, Q_t=-\dot{\bar{R}}_t(CR_t^\top+\Sigma_0\bar{R}_t^\top), \Upsilon_t=H_tK_{t1}+g_t^2\Psi_t^{-1}\Psi_1^\top$, where C is the covariance of X_0,X_1 in the optimal $\mathbb{P}_{01}$, and $\dot{R}_t,\dot{\bar{R}}_t$ are the time-derivatives of $R_t,\bar{R}_t$.

5.D.1 Preliminaries for the proof

We first introduce the following three lemmas and the definition of infinitesimal generators.

Lemma 5.27 (Central identity of Quantum Field Theory [Zee, 2010]). *For all matrix $M \succ 0$ and all sufficiently regular analytical function v (e.g., polynomials or $v \in \mathcal{C}^\infty(\mathbb{R}^d)$ with compact support), we have*

$$(2\pi)^{-\frac{d}{2}} (\det M)^{-\frac{1}{2}} \int_{\mathbb{R}^d} v(x) \exp\left(-\frac{1}{2} x^\top M x\right) \mathrm{d}x = \exp\left(\frac{1}{2} \partial_x^\top M^{-1} \partial_x\right) v(x)\Big|_{x=0} \tag{5.44}$$

where $\exp(D) = I + D + \frac{1}{2}D^2 + \cdots$, for a differential operator D.

Lemma 5.28 (Conditional Gaussians). *Let $(Y_0, Y_1) \sim \mathcal{N}\left(\begin{bmatrix}\mu_0 \\ \mu_1\end{bmatrix}, \begin{bmatrix}\Sigma_{00} & \Sigma_{01} \\ \Sigma_{10} & \Sigma_{11}\end{bmatrix}\right)$. Then, $Y_0|Y_1 = y$ is Gaussian with*

$$\mathbb{E}[Y_0|Y_1 = y] = \mu_0 + \Sigma_{01}\Sigma_{11}^{-1}(y - \mu_1), \text{ and } \mathrm{Cov}(Y_0|Y_1 = y) = \Sigma_{00} - \Sigma_{01}\Sigma_{11}^{-1}\Sigma_{10}. \tag{5.45}$$

Definition 5.29 (Infinitesimal generator of a stochastic process). For a sufficiently regular time-dependent function $\phi(t,x) \in \mathbb{R}^+ \times \mathbb{R}^n \to \mathbb{R}$, the infinitesimal generator of a stochastic process X_t for $\phi(t,x)$ can be defined as

$$\mathcal{A}_t \phi(t,x) = \lim_{h \to 0} \frac{\mathbb{E}[\phi(t+h, X_{t+h})|X_t = x] - \phi(t,x)}{h}. \tag{5.46}$$

For an Itô process defined as the solution to the SDE

$$\mathrm{d}X_t = f(t, X_t)\,\mathrm{d}t + g(t, X_t)\,\mathrm{d}W_t, \tag{5.47}$$

with $f(t,x), g(t,x) : \mathbb{R}^+ \times \mathbb{R}^n \to \mathbb{R}^n$, the generator is given as

$$\mathcal{A}_t \phi(t,x) = \partial_t \phi(t,x) + f(t,x)^\top \nabla_x \phi(t,x) + \frac{1}{2} \mathrm{Tr}[g(t,x) g(t,x)^\top \Delta \phi(t,x)] \tag{5.48}$$

where $\Delta := \nabla_x^2$ is the Euclidean Laplacian operator.

5.D.2 Outline of the proof

Our proofs follow the idea from Bunne et al. [2023, Theorem 3].

For Theorem 5.6. We follow the following two steps:

1. We first solve the associated *static* G$\mathcal{T}$SBP. Specifically, we formulate the equivalent E-OT problem, which has the transport cost dependent on the transition kernel of the $\mathcal{T}$SDE. By introducing new variables, we can convert this involved transport cost to a quadratic cost over new variables, thus, converting the G$\mathcal{T}$SBP$_{\text{static}}$ to a classical Gaussian E-OT. Based on the existing results [Janati et al., 2020; Mallasto et al., 2022], we can then obtain the optimal coupling over the transformed variables. The coupling over the original variables can be recovered via an inverse transform.

2. From Theorem 5.11, we can obtain the solution of G$\mathcal{T}$SBP based on that the optimal $\mathbb{P}$ is in the reciprocal class of $\mathbb{Q}_{\mathcal{T}}$, specifically, by composing the static solution with the $\mathbb{Q}_{\mathcal{T}}$-bridge. This is an optimality condition obtained from the reduction to the static problem. From that, we know that the solution $\mathbb{P}$ is a Markov Gaussian process and shares the same bridge as $\mathbb{Q}_{\mathcal{T}}$ [cf. Proposition 5.12]. This further allows us to characterize the mean and covariance of the time-marginal.

For Theorem 5.7. Let $\mathbb{P} \in \mathcal{P}(\Omega)$ be a finite-energy diffusion [Föllmer & Wakolbinger, 1986]; that is, under $\mathbb{P}$, the canonical process X has a (forward) Itô differential. Furthermore, since $\mathbb{P}$ is in the reciprocal class of $\mathbb{Q}_{\mathcal{T}}$, it has the SDE representation in the class of (5.7) from the SOC formulation where the drift $f_{\mathcal{T}}(t, x : L)$ is to be determined. We then proceed the following two steps:

1. For the SDE (5.7) of the optimal process X_t, we first compute its *infinitesimal generator* [Protter, 2005] for a test function $\phi(t,x) \in \mathbb{R}^+ \times \mathbb{R}^n \to \mathbb{R}$ by definition using (5.46).
2. Second, we express the generator in terms of its given solution in (5.48) for the SDE (5.7)

$$\mathcal{A}_t\phi(t,x) = \partial_t\phi(t,x) + f_{\mathcal{T}}^\top \nabla_x \phi(t,x) + \frac{1}{2} g_t^2 \nabla_x^2 \phi(t,x). \tag{5.49}$$

By matching the generators computed in both ways, we then obtain the closed-form of the drift term.

5.D.3 Detailed proof of Theorem 5.6

Step 1: Solve G$\mathcal{T}$SBP$_{\text{static}}$

First, recall that $Y_t|Y_0 = y_0$ is a Gaussian process with mean $m_t := \mathbb{E}(Y_t|y_0)$ and covariance K_{tt} [cf. cond. mean and cond. cross cov], respectively. Thus, we have the transition probability density

$$\begin{aligned}\mathbb{Q}_{\mathcal{T}1|0}(y_1|y_0) &\propto \exp\left(-\frac{1}{2}(y_1 - \Psi_1 y_0 - \xi_1)^\top K_{11}^{-1}(y_1 - \Psi_1 y_0 - \xi_1)\right) \\ &\propto \exp\left(-\frac{1}{2}(y_1 - m_1)^\top K_{11}^{-1}(y_1 - m_1)\right).\end{aligned} \tag{5.50}$$

By introcuding the variables $\tilde{Y}_0 = K_{11}^{-\frac{1}{2}}(\Psi_1 Y_0 + \xi_1)$ and $\tilde{Y}_1 = K_{11}^{-\frac{1}{2}} Y_1$, we have

$$\mathbb{Q}_{\mathcal{T}1|0}(y_1|y_0) \propto \exp\left(-\frac{1}{2}\|\tilde{y}_1 - \tilde{y}_0\|^2\right). \tag{5.51}$$

Furthermore, if the joint distribution $\mathbb{P}_{01}$ has marginals $Y_0 \sim \nu_0$ and $Y_1 \sim \nu_1$, then after the change of variables ($Y_0 \to \tilde{Y}_0$ and $Y_1 \to \tilde{Y}_1$), it gives rise to a joint distribution $\tilde{\mathbb{P}}_{01}$ with marginals $\tilde{Y}_0 \sim \tilde{\nu}_0 = \mathcal{N}(\tilde{\mu}_0, \tilde{\Sigma}_0)$ and $\tilde{Y}_1 \sim \tilde{\nu}_1 = \mathcal{N}(\tilde{\mu}_1, \tilde{\Sigma}_1)$, where

$$\begin{aligned}\tilde{\mu}_0 &= K_{11}^{-\frac{1}{2}}(\Psi_1 \mu_0 + \xi_1), & \tilde{\Sigma}_0 &= K_{11}^{-\frac{1}{2}} \Psi_1 \Sigma_0 \Psi_1^\top K_{11}^{-\frac{1}{2}}, \\ \tilde{\mu}_1 &= K_{11}^{-\frac{1}{2}} \mu_1, & \tilde{\Sigma}_1 &= K_{11}^{-\frac{1}{2}} \Sigma_1 K_{11}^{-\frac{1}{2}}.\end{aligned} \tag{5.52}$$

That is, there is an one-to-one correspondence between $\mathbb{P}_{01}$ and $\tilde{\mathbb{P}}_{01}$. This allows us to expand the objective of G$\mathcal{T}$SBP$_{\text{static}}$ in terms of *minimization* as follows

$$\begin{aligned} D_{\mathsf{KL}}(\mathbb{P}_{01}\|\mathbb{Q}_{\mathcal{T}01}) &= \int_{\mathbb{R}^n\times\mathbb{R}^n} \mathbb{P}_{01}(y_0,y_1)\log\left(\frac{\mathbb{P}_{01}(y_0,y_1)}{\mathbb{Q}_{\mathcal{T}01}(y_0,y_1)}\right)\mathrm{d}y_0\,\mathrm{d}y_1 \\ &= -\int \log(\mathbb{Q}_{\mathcal{T}01}(y_0,y_1))\,\mathrm{d}\mathbb{P}_{01}(y_0,y_1) + \int \log(\mathbb{P}_{01})\,\mathrm{d}\mathbb{P}_{01} \\ &= \frac{1}{2}\int \|\tilde{y}_1-\tilde{y}_0\|^2\,\mathrm{d}\mathbb{P}_{01}(y_0,y_1) + \int \log(\mathbb{P}_{01})\,\mathrm{d}\mathbb{P}_{01} + \text{const. 1} \qquad (\star) \\ &= \frac{1}{2}\int \|\tilde{y}_1-\tilde{y}_0\|^2\,\mathrm{d}\tilde{\mathbb{P}}_{01}(\tilde{y}_0,\tilde{y}_1) + \int \log(\tilde{\mathbb{P}}_{01})\,\mathrm{d}\tilde{\mathbb{P}}_{01} + \text{const. 2} \\ &\equiv D_{\mathsf{KL}}(\tilde{\mathbb{P}}_{01}\|\mathbb{Q}_{\mathcal{T}01}) \end{aligned}$$

where the second last step results from $\int \log(\tilde{\mathbb{P}}_{01})\,\mathrm{d}\tilde{\mathbb{P}}_{01} = \int \log(\mathbb{P}_{01})\,\mathrm{d}\mathbb{P}_{01} + \text{const.}$. To obtain ($\star$), we notice that $\mathbb{Q}_{\mathcal{T}01}(y_0,y_1) = \mathbb{Q}_{\mathcal{T}1|0}(y_1|y_0)\mathbb{Q}_{\mathcal{T}0}(y_0)$, and we have

$$\begin{aligned} &-\int \log(\mathbb{Q}_{\mathcal{T}01}(y_0,y_1))\,\mathrm{d}\mathbb{P}_{01}(y_0,y_1) \\ =&-\int \log(\mathbb{Q}_{\mathcal{T}1|0}(y_1|y_0))\,\mathrm{d}\mathbb{P}_{01}(y_0,y_1) - \int \log(\mathbb{Q}_{\mathcal{T}0}(y_0))\,\mathrm{d}\mathbb{P}_{01}(y_0,y_1) \\ =&-\int \log(\mathbb{Q}_{\mathcal{T}1|0}(y_1|y_0))\,\mathrm{d}\mathbb{P}_{01}(y_0,y_1) - \int \log \mathbb{Q}_{\mathcal{T}0}(y_0)\,\mathrm{d}\mathbb{P}_0(y_0) \end{aligned}$$

where the last equality holds since we can remove the dependence on y_1 in the second term by integrating over y_1, thus, appearing as a constant in the optimization over $\mathbb{P}_{01}$. Moreover, the expression ($\star$) is in fact the equivalent *E-OT* associated to the $\mathcal{T}$SBP

$$\min_{\mathbb{P}_{01}} \frac{1}{2}\int \|y_1-\Psi_1 y_0-\xi_1\|^2_{K_{11}^{-1}}\,\mathrm{d}\mathbb{P}_{01}(y_0,y_1) + \int \log(\mathbb{P}_{01})\,\mathrm{d}\mathbb{P}_{01}. \qquad (\mathcal{T}\text{E-OT})$$

Note that by definition we have $D_{\mathsf{KL}}(\mathbb{P}_{01}\|\nu_0\otimes\nu_1) = \int \log(\mathbb{P}_{01})\,\mathrm{d}\mathbb{P}_{01} - \int \log(\nu_0\otimes\nu_1)\,\mathrm{d}\mathbb{P}_{01} = \int \log(\mathbb{P}_{01})\,\mathrm{d}\mathbb{P}_{01} - \int \log\nu_0\,\mathrm{d}\nu_0 - \int \log\nu_1\,\mathrm{d}\nu_1$ where the last two terms are constants.

Thus, solving G$\mathcal{T}$SBP$_{\text{static}}$ is equivalent to solving the following problem

$$\min_{\tilde{\mathbb{P}}_{01}\in\mathcal{P}(\mathbb{R}^n\times\mathbb{R}^n)} D_{\mathsf{KL}}(\tilde{\mathbb{P}}_{01}\|\mathbb{Q}_{\mathcal{T}01}) \equiv \int \frac{1}{2}\|\tilde{y}_1-\tilde{y}_0\|^2\,\mathrm{d}\tilde{\mathbb{P}}_{01}(\tilde{y}_0,\tilde{y}_1) + \int \log(\tilde{\mathbb{P}}_{01})\,\mathrm{d}\tilde{\mathbb{P}}_{01} \quad (5.53)$$

with $\tilde{\mathbb{P}}_0 = \tilde{\nu}_0$ and $\tilde{\mathbb{P}}_1 = \tilde{\nu}_1$. This is a classical static Gaussian E-OT between $\tilde{\nu}_0$ and $\tilde{\nu}_1$ with $\sigma = 1$. The closed-form solution is given by the joint Gaussian [cf. Theorem 5.8]

$$\tilde{\mathbb{P}}^\star_{01} = \mathcal{N}\left(\begin{bmatrix}\tilde{\mu}_0\\ \tilde{\mu}_1\end{bmatrix}, \begin{bmatrix}\tilde{\Sigma}_0 & \tilde{C}\\ \tilde{C}^\top & \tilde{\Sigma}_1\end{bmatrix}\right) \quad (5.54)$$

where

$$\tilde{C} = \frac{1}{2}(\tilde{\Sigma}_0^{1/2}\tilde{D}\tilde{\Sigma}_0^{-1/2} - I), \quad \tilde{D} = (4\tilde{\Sigma}_0^{1/2}\tilde{\Sigma}_1\tilde{\Sigma}_0^{1/2} + I)^{1/2}. \quad (5.55)$$

Finally, via the inverse transforms $Y_0 = \Psi_1^{-1}(K_{11}^{\frac{1}{2}}\tilde{y}_0 - \xi_1)$ and $Y_1 = K_{11}^{\frac{1}{2}}\tilde{y}_1$, we can obtain the solution to the original problem G$\mathcal{T}$SBP$_{\text{static}}$ as

$$\mathbb{P}_{01}^{\star} \sim \mathcal{N}\left(\begin{bmatrix}\mu_0\\ \mu_1\end{bmatrix}, \begin{bmatrix}\Sigma_0 & C\\ C^\top & \Sigma_1\end{bmatrix}\right) \tag{G$\mathcal{T}$SB$_{\text{static}}$}$$

where $C = \Psi_1^{-1}K_{11}^{\frac{1}{2}}\tilde{C}K_{11}^{\frac{1}{2}}$.

Step 2: From static to dynamic via disintegration formula

From Theorem 5.11, we know that the solution $\mathbb{P}$ to G$\mathcal{T}$SBP shares its bridges with the reference $\mathbb{Q}_{\mathcal{T}}$. We denote by $\mathbb{Q}_{\mathcal{T}}^{y_0y_1}$ the process Y conditioning on $Y_0 = y_0$ and $Y_1 = y_1$ under $\mathbb{Q}_{\mathcal{T}}$, i.e., $Y|y_0, y_1 \sim \mathbb{Q}_{\mathcal{T}}^{y_0y_1} = \mathbb{Q}_{\mathcal{T}}[Y_0 = y_0, Y_1 = y_1]$. It is the bridge of $\mathbb{Q}_{\mathcal{T}}$, following

$$\mathbb{Q}_{\mathcal{T}}(\cdot) = \int_{\mathbb{R}^n\times\mathbb{R}^n} \mathbb{Q}_{\mathcal{T}}^{y_0y_1}(\cdot)\mathbb{Q}_{\mathcal{T}01}(\mathrm{d}y_0\,\mathrm{d}y_1). \tag{5.56}$$

5

In the classical case of Brownian motion $Y = W \sim \mathbb{Q}_W$, $\mathbb{Q}_W^{y_0y_1}$ is often referred to as the *Brownian bridge*. Here, we aim to first find the $\mathbb{Q}_{\mathcal{T}}^{y_0y_1}$-bridge, and then construct the optimal solution $\mathbb{P}^\star$ by composing the static solution $\mathbb{P}_{01}^\star$ with the $\mathbb{Q}_{\mathcal{T}}^{y_0y_1}$-bridge [cf. **??** in Theorem 5.11].

From the transition kernel in Lemma 5.4, we have the conditional distributions $Y_t|y_0 \sim \mathcal{N}(m_t, K_{tt})$ and $Y_1|y_0 \sim \mathcal{N}(m_1, K_{11})$. Thus, the joint distribution of Y_t and Y_1 given y_0 follows

$$Y_t, Y_1|y_0 \sim \mathcal{N}\left(\begin{bmatrix}m_t\\ m_1\end{bmatrix}, \begin{bmatrix}K_{tt} & K_{t1}\\ K_{1t} & K_{11}\end{bmatrix}\right). \tag{5.57}$$

Applying Lemma 5.28, we know that $Y_t|y_0, Y_1 = y_1$ is Gaussian with mean

$$\begin{aligned}\mathbb{E}(Y_t|y_0, Y_1 = y_1) &= m_t + K_{t1}K_{11}^{-1}(y_1 - m_1)\\ &= \Psi_t y_0 + \xi_t + K_{t1}K_{11}^{-1}(y_1 - \Psi_1 y_0 - \xi_1)\\ &= (\Psi_t - K_{t1}K_{11}^{-1}\Psi_1)y_0 + K_{t1}K_{11}^{-1}y_1 + \xi_t - K_{t1}K_{11}^{-1}\xi_1\\ &\triangleq \bar{R}_t y_0 + R_t y_1 + \xi_t - R_t\xi_1\end{aligned} \tag{5.58}$$

where we recall the definitions of R_t and $\bar{R}_t$ in (5.8), and covariance

$$\Gamma_t := \mathrm{Cov}(Y_t|Y_0 = y_0, Y_1 = y_1) = K_{tt} - K_{t1}K_{11}^{-1}K_{1t}. \tag{5.59}$$

Since a Gaussian process is completely determined by its mean and covariance, we have

$$Y_t|Y_0, Y_1 \overset{\text{law}}{=} \bar{R}_t Y_0 + R_t Y_1 + \xi_t - R_t\xi_1 + \Gamma_t Z \sim \mathbb{Q}_{\mathcal{T}t}^{y_0y_1} \tag{5.60}$$

where $Z \sim \mathcal{N}(0, I)$ is independent of Y_t. Now, the Disintegration of Measures and Theorem 5.11 allow us to construct the solution to G$\mathcal{T}$SBP by first generating $(X_0, X_1) \sim \mathbb{P}_{01}^\star$

in $G\mathcal{T}SB_{\text{static}}$, then connecting X_0 and X_1 using the $\mathbb{Q}_{\mathcal{T}}^{y_0 y_1}$-bridge. This is equivalent to, for $X_0 \sim \nu_0, X_1 \sim \nu_1$ and $Z \sim \mathcal{N}(0, I), Z \perp (X_0, X_1)$, building a process as

$$X_t \overset{\text{law}}{=} \bar{R}_t X_0 + R_t X_1 + \xi_t - R_t \xi_1 + \Gamma_t Z \sim \mathbb{P}^\star_t, \tag{5.61}$$

which in fact is a *stochastic interpolant* for stochastic processes over topological domains, generalizing the same notion in Euclidean domains in Albergo et al. [2024, Definition 1]. *Note that* since $\mathbb{Q}_{\mathcal{T}}$ is a stochastic process following an Itô SDE, which is a Markov process, the solution $\mathbb{P}$ is also a Markov process [cf. Proposition 5.12]. Finally, we obtain the mean and covariance of the time-marginal X_t as

$$\begin{aligned}
\mu_t &= \bar{R}_t \mu_0 + R_t \mu_1 + \xi_t - R_t \xi_1, \\
\Sigma_t &= \bar{R}_t \Sigma_0 \bar{R}_t^\top + R_t \Sigma_1 R_t^\top + \bar{R}_t C R_t^\top + R_t C^\top \bar{R}_t^\top + K_{tt} - K_{t1} K_{11}^{-1} K_{1t}.
\end{aligned} \tag{5.62}$$

This concludes the proofs of Theorem 5.6 and Corollary 5.26.

5.D.4 Detailed Proof of Theorem 5.7

Step 1: Compute the infinitesimal generator of X_t by definition

For some time-varying function $\phi(t, x)$, by definition, the infinitesimal generator of X_t is given by (5.46). Since X_t is a Gaussian process, we could express the conditional expectation using Lemma 5.28. As we are only interested in the terms that are of order $O(h)$, we then ignore the higher-order terms. First, we compute the first-order approximation of Σ_t in (5.42)

$$\begin{aligned}
\dot{\Sigma}_t = &\dot{\bar{R}}_t \Sigma_0 \bar{R}_t^\top + \bar{R}_t \Sigma_0 \dot{\bar{R}}_t^\top + \dot{R}_t \Sigma_1 R_t^\top + R_t \Sigma_1 \dot{R}_t^\top \\
&+ \dot{\bar{R}}_t C R_t^\top + \bar{R}_t C \dot{R}_t^\top + \dot{R}_t C^\top \bar{R}_t^\top + R_t C^\top \dot{\bar{R}}_t^\top \\
&+ \partial_t K_{tt} - (\partial_t K_{t1}) K_{11}^{-1} K_{1t} - (K_{t1} K_{11}^{-1}) \partial_t K_{1t} \\
\triangleq &\, (P_t^\top + P_t) - (Q_t + Q_t^\top) + \partial_t K_{tt} - (\partial_t K_{t1}) K_{11}^{-1} K_{1t} - (K_{t1} K_{11}^{-1}) \partial_t K_{1t}
\end{aligned} \tag{5.63}$$

where at the last equality we recall the definitions of P_t and Q_t in (5.8). Next, denote by $\Sigma_{t,t+h}$ the covariance process of X_t evaluated at t and $t+h$. We can estimate $\Sigma_{t,t+h}$ up to the first order of $o(h)$ as

$$\begin{aligned}
\Sigma_{t,t+h} :=& \mathbb{E}[(X_t - \mu_t)(X_{t+h} - \mu_{t+h})^\top] \\
=& \bar{R}_t \Sigma_0 \bar{R}_{t+h}^\top + R_t \Sigma_1 R_{t+h}^\top + \bar{R}_t C R_{t+h}^\top + R_t C^\top \bar{R}_{t+h}^\top + K_{t,t+h} - K_{t1} K_{11}^{-1} K_{1,t+h} \\
=& \Sigma_t + \bar{R}_t \Sigma_0 (\bar{R}_{t+h} - \bar{R}_t)^\top + R_t \Sigma_1 (R_{t+h} - R_t)^\top + \bar{R}_t C (R_{t+h} - R_t)^\top \\
&+ R_t C^\top (\bar{R}_{t+h} - \bar{R}_t)^\top + (K_{t,t+h} - K_{tt}) - K_{t1} K_{11}^{-1} (K_{1,t+h} - K_{1t}) \\
\overset{(a)}{=}& \Sigma_t + h(\bar{R}_t \Sigma_0 \dot{\bar{R}}_t^\top + R_t \Sigma_1 \dot{R}_t^\top + \bar{R}_t C \dot{R}_t^\top + R_t C^\top \dot{\bar{R}}_t^\top) + o(h) \\
&+ (K_{t,t+h} - K_{tt}) - K_{t1} K_{11}^{-1} (K_{1,t+h} - K_{1t}) \\
\overset{(b)}{=}& \Sigma_t + h(P_t - Q_t^\top + \partial_{t_2} K_{t_1,t_2}|_{t_1=t,t_2=t} - K_{t1} K_{11}^{-1} \partial_{t_2} K_{t_1,t_2}|_{t_1=1,t_2=t}) + o(h)
\end{aligned} \tag{5.64}$$

where we obtain (a) by plugging in $\lim_{h\to 0}\frac{1}{h}(R_{t+h}-R_t)=\dot{R}_t$ and $\lim_{h\to 0}\frac{1}{h}(\bar{R}_{t+h}-\bar{R}_t)$; and likewise, we obtain (b) by recognizing the definitions of P_t and $Q_t^\top$ in (5.8) and using the partial derivatives

$$\begin{aligned}\lim_{h\to 0}\frac{1}{h}(K_{t,t+h}-K_{t,t})&=\partial_{t_2}K_{t_1,t_2}|_{t_1=t,t_2=t},\quad t_1\le t_2\\ \lim_{h\to 0}\frac{1}{h}(K_{1,t+h}-K_{1,t})&=\partial_{t_2}K_{t_1,t_2}|_{t_1=1,t_2=t}=\partial_t K_{1t},\quad t_1>t_2.\end{aligned}\tag{5.65}$$

Following the similar procedure, we can obtain a first-order approximation of $\Sigma_{t+h,t}$ as

$$\begin{aligned}\Sigma_{t+h,t}:=&\mathbb{E}[(X_{t+h}-\mu_{t+h})(X_t-\mu_t)^\top]\\ =&\Sigma_t+h(\dot{\bar{R}}_t\Sigma_0\bar{R}_t^\top+\dot{R}_t\Sigma_1R_t^\top+\dot{\bar{R}}_tCR_t^\top+\dot{R}_tC\bar{R}_t^\top)+o(h)\\ &+(K_{t+h,t}-K_{tt})-(K_{t+h,1}-K_{t1})K_{11}^{-1}K_{1t}\\ =&\Sigma_t+h(P_t^\top-Q_t+\partial_{t_1}K_{t_1,t_2}|_{t_1=t,t_2=t}-\partial_{t_1}K_{t_1,t_2}|_{t_1=t,t_2=1}K_{11}^{-1}K_{1t})+o(h),\end{aligned}\tag{5.66}$$

where the partial derivatives should be understood as

$$\begin{aligned}\lim_{h\to 0}\frac{1}{h}(K_{t+h,t}-K_{t,t})&=\partial_{t_1}K_{t_1,t_2}|_{t_1=t,t_2=t},\quad t_1>t_2\\ \lim_{h\to 0}\frac{1}{h}(K_{t+h,1}-K_{t,1})&=\partial_{t_1}K_{t_1,t_2}|_{t_1=t,t_2=1}=\partial_t K_{t1},\quad t_1\le t_2.\end{aligned}\tag{5.67}$$

Since Eqs. (5.63), (5.64) and (5.66) are all involved with the partial derivatives of K_{t_1,t_2}, we can compute them by the closed-form of the transition matrix as

$$\begin{aligned}\partial_{t_1}K_{t_1,t_2}&=\partial_{t_1}\left\{\Psi_{t_1}\left[\int_0^{t_1}g_s^2\Psi_s^{-2}\,\mathrm{d}s\right]\Psi_{t_2}^\top\right\},\quad \text{for } t_1\le t_2\\ &\overset{(a)}{=}\Psi_{t_1}H_{t_1}\left[\int_0^{t_1}g_s^2\Psi_s^{-2}\,\mathrm{d}s\right]\Psi_{t_2}^\top+g_{t_1}^2\Psi_{t_1}^{-1}\Psi_{t_2}^\top\\ &\overset{(b)}{=}H_{t_1}K_{t_1,t_2}+g_{t_1}^2\Psi_{t_1}^{-1}\Psi_{t_2}^\top,\quad \text{for } t_1\le t_2,\end{aligned}\tag{5.68}$$

where we use the symmetry of Ψ_t. At (a) we use

$$\partial_t\Psi_t=\partial_t\exp\Big(\int_0^t H_s\,\mathrm{d}s\Big)=\exp\Big(\int_0^t H_s\,\mathrm{d}s\Big)H_t=\Psi_tH_t,\tag{5.69}$$

and at (b) we use the commutativity of H_t and Ψ_t [cf. Lemmas 5.14 and 5.18]. Similarly, we have

$$\begin{aligned}\partial_{t_2}K_{t_1,t_2}&=\partial_{t_2}\left\{\Psi_{t_1}\left[\int_0^{t_1}g_s^2\Psi_s^{-2}\,\mathrm{d}s\right]\Psi_{t_2}^\top\right\}=K_{t_1,t_2}H_{t_2}^\top,\quad \text{for } t_1\le t_2\\ \partial_{t_1}K_{t_1,t_2}&=\partial_{t_1}\left\{\Psi_{t_1}\left[\int_0^{t_2}g_s^2\Psi_s^{-2}\,\mathrm{d}s\right]\Psi_{t_2}^\top\right\}=H_{t_1}K_{t_1,t_2},\quad \text{for } t_1>t_2\\ \partial_{t_2}K_{t_1,t_2}&=\partial_{t_2}\left\{\Psi_{t_1}\left[\int_0^{t_2}g_s^2\Psi_s^{-2}\,\mathrm{d}s\right]\Psi_{t_2}^\top\right\}=g_{t_2}^2\Psi_{t_1}\Psi_{t_2}^{-1}+K_{t_1,t_2}H_{t_2}^\top,\quad \text{for } t_1>t_2.\end{aligned}\tag{5.70}$$

We notice that $(\partial_t K_{1t})^\top = \partial_t K_{t1} = g_t^2 \Psi_t^{-1} \Psi_1^\top + H_t K_{t1} K_{11}^{-1} K_{1t}$. Now, by introducing in (5.64) the variable

$$\begin{aligned} S &\triangleq P_t - Q_t^\top + \partial_{t_2} K_{t_1,t_2}|_{t_1=t,t_2=t} - K_{t1} K_{11}^{-1} \partial_{t_2} K_{t_1,t_2}|_{t_1=1,t_2=t} \\ &= P_t - Q_t^\top + K_{tt} H_t^\top - K_{t1} K_{11}^{-1} (g_t^2 \Psi_1 \Psi_t^{-1} + K_{1t} H_t^\top) \\ &= P_t - Q_t^\top + K_{tt} H_t^\top - K_{t1} K_{11}^{-1} (\partial_t K_{1t}), \end{aligned} \tag{5.71}$$

we can then express the covariance process as

$$\Sigma_{t,t+h} = \Sigma_t + h S_t + o(h), \tag{5.72}$$

and

$$\begin{aligned} \Sigma_{t+h,t} &= \Sigma_t + h(P_t^\top - Q_t + H_t^\top K_{tt} - (g_t^2 \Psi_t^{-1} \Psi_1^\top + H_t K_{t1}) K_{11}^{-1} K_{1t}) + o(h) \\ &= \Sigma_t + h(P_t^\top - Q_t + H_t^\top K_{tt} - (\partial_t K_{t1}) K_{11}^{-1} K_{1t}) + o(h) \\ &= \Sigma_t + h S_t^\top + o(h). \end{aligned} \tag{5.73}$$

Lastly, using Lemma 5.28, we see tha variable $X_{t+h}|X_t = x$ is a Gaussian process with mean

$$\begin{aligned} \breve{\mu}_{t+h} :&= \mu_{t+h} + \Sigma_{t+h,t} \Sigma_t^{-1} (x - \mu_t) \\ &\overset{(a)}{=} \mu_t + h\dot{\mu}_t + (\Sigma_t + h S_t^\top) \Sigma^{-1} (x - \mu_t) + o(h) \\ &= \mu_t + h\dot{\mu}_t + (I + h S_t^\top \Sigma_t^{-1})(x - \mu_t) + o(h) \\ &= x + h(\dot{\mu}_t + S_t^\top \Sigma_t^{-1} (x - \mu_t)) + o(h) \end{aligned} \tag{5.74}$$

where in (a) we used $\Sigma_t = \Sigma_t^\top$, and covariance

$$\begin{aligned} \breve{\Sigma}_{t+h} &= \Sigma_{t+h} - \Sigma_{t+h,t} \Sigma_t^{-1} \Sigma_{t,t+h} \\ &= \Sigma_t + h\dot{\Sigma}_t - (\Sigma_t + h S_t^\top) \Sigma_t^{-1} (\Sigma_t + h S_t) + o(h) \\ &= \Sigma_t + h\dot{\Sigma}_t - (\Sigma_t + h S_t^\top + h S_t) + o(h) \\ &\overset{(b)}{=} h(\dot{\Sigma}_t - S_t - S_t^\top) + o(h). \end{aligned} \tag{5.75}$$

By seeing K_{tt} as a matrix function of t, we have

$$\partial_t K_{tt} = \partial_t \left\{ \Psi_t \left[\int_0^t g_s^2 \Psi_s^{-2} \, \mathrm{d}s \right] \Psi_t^\top \right\} = H_t K_{tt} + g_t^2 I + K_{tt} H_t^\top. \tag{5.76}$$

This reduces (5.63) to

$$\dot{\Sigma}_t = (P_t^\top + P_t) - (Q_t + Q_t^\top) + \partial_t K_{tt} - (\partial_t K_{t1}) K_{11}^{-1} K_{1t} - (K_{t1} K_{11}^{-1}) \partial_t K_{1t} \tag{5.77}$$

5

where the last two items appear in $S_t^\top$ and S_t, respectively. Thus, we obtain

$$\begin{aligned}\check{\Sigma}_{t+h} =& h(\dot{\Sigma}_t - S_t - S_t^\top) + o(h)\\ =& h\Big\{ \big[(P_t^\top + P_t) - (Q_t + Q_t^\top) + \partial_t K_{tt} - (\partial_t K_{t1}) K_{11}^{-1} K_{1t} - (K_{t1} K_{11}^{-1}) \partial_t K_{1t}\big]\\ & - \big[P_t - Q_t^\top + K_{tt} H_t^\top - K_{t1} K_{11}^{-1} (\partial_t K_{1t})\big]\\ & - \big[P_t^\top - Q_t + H_t^\top K_{tt} - (\partial_t K_{t1}) K_{11}^{-1} K_{1t}\big]\Big\} + o(h)\\ =& h g_t^2 I + o(h).\end{aligned} \tag{5.78}$$

We can now compute $\mathbb{E}[\phi(t+h, X_{t+h})|X_t = x]$ as follows

$$\begin{aligned}&\mathbb{E}[\phi(t+h, X_{t+h})|X_t = x]\\ &= (2\pi)^{\frac{d}{2}} (\det \check{\Sigma}_{t+h})^{-\frac{1}{2}} \int_{\mathbb{R}^n} \phi(t+h, x') \cdot \exp\Big(-\frac{1}{2}(x' - \check{\mu}_{t+h})^\top \check{\Sigma}_{t+h}^{-1} (x' - \check{\mu}_{t+h})\Big)\, \mathrm{d}x'\\ &\overset{(a)}{=} (2\pi)^{\frac{d}{2}} (\det \check{\Sigma}_{t+h})^{-\frac{1}{2}} \int_{\mathbb{R}^n} \phi(t+h, \tilde{x} + \check{\mu}_{t+h}) \cdot \exp\Big(-\frac{1}{2}\tilde{x}^\top \check{\Sigma}_{t+h}^{-1} \tilde{x}\Big)\, \mathrm{d}\tilde{x}\end{aligned} \tag{5.79}$$

where in (a) we apply a change-of-variable $\tilde{x} := x' - \check{\mu}_{t+h}$. We further apply Lemma 5.27 and arrive at

$$\begin{aligned}\mathbb{E}[\phi(t+h, X_{t+h})|X_t = x] &= \exp\Big(\frac{1}{2}\partial_{\tilde{x}}^\top \check{\Sigma}_{t+h} \partial_{\tilde{x}}\Big) \phi(t+h, \tilde{x} + \check{\mu}_{t+h})\Big|_{\tilde{x}=0}\\ &\overset{(a)}{=} \Big(I + \frac{1}{2}\partial_{\tilde{x}}^\top \check{\Sigma}_{t+h} \partial_{\tilde{x}} + o(\text{h.o.t.})\Big) \phi(t+h, \tilde{x} + \check{\mu}_{t+h})|_{\tilde{x}=0}\\ &\overset{(b)}{=} \phi(t+h, \check{\mu}_{t+h}) + \frac{1}{2} h g_t^2 \Delta \phi(t+h, \check{\mu}_{t+h}) + o(h),\end{aligned} \tag{5.80}$$

where we expand the power series of $\exp(\frac{1}{2}\partial_{\tilde{x}}^\top \check{\Sigma}_{t+h} \partial_{\tilde{x}})$ and ignore the higher-order-terms in (a), and plug in (5.78) in (b). Recalling $\check{\mu}_{t+h}$ in (5.74), we can expand the Taylor series of $\phi(t+h, \check{\mu}_{t+h})$ in the second variabel at x as

$$\begin{aligned}\phi(t+h, \check{\mu}_{t+h}) =& \phi\big(t+h, x + h(\dot{\mu}_t + S_t^\top \Sigma_t^{-1}(x - \mu_t))\big)\\ =& \phi(t+h, x) + h\langle \nabla \phi(t+h, x), \dot{\mu}_t + S^\top \Sigma_t^{-1}(x - \mu_t)\rangle + o(h).\end{aligned} \tag{5.81}$$

Therefore, we have

$$\begin{aligned}&\mathbb{E}[\phi(t+h, X_{t+h})|X_t = x] =\\ &\phi(t+h, x) + h\langle \nabla \phi(t+h, x), \dot{\mu}_t + S^\top \Sigma_t^{-1}(x - \mu_t)\rangle + \frac{1}{2} h g_t^2 \Delta \phi(t+h, x) + o(h).\end{aligned} \tag{5.82}$$

Now we can express the infinitesimal generator of X_t as

$$\begin{aligned}&\lim_{h\to 0} \frac{\mathbb{E}[\phi(t+h, X_{t+h})|X_t = x] - \phi(t, x)}{h}\\ =& \lim_{h\to 0} \frac{u(t+h, x) - u(t, x)}{h} + \langle \nabla \phi(t, x), \dot{\mu}_t + S^\top \Sigma_t^{-1}(x - \mu_t)\rangle + \frac{1}{2} g_t^2 \Delta \phi(t, x)\\ =& \partial_t u(t, x) + \langle \nabla \phi(t, x), \dot{\mu}_t + S^\top \Sigma_t^{-1}(x - \mu_t)\rangle + \frac{1}{2} g_t^2 \Delta \phi(t, x).\end{aligned} \tag{5.83}$$

Step 2: Match the solution of generator for an Itô SDE

From Caluya & Halder [2021]; Léonard [2014], we search for the optimal solution to $\mathcal{T}$SBP within the class of stochastic processes following an SDE:

$$\mathrm{d}X_t = f_{\mathcal{T}}(t, X_t)\,\mathrm{d}t + g_t\,\mathrm{d}W_t. \tag{5.84}$$

Recalling the solution of an infinitesimal generator in (5.48) for this SDE

$$\mathcal{A}_t\phi(t,x) = \partial_t\phi(t,x) + f_{\mathcal{T}}(t,x)^\top \nabla_x\phi(t,x) + \frac{1}{2}g_t^2\Delta\phi(t,x), \tag{5.85}$$

we then match it with the generator obtained by definition in (5.83). We observe that the two are equivalent if we set

$$f_{\mathcal{T}}(t,x) = \dot{\mu}_t + S^\top\Sigma_t^{-1}(x-\mu_t). \tag{5.86}$$

This concludes the proof of Theorem 5.7.

5

5.D.5 Conditional Distribution of $X_t|X_0$

Corollary 5.30 (Conditional distribution of $X_t|X_0$)**.** *Let X_t be the stochastic process associated to the solution $\mathbb{P}$ to G$\mathcal{T}$SBP. Given an initial sample $x_0 \sim \nu_0$, the conditional distribution $\nu(X_t|X_0 = x_0) \sim \mathcal{N}(\mu_{t|0}, \Sigma_{t|0})$ is Gaussian with*

$$\begin{aligned}
\mu_{t|0} &= \bar{R}_t x_0 + R_t\mu_1 + R_t C^\top\Sigma_0^{-1}(x_0-\mu_0) + \xi_t - R_t\xi_1,\\
\Sigma_{t|0} &= R_t\Sigma_1 R_t^\top - R_t C^\top\Sigma_0^{-1}CR_t^\top + K_{tt} - K_{t1}K_{11}^{-1}K_{1t}.
\end{aligned} \tag{5.87}$$

Similarly, given a final sample x_1, the conditional distribution $\nu(X_t|X_1 = x_1) \sim \mathcal{N}(\mu_{t|1}, \Sigma_{t|1})$ is Gaussian with

$$\begin{aligned}
\mu_{t|1} &= R_t x_1 + \bar{R}_t\mu_0 + \bar{R}_t C\Sigma_1^{-1}(x_1-\mu_1) + \xi_t - R_t\xi_1,\\
\Sigma_{t|1} &= \bar{R}_t\Sigma_0\bar{R}_t^\top - \bar{R}_t C\Sigma_1^{-1}C^\top\bar{R}_t^\top + K_{tt} - K_{t1}K_{11}^{-1}K_{1t}.
\end{aligned} \tag{5.88}$$

Proof. First, recall the stochastic interpolant expression in (5.5) of the solution to G$\mathcal{T}$SBP and its mean μ_t and covariance Σ_t in (5.42). Due to the Gaussian nature of the process, we can write the joint distribution of X_t and X_0 as

$$\begin{bmatrix} X_t \\ X_0 \end{bmatrix} \sim \mathcal{N}\left(\begin{bmatrix} \mu_t \\ \mu_0 \end{bmatrix}, \begin{bmatrix} \Sigma_t & \Sigma_{t,0} \\ \Sigma_{0,t} & \Sigma_0 \end{bmatrix}\right) \tag{5.89}$$

where we work out the covariance between X_t and X_0 below

$$\begin{aligned}
\Sigma_{t,0} &= \mathbb{E}[(X_t-\mu_t)(X_0-\mu_0)^\top]\\
&= \bar{R}_t\Sigma_0 + R_t\mathrm{Cov}(X_1,X_0) + \mathrm{Cov}(\xi_t,X_0) - R_t\mathrm{Cov}(\xi_1,X_0) + \mathrm{Cov}(\zeta_t,X_0)\\
&= \bar{R}_t\Sigma_0 + R_t\mathrm{Cov}(X_1,X_0) && \text{(since } \xi_t \text{ is deterministic, } \zeta_t \perp X_0\text{)}\\
&= \bar{R}_t\Sigma_0 + R_t C^\top. && \text{(by } \mathbb{P}_{01}^* \text{ in G}\mathcal{T}\text{SB}_{\text{static}}\text{)}
\end{aligned}$$

Based on Lemma 5.28, we know that $X_t|X_0 = x_0$ is Gaussian with mean $\mu_{t|0}$ and covariance $\Sigma_{t|0}$ given by

$$\begin{aligned}\mu_{t|0} &= \mu_t + \Sigma_{t,0}\Sigma_0^{-1}(x_0 - \mu_0)\\ \Sigma_{t|0} &= \Sigma_t - \Sigma_{t,0}\Sigma_0^{-1}\Sigma_{0,t}.\end{aligned} \tag{5.90}$$

By substituting the expressions of μ_t and Σ_t from (5.42) and canceling out terms, we complete the proof for $X_t|X_0 = x_0$. The proof for $X_t|X_1 = x_1$ follows similarly. □

5.E Topological SB Generative Models

Here, we provide more details on the $\mathcal{T}$SB-based models. First, we give the likelihood of the model which allows for the training objective in (5.9), and the probability flow ODEs corresponding to the FB-$\mathcal{T}$SDEs in (5.3). Then, we discuss the variants of score-based and diffusion bridges models for topological signals, as well as their training objectives, with the goal of illustrating how $\mathcal{T}$SB-based models connect to these models.

5

5.E.1 Likelihood Training for Topological SBP

The likelihood for the Euclidean SBP by Chen et al. [2022b] extends to the topological case.

Corollary 5.31 (Likelihood for $\mathcal{T}$SB models; Chen et al. [2022b])**.** *Given the optimal solution of $\mathcal{T}$SBP satisfying the FB-$\mathcal{T}$SDE system in (5.3), the log-likelihood of the $\mathcal{T}$SB model at an initial signal sample x_0 can be expressed as*

$$\mathcal{L}_{\mathcal{T}SB}(x_0) = \mathbb{E}[\log \nu_1(X_1)] - \int_0^1 \mathbb{E}\left[\frac{1}{2}\|Z_t\|^2 + \frac{1}{2}\|\hat{Z}_t\|^2 + \nabla\cdot(g_t\hat{Z}_t - f_t) + \hat{Z}_t^\top Z_t\right] \mathrm{d}t \tag{5.91}$$

where the expectation is taken over the forward SDE in (5.3) with the initial condition $X_0 = x_0$.

Corollary 5.32 (Probability flow ODE for $\mathcal{T}$SB)**.** *The following ODE characterizes the probability flow of the optimal processes of $\mathcal{T}$SB in (5.3)*

$$\mathrm{d}X_t = \left[f_t + g_t Z_t - \frac{1}{2}g_t(Z_t + \hat{Z}_t)\right]\mathrm{d}t \tag{5.92}$$

and we have that for all t, $\mathbb{P}_t = p_t^{(5.92)}$, i.e., the time marginal of the path measure $\mathbb{P}$ is equal to the probability flow p_t of this ODE.

This is a direct result from the probability flow for general SB [Chen et al., 2022b], which extends the probability flow for score-based models [Song et al., 2020b], and relates to the flow-based training.

5.E.2 Score Matching for Topological Signals

As discussed in Section 5.5 and by Chen et al. [2022b], SB-based models generalize the score-based models [Song et al., 2020b]. Here, we provide a detailed derivation on how a score-based model can be built for topological signals, specifically, on the score matching

objective, since there is no direct literature on this. First, we show in detail that the likelihood training based on (5.9b) returns a score matching objective for topological signals when $Z_t = 0$ and the final ν_1 is a simple Gaussian. The backward training objective in this case becomes

$$\begin{aligned} l(x_0;\hat{\theta}) &= \int_0^1 \mathbb{E}_{X_t\sim(5.3a)}\left[\frac{1}{2}\|\hat{Z}_t^{\hat{\theta}}\|^2 + g_t\nabla\cdot\hat{Z}_t^{\hat{\theta}}\Big|X_0 = x_0\right]\mathrm{d}t \\ &= \int_0^1 \mathbb{E}_{X_t\sim(5.3a)}\left[\frac{1}{2}g_t^2\|s_t(\hat{\theta})\|^2 + g_t^2\nabla\cdot s_t(\hat{\theta})\Big|X_0 = x_0\right]\mathrm{d}t \end{aligned}$$

where we introduce a score function $s_t(\hat{\theta})$ to approximate $\nabla\log p_{t|0}(X_t|X_0 = x_0)$, following $\hat{Z}_t^{\hat{\theta}} = g_t s_t(\hat{\theta})$. Here, $p_{t|0}$ is the transition kernel of the $\mathcal{T}$SDE [cf. Lemma 5.4]. By using the trace estimator [Hutchinson, 1989] to compute the divergence, i.e.,

$$\nabla\cdot s_t(\hat{\theta}) = \mathbb{E}_{u\sim\mathcal{N}(0,I)}[u^\top s_t(\hat{\theta})u], \tag{5.93}$$

and setting the weighting function $\lambda(t) := g_t^2$, we then obtain the *sliced score matching* objective [Song et al., 2020a;b, Eq. 19] for topological signals based on $\mathcal{T}$SDE, which has the form

$$\hat{\theta} = \arg\min \mathbb{E}_{t\sim\mathcal{U}(0,1)}\left\{\lambda(t)\mathbb{E}_{x_0}\mathbb{E}_{x_t}\mathbb{E}_{u\sim\mathcal{N}(0,I)}\left[\frac{1}{2}\|s_t(\hat{\theta})\|^2 + u^\top s_t(\hat{\theta})u\right]\right\}, \tag{5.94}$$

and is equivalent to $l(x_0;\hat{\theta})$. This does not require a closed-form solution for the true score function $\nabla\log p_{t|0}$. The associated FB-$\mathcal{T}$SDEs now become the forward-backward processes for the score-based models

$$\mathrm{d}X_t = f_t\,\mathrm{d}t + g_t\,\mathrm{d}W_t, \tag{5.95}$$

$$\mathrm{d}X_t = (f_t - g_t^2 s_t(\hat{\theta}))\,\mathrm{d}t + g_t\,\mathrm{d}W_t. \tag{5.96}$$

Closed-form score matching For $\mathcal{T}$SHeat$_{\mathrm{BM}}$ and $\mathcal{T}$SHeat$_{\mathrm{VE}}$, since we have their closed-form transition kernels in (5.2), we can use the direct score matching objective [Song et al., 2020b, Eq. 7] to train a score-based model for topological signals

$$\hat{\theta} = \arg\min \mathbb{E}_{t\sim\mathcal{U}(0,1)}\left\{\lambda(t)\mathbb{E}_{x_0}\mathbb{E}_{x_t|x_0}\left[\|s_t(\hat{\theta}) - \nabla\log p_{t|0}(x_t|x_0)\|^2\right]\right\} \tag{5.97}$$

where $\nabla\log p_{t|0}$ can be readily obtained based on (5.2).

5.E.3 Diffusion Bridges for Topological Signals

As discussed earlier, SB models are closely related to stochastic interpolants, flow- and score-based models. We further remark that from the $\mathcal{T}$SDE, we can build the *topological diffusion bridge* to directly construct transport models between any topological distributions via *Doob's h-transform* [Särkkä & Solin, 2019]. This has been evidenced in Euclidean domains

5

by converting existing diffusion processes (BM, VE, VP) to diffusion bridges so to arrive at arbitrary distributions, and training upon score matching [Delbracio & Milanfar, 2023; Heng et al., 2021; Li et al., 2023; Liu et al., 2022; Zhou et al., 2024].

Specifically, consider the $\mathcal{T}$SDE. To let it arrive at a final sample x_1, the Doob's h-transform gives

$$\mathrm{d}X_t = \left[f_t + g_t^2 \nabla \log p_{1|t}(x_1|X_t)\right]\mathrm{d}t + g_t\,\mathrm{d}W_t, \quad X_1 = x_1,\ x_0 \sim \nu_0 \tag{5.98}$$

where $p_{1|t}(x_1|x_t)$ is the transition kernel of the $\mathcal{T}$SDE satisfying the associated backward FPK, given by Lemma 5.19 (cf. Lemma 5.4). We can further find the time-reversal process for (5.98) [Zhou et al., 2024, Theorem 1]

$$\mathrm{d}X_t = \left[f_t - g_t^2(\nabla \log q_{t|1}(x_t|x_1) - \nabla \log p_{1|t}(x_1|x_t))\right]\mathrm{d}t + g_t\,\mathrm{d}W_t, \quad X_1 = x_1 \tag{5.99}$$

where $q_{t|1}(x_t|x_1)$ is the transition kernel of the new SDE in (5.98) (instead of $\mathcal{T}$SDE) conditioned on $Y_1 = x_1$. The goal is to learn this new score function $\nabla \log q_{t|1}(x_t|x_1)$, which can be achieved by applying the score matching [Song et al., 2020b].

5

Given paired training samples $(x_0, x_1) \sim q_{01}$, Zhou et al. [2024, Theorem 2] considered a score matching objective to learn the new score function

$$\hat{\theta} = \arg\min \mathbb{E}_{x_t, x_0, x_1, t}\left[\lambda(t)\|\tilde{s}_t(\hat{\theta}) - \nabla \log q_{t|1}(x_t|x_1)\|^2\right]. \tag{5.100}$$

However, we cannot directly find closed-form $q_{t|1}$ in the topological case, we need to use sliced score matching for this case.

Note that we can view that the underlying topological process X now follows the new SDE pair (5.98) and (5.99) as the forward and backward processes, respectively. In this sense, the topological diffusion bridge is a special case of the $\mathcal{T}$SB when setting the policies as $Z_t = g_t \nabla \log p_{1|t}(x_1|x_t)$ and $\hat{Z}_t = g_t \nabla[\log q_{t|1}(x_t|x_1) - \nabla \log p_{1|t}(x_1|x_t)]$. When performing learning on these policies, since Z_t is fixed once $\mathcal{T}$SDE is given, the learning boils down to training the parameterized $\hat{Z}_t^{\hat{\theta}}$.

5.F Additional Experiments and Details

We first describe the synthetic experiment on matching Gaussian topological signal distributions based on the closed-form G$\mathcal{T}$SB. Then, we detail the generative modeling experiments conducted on real-world datasets based on $\mathcal{T}$SB-models.

5.F.1 Closed-form G$\mathcal{T}$SB corroboration

Graph GP matching: We build a synthetic graph with 30 nodes and 67 edges, as shown in Fig. 4.3 *(Left)*. From its graph Laplacian L, we construct the initial distribution of node signals as a Matérn GP with zero mean and the kernel $\Sigma_0 = (I + L)^{-1.5}$, and the final distribution as a diffusion GP with zero mean and the kernel $\Sigma_1 = \exp(-20L)$. We consider G$\mathcal{T}$SB closed forms X_t in (5.5) driven by both $\mathcal{T}\text{SHeat}_{\text{BM}}$ and $\mathcal{T}\text{SHeat}_{\text{VE}}$. For the former, we

set $c = 0.5$ and $g = 0.01$, labeled as GTSB-BM. For the latter, we consider $\sigma_{\min} = 0.01$ and $\sigma_{\max} = 1$ with $c = 0.01$ and $c = 10$, labeled as GTSB-VE1 and GTSB-VE2, respectively.

We then compute the covariances Σ_t of the time marginals, which has a closed-form given by Corollary 5.26, and obtain the samples based on the closed-form conditional distribution [cf. Corollary 5.30] given an initial sample, illustrated in Fig. 5.4. We also measure the Bures-Wasserstein (BW) distance [Bures, 1969] of Σ_t and Σ_1 to evaluate the bridge quality, shown in Section 5.6.

Edge GP matching: We also consider matching two edge GPs which are able to model the discretized edge flows in a SC_2 of the vector fields defined on a 2D plane [Chen et al., 2021c]. The initial edge GP has a zero mean and a divergence-free diffusion kernel with $\kappa_C = 10$, while the final edge GP has a zero mean and a curl-free diffusion kernel with $\kappa_G = 10$ [cf. Chapter 4]. We construct the closed-form G$\mathcal{T}$SB X_t with the $\mathcal{T}$SHeat$_{\mathrm{BM}}$ as the reference dynamics, where $c = 1$ and $g = 0.01$. We obtain the samples from the closed-form SDE representation (5.7), shown in Fig. 5.5. We can see that the forward samples are able to reach the final state, and the backward samples are able to reach the initial state, despite some noise due to numerical simulation.

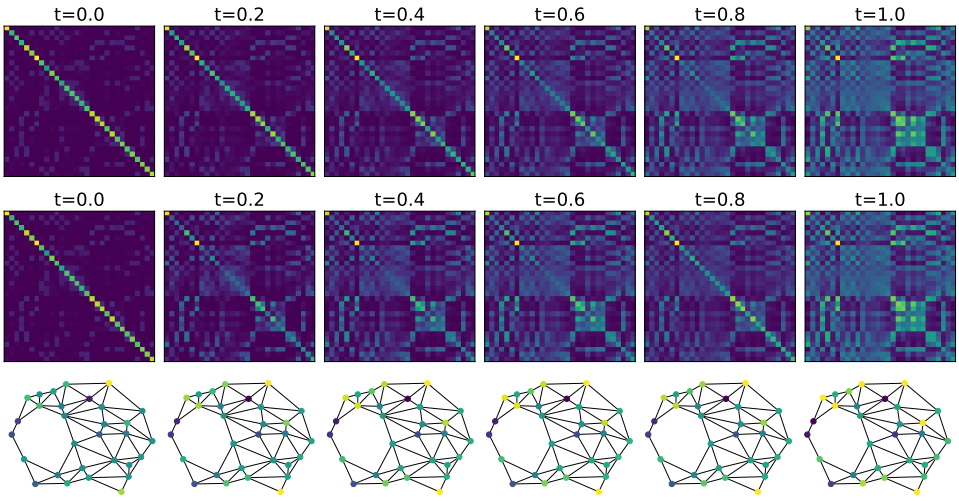


Figure 5.4: Covariances of the time marginals of the G$\mathcal{T}$SB driven by $\mathcal{T}$SHeat$_{\mathrm{BM}}$ *(Top)* and $\mathcal{T}$SHeat$_{\mathrm{VE}}$ *(Center)*, as well as the samples conditioned on the inisital signal *(Bottom)*.

5.F.2 $\mathcal{T}$SB-based Generative Modeling and Matching

Data

Heat flows: We use the heatflow dataset from Southeastern Australia from Mather et al. [2018], which collects the heatflow measurements with coordinates in total 294 from 1982 to 2016. Here we split the data into two parts, before and after 2010 (there is a significant change in the heat flow pattern), to understand the evolution of the heat flow by modeling

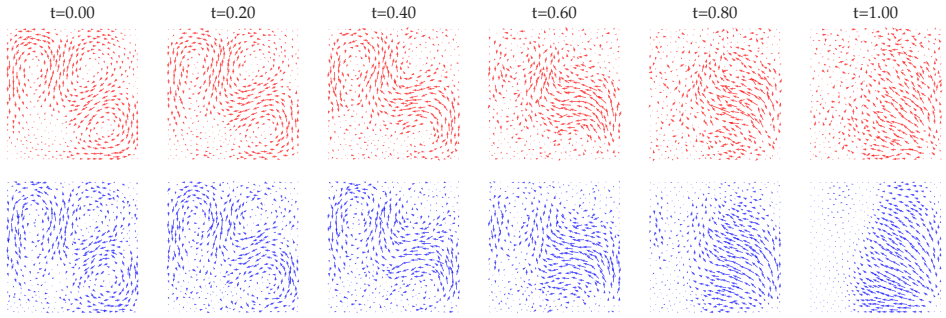


Figure 5.5: Forward and backward samples based on the closed-form SDE in (5.7) with respect to $\mathcal{T}\text{SHeat}_{\text{BM}}$.

them as initial and terminal data. That is, for this dataset, we consider the *signal matching* task.

Seismic magnitudes: We use the seismic event catalogue for M5.5+ (from 1990 to 2018) from IRIS which consists of 12,940 recorded earthquake events with magnitudes greater than 5.5. To process these events, we use the `stripy` toolbox to obtain the *icosahedral* triangulated mesh of the Earth surface [Moresi & Mather, 2019]. Using the refinement of level three, this spherical mesh has 1,922 vertices. We refer to Fig. 5.6 for a visualization of such a mesh of level one for better clarity.

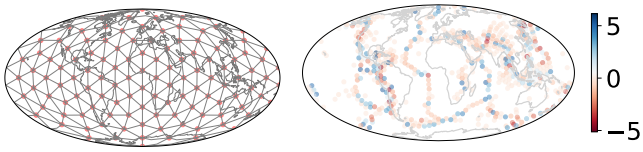


Figure 5.6: Earth mesh *(Left)* and a node signal sample of the earthquake magnitudes *(Right)*.

Upon this mesh, we first associate each earthquake event to the nearest mesh vertex based on its longitude and latitude. All events are located on 576 unique vertices of the mesh. Using these unique vertices, we then construct a *10-nearest neighbour* graph based on the geodesic distance between the vertices, and we use the symmetric normalized graph Laplacian. Lastly, on top of this graph, we associate the yearly earthquake events to its vertices and take the magnitudes as node signals, resulting in 29 such signals. Followed by this, we preprocess the magnitudes by removing the mean over the years. For this dataset, we consider the *signal generation* task.

Traffic flows: We consider the PeMSD4 dataset which contains traffic flow in California from 01-01-2018 to 28-02-2018 over 307 sensors. We convert the node data into edge flows over a SC_2 with 307 nodes, 340 edges and 29 triangles, following Chen et al. [2022c], and use the normalized Hodge Laplacian. For this dataset, we consider the *signal generation* task.

Ocean currents: We consider the *Global Lagrangian Drifter Data*, which was collected by NOAA Atlantic Oceanographic and Meteorological Laboratory. The dataset itself is a 3D

point cloud after converting the locations of buoys to the *earth-centered, earth-fixed* (ECEF) coordinate system. We follow the procedure in Chen & Meila [2021]; Chen et al. [2021c] to first sample 1,500 buoys furthest from each other, then construct a weighted SC_2 as a Vietoris-Rips (VR) complex with around 20k edges and around 90k triangles. For this dataset, we consider the *signal matching* task Upon the weighted Hodge Laplacian, we use edge GP learned by Yang et al. [2024] from the drifter collected data as the initial distribution and synthetize a curl-free edge GP as the final distribution. These two GPs have rather different behaviors, able to model ocean currents with different behaviors and make the matching task challenging. From these GPs, we can generate the samples for training and testing in an efficient way based on eigenpairs associated to the 500 largest eigenvalues, analogous to using Karhunen-Loéve type-decomposition for continuous GPs.

Brain fMRI signals: We consider the Human Connectome Project (HCP) [Van Essen et al., 2013] Young Adult dataset where we model the human brain network as a graph and perform the *matching* task on the measured fMRI signals recorded when the subject performed different tasks. We use the HCP recommended brain atlas [Glasser et al., 2016] where each hemisphere is divided into 180 cortical parcels. This results in a total of 360 brain regions. We then build a graph based on the physical conenction patterns between these regions where the edge weights measure the strength of the axonal connections between two regions, i.e., proportional to the inverse of the square distance [Perinelli et al., 2019]. We use the symmetric normalized graph Laplacian. In our experiments, the two sets of the fMRI signals, respectively, correspond to the liberal and aligned brain activities. The former is associated with brain regions involved in high-level cognition, like decision making and memory, whereas the latter is associated with the sensory regions, like visual and auditory, meaning that functional signals are aligned with anatomical brain structure, as shown in Fig. 5.7.

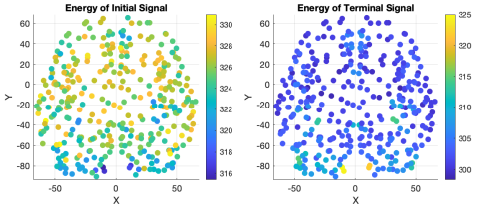


Figure 5.7: The energies of the initial (liberal) *(Left)* and final (aligned) brain signals *(Right)*.

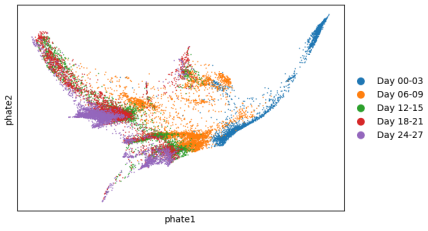


Figure 5.8: Two-dim `phate` embedding of the single-cell data [Moon et al., 2019].

Single-cell data: We consider the single-cell embryoid body data from [Moon et al., 2019], which describes the differentiation of human embryonic stem cells grown as embryoid bodies into diverse cell lineages over a period of 27 days. These cell data, $X_1, X_2, \ldots, X_5$, are collected at 5 timepoints (day 0–3, day 6–9, day 12–15, day 18–21, day 24–27, indexed by $t \in \{1,2,3,4,5\}$), resulting in total 18,203 observations. We followed the preprocessing steps provided by `TorchCFM` [Tong et al., 2024a;b]. Please refer to this link for the direct use of preprocessed data [Tong, 2023]. Followed by this, we consider the two-dimensional `phate` embedding for the data [Moon et al., 2019], resulting in the data coordinates of

dimension $18,203 \times 2$, as illustrated in Fig. 5.8. From the preprocessing, we can build a sparse k-nearest neighbouring graph over *the entire set of data observations*. That is, we have an adjacency matrix of dimension $18,203 \times 18,203$.

In our experiment, we aim to transport the observed data from day 0–3 to day 24–27, i.e., from $t = 1$ to $t = 5$. Thus, we build the two boundary distributions based on the normalized indicator functions, which indicate the associated nodes of the data points observed at these two timepoints. That is,

$$\nu_0 := \mathbf{1}_{X_1} / \textstyle\sum_{j \in X_1} \mathbf{1}_{X_1}(j) \tag{5.101}$$

where the sum is over the nodes associated to the first-timepoint observations in X_1, as the initial distribution, and similarly, $\nu_1 := \mathbf{1}_{X_5} / \sum_{j \in X_5} \mathbf{1}_{X_5}(j)$ as the final one. After training the models, using the final sample $\hat{X}_{t=5}$ obtained from the learned $\mathcal{T}$SB given the initial observations, we can obtain the predictions at the five timepoints based on the sorting (from large to small) of $\hat{X}_{t=5}$. Specifically, given the indices after sorting, $\mathrm{idx} = \arg\mathrm{sort}(\hat{X}_{t=5})$, we partition them into the disjoint sets, $\mathrm{idx} = \mathcal{S}_1 \cup \mathcal{S}_2 \cup \cdots \cup \mathcal{S}_5$ with $|\mathcal{S}_t| = n_t$ the number of observations at timepoint t for $t = 1, \ldots, 5$. We then have the prediction labels given by $\mathcal{S}_t$ that indicate the nodes supporting the data points predicted at timepoint t. The disjointed indices in $\mathcal{S}_t$ essentially provide a labeling of the whole predictions for the five timepoints. We found that using adjacency matrix as the convolution operator in the reference dynamics performs better in practice.

5

Model

Models. We consider the following two sets of methods:

- Euclidean SB-based models with BM, VE and VP reference processes [Chen et al., 2022b], which we refer to as SB-BM, SB-VE and SB-VP, respectively.
- Topological SB-based model with $\mathcal{T}$SHeat$_{\mathrm{BM}}$, $\mathcal{T}$SHeat$_{\mathrm{VE}}$ and $\mathcal{T}$SHeat$_{\mathrm{VP}}$ as the reference processes, which we refer to as TSB-BM, TSB-VE and TSB-VP, respectively.

For some datasets, we also apply the Gaussian SB SDE solution as the reference dynamics: Euclidean SB-based models with the closed-form GSB SDEs (under BM and VE reference processes) as the reference [Bunne et al., 2023], which we refer to as GSB-BM and GSB-VE, respectively; and, topological SB-based model with the closed-form G$\mathcal{T}$SB SDEs (5.7) (under $\mathcal{T}$SHeat$_{\mathrm{BM}}$ and $\mathcal{T}$SHeat$_{\mathrm{VE}}$) as the reference, which we refer to as GTSB-BM and GTSB-VE, respectively.

Improving reference dynamics. Our proposed three types of reference dynamics have fixed diffusion rates c. This may limit the model flexibility in capturing the dynamics of the data, which we found especially in matching ocean current data. Thus, for this task, we allow the time-varying diffusion rate c_t. Specifically, we set it to be linearly increasing as $c_t = c_{\min} + t(c_{\max} - c_{\min})$ for some $c_{\min}, c_{\max}$. Moreover, due to the nonlinearity of the underlying process (from a non-curl-free GP to a curl-free GP), we also consider the heterogeneous heat diffusion, as dicussed in (5.31) where the down and up diffusion rates are different.

Policy models. For the parameterization of the optimal policies $(Z_t^\theta, \hat{Z}_t^{\hat{\theta}})$, we first obtain the time and signal embeddings individually. To obtain the signal embedding from the input, we consider the following two sets of models as the signal module:

- ResBlock model: one multi-layer perceptron (MLP) followed by a number of residual block modules where each block has three MLPs with sigmoid linear unit (SiLU) activations.
- Topological neural network (TNN) model: For node signals, we consider two-layer GCNs [Kipf & Welling, 2017] followed by one MLP; For edge flows in a SC_2, we consider two-layer SNNs [Roddenberry et al., 2021] followed by one MLP, where each SNN layer has the linear convolution $X \leftarrow L_u X W_2 + X W_1 + L_d X W_0$ with the down and up Laplacians L_d, L_u and the learnable weights W_0, W_1, W_2.

To obtain the time embedding, we pass the *sinusoidal* positional encoding of the discretized timepoint through a two-layer MLP module with SiLU activations. We then sum the two embeddings and pass it through a two-layer MLP output module with SiLU to obtain the final parameterization.

Implementation Details

Our implementation is built upon the SB-framework by Chen et al. [2022b]. We use `AdamW` optimizer with a learning rate of 10^{-4} and Exponential Moving Average (EMA) with the decay rate of 0.99. For the reference processes with BM involved, we treat the noise scale g as a hyperparameter and optimize it by grid search. For the reference processes with VE and VP involved, we grid search the noise scales $\sigma_{\min}, \sigma_{\max}$ and $\beta_{\min}, \beta_{\max}$. For $\mathcal{T}$SB-based models, we grid search the optimal diffusion rate c and the noise scales involved in the $\mathcal{T}$SHeat.

In computing the likelihood in (5.9) during training, we use the trace estimator following [Hutchinson, 1989] to compute the divergence. In generative procedures, we apply the *predictor-corrector* sampling [Song et al., 2020b] to improve performance. To evaluate the models, we compute the *negative log-likelihoods* (NLLs) in (5.91) for generation tasks. For both generation and matching tasks, we assess the 1- and (square rooted) 2-*Wasserstein distances* between the predicted and true signals.

Results

Heat flows: For matching the two types of heat flows, we observe from Table 5.2 that: (i) $\mathcal{T}$SB-based models are consistently better than SB-based models; and (ii) using GCNs for policy models increases the performance by a large margin for both sets of models.

Seismic magnitudes: In generative modeling for seismic magnitudes, while we have the similar observations as for the previous datasets, from Table 5.4, we also observe that the G$\mathcal{T}$SB-based models are able to achieve the best performance, and likewise GSB-based models also increase the performance of SB-models. In Table 5.6, we report the 1- and 2-Wasserstein distances between the generated samples and the true ones.

Traffic flows: In geneartive modeling of traffic flows, we observe from Table 5.3 that: $\mathcal{T}$SB-based models achieve smaller NLLs and using SNNs for both models improves the

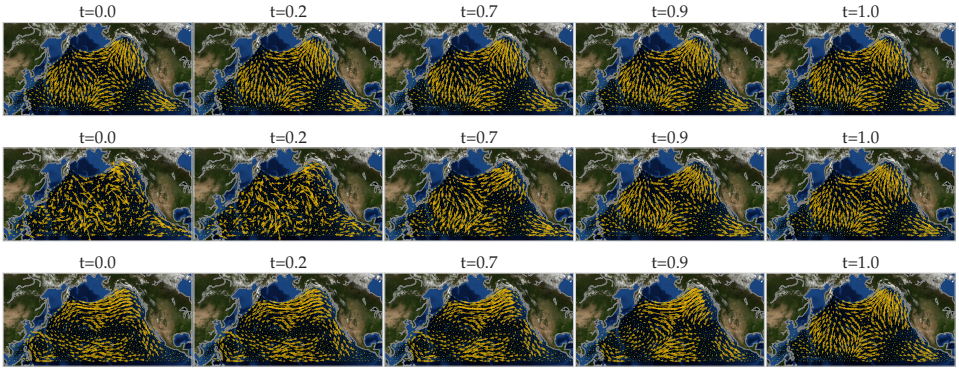


Figure 5.9: Backward sampled ocean currents using TSB-BM *(Top)*, SB-BM *(Center)* and GTSB-BM *(Bottom)*.

Table 5.2: Heat flow matching results.

Method	ResBlock		GCN	
	Forward	Backward	Forward	Backward
SB-BM	-0.10 ± 0.02	-0.08 ± 0.03	-0.74 ± 0.05	-0.72 ± 0.05
SB-VE	-0.12 ± 0.02	-0.10 ± 0.01	-1.20 ± 0.07	-0.95 ± 0.07
SB-VP	-0.09 ± 0.02	-0.08 ± 0.02	-0.83 ± 0.04	-0.66 ± 0.11
TSB-BM	-0.29 ± 0.02	-0.27 ± 0.02	-0.83 ± 0.05	-0.81 ± 0.05
TSB-VE	-0.31 ± 0.02	-0.29 ± 0.01	-1.26 ± 0.05	-0.97 ± 0.08
TSB-VP	-0.57 ± 0.02	-0.55 ± 0.02	-1.01 ± 0.03	-0.92 ± 0.03

Table 5.3: Traffic flow results.

Method	ResBlock	SNN
SB-BM	0.82 ± 0.00	0.18 ± 0.02
SB-VE	0.77 ± 0.00	-0.42 ± 0.01
SB-VP	0.79 ± 0.00	-0.09 ± 0.01
TSB-BM	0.40 ± 0.00	0.02 ± 0.03
TSB-VE	0.01 ± 0.00	-0.89 ± 0.02
TSB-VP	0.02 ± 0.00	-0.32 ± 0.01

Table 5.4: Seismic magnitudes results.

Method	ResBlock	GCN
SB-BM	2.78 ± 0.01	2.71 ± 0.03
SB-VE	2.97 ± 0.03	2.73 ± 0.05
SB-VP	2.28 ± 0.02	2.01 ± 0.03
GSB-BM	1.86 ± 0.02	1.83 ± 0.05
GSB-VE	1.68 ± 0.03	1.46 ± 0.07
TSB-BM	2.13 ± 0.01	1.82 ± 0.02
TSB-VE	2.22 ± 0.02	1.53 ± 0.03
TSB-VP	2.00 ± 0.02	1.51 ± 0.02
GTSB-BM	1.58 ± 0.01	1.43 ± 0.04
GTSB-VE	1.49 ± 0.02	1.06 ± 0.04

Table 5.5: Ocean current matching results.

Method	Foward	Backward
SB-BM	7.21 ± 0.00	7.21 ± 0.00
SB-VE	7.17 ± 0.02	7.17 ± 0.02
GSB-BM	1.09 ± 0.01	0.97 ± 0.00
GSB-VE	0.83 ± 0.01	0.49 ± 0.00
TSB-BM	6.94 ± 0.01	3.70 ± 0.00
TSB-VE	6.89 ± 0.00	3.60 ± 0.00
GTSB-BM	1.09 ± 0.01	0.97 ± 0.00
GTSB-VE	0.53 ± 0.00	0.47 ± 0.00

performance. This observation is consistent with the Wasserstein metrics reported in Table 5.6.

Ocean currents: In matching ocean currents with two types of different-behaving edge GPs, *note that the initial sample in the forward process in Fig. 5.3 and the final sample in the backward process in Fig. 5.9 are the true samples.* From these two figures, we first observe that SB-based models fail to learn the dynamics to reach the expected end states, as shown in Figs. 5.3 and 5.9. On the other hand, $\mathcal{T}$SB-based models are able to reach an end state with small curl component in the forward process, yet with some discrepancy from the true one.

Table 5.6: Overall 1- and (square rooted) 2-Wasserstein distances for generating and matching.

Method	Seismic magnitudes		Traffic flows		Brain signals		Single-cell data	
	W_1	W_2	W_1	W_2	W_1	W_2	W_1	W_2
SB−BM	$11.73_{\pm 0.05}$	$8.29_{\pm 0.04}$	$18.69_{\pm 0.02}$	$13.36_{\pm 0.01}$	$12.08_{\pm 0.08}$	$8.58_{\pm 0.05}$	$0.33_{\pm 0.01}$	$0.40_{\pm 0.01}$
SB−VE	$11.49_{\pm 0.04}$	$8.13_{\pm 0.03}$	$19.04_{\pm 0.02}$	$13.61_{\pm 0.02}$	$17.46_{\pm 0.14}$	$12.42_{\pm 0.09}$	$0.33_{\pm 0.01}$	$0.39_{\pm 0.01}$
SB−VP	$12.61_{\pm 0.06}$	$8.92_{\pm 0.04}$	$18.22_{\pm 0.03}$	$13.02_{\pm 0.02}$	$13.41_{\pm 0.05}$	$9.54_{\pm 0.04}$	$0.33_{\pm 0.01}$	$0.40_{\pm 0.00}$
TSB−BM	$9.01_{\pm 0.03}$	$6.37_{\pm 0.03}$	$10.57_{\pm 0.02}$	$7.62_{\pm 0.01}$	$7.51_{\pm 0.08}$	$5.51_{\pm 0.06}$	$0.14_{\pm 0.03}$	$0.28_{\pm 0.05}$
TSB−VE	$7.69_{\pm 0.04}$	$5.44_{\pm 0.03}$	$10.51_{\pm 0.02}$	$7.58_{\pm 0.01}$	$7.59_{\pm 0.05}$	$5.55_{\pm 0.04}$	$0.14_{\pm 0.02}$	$0.27_{\pm 0.04}$
TSB−VP	$8.40_{\pm 0.04}$	$5.95_{\pm 0.03}$	$9.92_{\pm 0.02}$	$7.16_{\pm 0.01}$	$7.67_{\pm 0.11}$	$5.64_{\pm 0.09}$	$0.14_{\pm 0.01}$	$0.22_{\pm 0.03}$

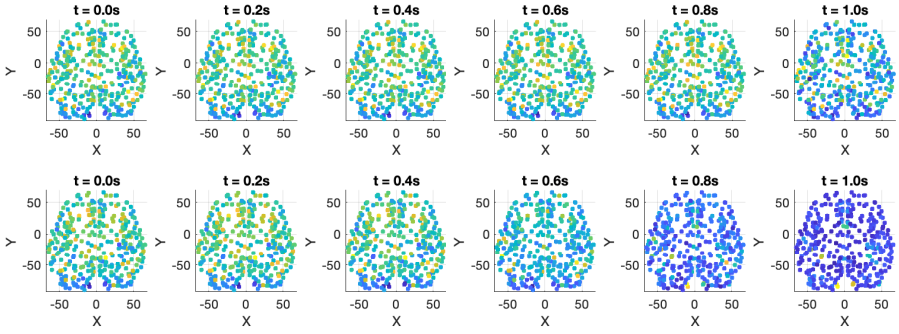


Figure 5.10: Intermediate samples of brain signals learned using SB−VE *(Top)* and TSB−VE *(Bottom)*.

Moreover, the learned backward dynamics remains noisy and does not completely return to the initial state. This implies that the underlying dynamics cannot be fully captured by the $\mathcal{T}$SHeat-type reference processes. This can be largely alleviated by using G$\mathcal{T}$SB-based models, where both the forward and backward processes reach the expected states with high fidelity. This is however because we have the initial and final distributions modeled by GPs, allowing for G$\mathcal{T}$SB to capture the underlying dynamics better. These observations are reflected in the square-rooted 2-Wasserstein distance results in Table 5.5 between the samples given by the learned forward process and the true ones, as well as for the backward ones.

Brain fMRI signals: In matching the two brain fMRI signals, we observe from Fig. 5.10 that TSB−VE-based model reaches the final state where the signals have lower energy over the brain, indicating aligned activities, whereas SB−VE fails to do so. This is quantatively reflected in terms of the Wasserstein metrics between the generated final samples and the groundtruth ones in Table 5.6.

Table 5.8: Intermediate prediction performance on single-cell data using TSB-BM and SB-BM.

Timepoint	TSB-BM			SB-BM			
	W_1	W_2	accuracy	W_1	W_2	accuracy	leave-one-out
2	$0.03_{\pm 0.00}$	$0.09_{\pm 0.00}$	0.80	$0.52_{\pm 0.01}$	$0.59_{\pm 0.01}$	0.28	0.24
3	$0.09_{\pm 0.00}$	$0.22_{\pm 0.01}$	0.42	$0.12_{\pm 0.00}$	$0.21_{\pm 0.00}$	0.23	0.20
4	$0.08_{\pm 0.00}$	$0.16_{\pm 0.01}$	0.45	$0.19_{\pm 0.00}$	$0.34_{\pm 0.00}$	0.26	0.26
5	$0.14_{\pm 0.03}$	$0.28_{\pm 0.05}$	0.70	$0.33_{\pm 0.01}$	$0.40_{\pm 0.01}$	0.24	1

Ablation study on graph normalizations. Here, we compare the performance of the TSB-based models using different ways of graph Laplacian normalizations. Specifically, we consider the random walk L_{RW} and the combinatorial L_{comb} graph Laplacians. For the latter, we normalize it by dividing the maximal eigenvalue of the Laplacian for stability. From Table 5.7, we notice that using the random walk has comparable performance with using the symmetric normalized Laplacian $L_{\text{norm}}^{\text{sym}}$, and using the combinatorial one is worse than the other two. This is not surprising since the combinatorial one does not encode the connection strength between brain regions.

Table 5.7: Ablation study results on graph normalizations for brain signal matching.

Method	TSB-BM	TSB-VE	TSB-VP
$W_1, L_{\text{norm}}^{\text{sym}}$	7.51 ± 0.08	7.59 ± 0.05	7.67 ± 0.11
$W_2, L_{\text{norm}}^{\text{sym}}$	5.51 ± 0.06	5.55 ± 0.04	5.64 ± 0.09
W_1, L_{RW}	7.51 ± 0.08	7.62 ± 0.09	7.65 ± 0.09
W_2, L_{RW}	5.52 ± 0.06	5.58 ± 0.07	5.62 ± 0.06
W_1, L_{comb}	8.06 ± 0.05	9.21 ± 0.06	9.29 ± 0.05
W_2, L_{comb}	5.80 ± 0.04	6.62 ± 0.05	6.73 ± 0.03

Single-cell data: We first measure the Wasserstein distances between the predicted single-cells and the groundtruth ones at the final timepoint, as reported in Table 5.6. We here provide the predictions in the two-dim `phate` embedding space for the SB-BM and TSB-BM models in Fig. 5.2 and the latter has a much better prediction. Moreover, from the final sample, we evaluate the predictions at the intermediate timepoints (see Appendix 5.F.2) in Table 5.8 where the performance of TSB-BM is consistently better than SB-BM. Since our method relies on a graph constructed from the entire data points, we also provide the leave-one-out accuracy for SB-BM by training on the entire data points leaving out the to-be-predicted timepoint. We see that while the accuracy for the final timepoint is perfect, the intermediate predictions remain poor. In contrast, TSB-BM, by making use of the topology, captures the underlying dynamics and predicts the intermediate states better.

Computational Complexity

Compared to SB-based models, the $\mathcal{T}$SB-based models introduce an additional topological convolution [cf. (5.1)] overhead, which however admits an efficient computation, as discussed in Section 5.5. We here provide a quantative comparison of the compelxity in terms of the training time and memory consumption. We measure them using SB-VE and TSB-VE models on different-sized *10-nearest neighbour* graphs built from Swiss roll point clouds. This comparison is done in a single training stage with 2,000 iterations, running on a single NVIDIA RTX 3080 GPU. As shown in Fig. 5.11, we observe that in the moderate scale ($\leq 10,000$) region, the training time and memory consumption of

TSB-VE are only slightly higher than SB-VE, with negligible difference. While this overhead becomes more significant as the scale further increases, both training time and memory can be reduced by exploiting the *sparse* structure (here implemented using `torch.tensor.to_sparse`) in the graph topology such that the computational overheads for for SB and TSB models remain comparable. Under the same settings, Table 5.9 compares SB-BM and TSB-BM models across all datasets. The additional memory and training time introduced by TSB-BM remain below 4%.

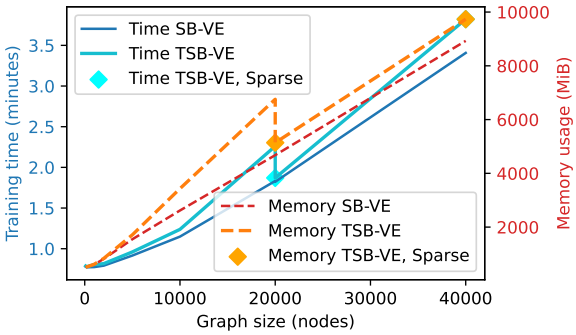


Figure 5.11: Training time and memory comparison when training SB-VE and TSB-VE w.r.t. different-sized graphs.

Dataset	TSB-BM	SB-BM
Seismic	516 50.17	512 51.48
Traffic	510 52.25	504 50.62
Ocean	5976 106.68	5892 102.67
Brain	486 49.62	468 48.97
Single-cell	4446 94.30	4294 92.54

Table 5.9: Complexity (first row: memory (in MiB), and second row: training time (in seconds)).

6

Conclusion

Depending on the data type and the involved domain, machine learning is often divided into sub-fields or sub-topics. For instance, computer vision focuses on images and videos; natural language processing focuses on texts; and graph machine learning puts its focus on data involved with graphs. The different nature of the data and the domains results in that the specific techniques for processing and learning in these sub-fields differ to some extent. For example, convolutions in image processing are operated within a regular grid of pixels, different from graph convolutions which operate within local neighborhoods in a graph. Also, the construction and use of kernels for images are different from graph kernels. While the definition of distances is rather straightforward in a grid, it remains unclear in a graph. However, in some way, the underlying principles to develop these techniques often share the same core. That is, in each sub-field, most of the techniques attempt to leverage the local relationships between data points to aid learning. This is often done by defining a *shift* operation, which shifts the data points in the domain to their local neighborhoods, and a *sum* operation, which aggregates the shifted data points. This offers interpretability to some extent and reduced complexity, and explores the invariants of the domain. For instance, convolution in image processing can be unified as a special graph convolution on grid graphs, because they are fundamentally a *shift-and-sum* operation, despite that the shift and sum operations might differ per domain. Also, kernel is a similarity measure between two points in the domain, though the definition of similarity varies per domain.

In this thesis, we have presented our attempt towards a systematic study of machine learning on simplicial complexes with signal processing principles such as convolutions and spectral analysis. Although this establishes a new sub-field, our methodologies are developed using the same train of thought nonetheless — respecting the underlying principles while taking into account the specific characteristics of the signals and the nature of the domain. Specifically, we bear in mind the Hodge decomposition of simplicial signals and the discrete nature of simplicial complexes. In other words, exploiting these intrinsic properties of signals and the domain allows us to incorporate *priors* or *inductive biases* to the processing and learning, which in turn leads to better efficiency, interpretability and flexibility. The tools we have developed make up a principled learning framework on simplicial complexes.

The simplicial convolutional filters in Chapter 2 perform efficient processing of simplicial signals within local simplicial neighborhoods, respect the convolution theorem, and allow for individual filtering of the signals in the different Hodge subspaces. The simplicial complex CNNs in Chapter 3 allow for efficient learning on simplicial complexes, and carry with interpretability and flexibility, as well as being stable to domain perturbations. Beyond the deterministic approaches, the Hodge-compositional simplicial Gaussian processes in Chapter 4 offer Gaussian modeling of simplicial signals, which is simple, principled, and interpretable, allowing for individual modeling in the different Hodge subspaces. These tools are well-suited for modeling, processing and learning of simplicial signals, and enjoy interpretability owing to their ability to account for the Hodge decomposition and the practical properties of the signals, e.g., beging divergence-free or curl-free. Finally, the topological Schrödinger bridge matching in Chapter 5 provides a framework for transporting one simplicial signal distribution to another and offers a principled way to generative modeling of simplicial signals. This work makes use of the three tools we have developed in the previous chapters, reflecting their fundamental roles in learning on simplicial complexes.

Future Work

While our work has presented a few fundamental tools for simplicial signals, much work remains unexplored in addition to the future work we have presented after each chapter. First, these tools can be, respectively, improved and extended, in analogy to how graph signal processing and machine learning tools advance over the years. In fact, some of the recent works have made efforts in this direction. For instance, simplicial filters have been accompanied with autoregressive time series filters for processing simplicial-time signals [Krishnan et al., 2024]. A number of more powerful simplicial neural network architectures have been developed, analogous to the advance of graph neural networks, as summarized in Besta et al. [2024]. While this is a promising general direction, we highlighted the importance of exploiting the Hodge decomposition of simplicial signals.

Second, as we have discussed in Chapter 4, simplicial signals (cochains) can be viewed as discrete analogues of *differential forms* on manifolds — informally, graph signals are analogous to scalar fields and edge flows to vector fields; and both graph Laplacians and Hodge Laplacians have their continuous counterparts. This discrete-continuous analogy has inspired us in developing some of the tools in this thesis. For example, when modeling the Hodge-compositional edge Gaussian processes, we build the kernels based on several differential equations which have their analogy in continuous domain. This train of thought can be futher exploited — by taking inspiration from time domain processing, different dynamics can be constructed on simplicial complexes. We have made attempts on this in Chapter 5 where we studied stochastic heat diffusions on simplicial complexes. In analogy to Nikitin et al. [2022] for the graph case, this allows for nonseparable temporal-simplicial kernels to model temporal-simplicial Gaussian processes.

Third, our current research on dealing with simplicial signals of different orders remains limited. In a 2-simplicial complex, when we have a system of node signals, edge flows, and triangle flows, our filtering and learning tools are limited to the basic linear interactions between them via the incidence matrices (first-order derivatives). Having the continuous

analogy in mind, we expect more complex interactions between these signals especially when there is a time factor involved. For example, Giambagli et al. [2022]; Nijholt & DeVille [2022]; Nurisso et al. [2024]; Ziegler et al. [2022], among others, constructed a system of differential equations on simplicial complexes to model the interactions between node signals, edge flows, and triangle flows, resembling the continuous diffusion, convection dynamics and their nonlinear variants, like Kuramoto dynamics. Machine learning models based on these dynamics have not been well studied yet, though a number of works have constructed GNNs inspired by complex dynamics on networks such as Chamberlain et al. [2021a;b]; Chen et al. [2022a]; Eliasof et al. [2021]; Rusch et al. [2022]. They may be helpful to model more complex interactions between simplicial signals across different orders in many real-worl physical systems involved with scalar fields, vector fields and other quantities, such as electromagnetic waves [Tarhasaari et al., 1999; Teixeira, 2001] and shallow water flow [Lee et al., 2018]. This in turn raises the question of modeling more real-world systems in simplicial complexes, and brings us to the next point.

Finally, future work can be explored application-wise. Notably, one can model electric power systems and circuits as graphs or simplicial 2-complexes where voltages and currents are edge flows. In these systems, the edge flows can be complex-valued due to the presence of reactive power, which poses the imporance of performing complex-valued simplicial signal learning. Also, most of the real-world physical applications in our work involve edge flows, which are limited to 1-simplicial signals. Potential applications involving simplicial signals of different orders and of higher-orders can be explored, which we foresee in the context of electric networks, biological (e.g., brain, population) networks, social networks. As a limitation, the experimental implementations in this work are more in the *exploratory* phase, as a proof of concept of our tools, which however allows domain expertise to bring them closer to practice. We also look forward to exploring the potential of our work in more forward physics and engineering problems that remain to be addressed. We note that Kerimov et al. [2024; 2025] have made progress in applying some of the tools developed in this work to water supply networks.

Overall, this thesis provides foundational advances in the intersection of topology, signal processing, statistics and dynamics, with the goal of contributing to the field of topological machine learning (including neural networks, statistical learning and generative modeling). Due to this interdisciplinary nature, we believe that our work can be further developed in both the signal processing and machine learning communities, as well as in the network science, physics and engineering communities. We hope that our work can inspire more researchers to explore the potential of simplicial machine learning in various applications, and to further the tools presented in this thesis.

Bibliography

Bibliography

Mathieu Alain, So Takao, Brooks Paige, and Marc Peter Deisenroth. Gaussian Processes on Cellular Complexes. In *Proceedings of the 41st International Conference on Machine Learning*, volume 235, pp. 879–905. PMLR, 2024. Cited on pages 107, 131, and 151.

Michael Samuel Albergo, Mark Goldstein, Nicholas Matthew Boffi, Rajesh Ranganath, and Eric Vanden-Eijnden. Stochastic Interpolants with Data-Dependent Couplings. In *Forty-first International Conference on Machine Learning*, 2024. Cited on pages 128, 135, and 159.

Luigi Ambrosio, Nicola Gigli, and Giuseppe Savaré. *Gradient flows: in metric spaces and in the space of probability measures*. Springer Science & Business Media, 2008. Cited on page 153.

Waleed Ammar, Dirk Groeneveld, Chandra Bhagavatula, Iz Beltagy, Miles Crawford, Doug Downey, Jason Dunkelberger, Ahmed Elgohary, Sergey Feldman, Vu Ha, et al. Construction of the literature graph in semantic scholar. *arXiv preprint arXiv:1805.02262*, 2018. Cited on pages 66 and 87.

D Vijay Anand, Soumya Das, and Moo K Chung. Hodge-Decomposition of Brain Networks. *arXiv preprint arXiv:2211.10542*, 2022. Cited on pages 50 and 72.

Brian D.O. Anderson. Reverse-time diffusion equation models. *Stochastic Processes and their Applications*, 12(3):313–326, May 1982. Cited on page 152.

Panos J Antsaklis and Anthony N Michel. *Linear systems*, volume 8. Springer, 1997. Cited on page 145.

Cedric Archambeau, Dan Cornford, Manfred Opper, and John Shawe-Taylor. Gaussian process approximations of stochastic differential equations. In *Gaussian Processes in Practice*, pp. 1–16. PMLR, 2007. Cited on page 145.

Jacob Austin, Daniel D Johnson, Jonathan Ho, Daniel Tarlow, and Rianne Van Den Berg. Structured denoising diffusion models in discrete state-spaces. *Advances in Neural Information Processing Systems*, 34:17981–17993, 2021. Cited on page 141.

Iskander Azangulov, Andrei Smolensky, Alexander Terenin, and Viacheslav Borovitskiy. Stationary Kernels and Gaussian Processes on Lie Groups and their Homogeneous Spaces I: the Compact Case. *arXiv preprint arXiv:2208.14960*, 2022. Cited on page 97.

Iskander Azangulov, Andrei Smolensky, Alexander Terenin, and Viacheslav Borovitskiy. Stationary Kernels and Gaussian Processes on Lie Groups and their Homogeneous Spaces II: non-compact symmetric spaces. *arXiv preprint arXiv:2301.13088*, 2023. Cited on page 97.

Sergio Barbarossa and Stefania Sardellitti. Topological signal processing over simplicial complexes. *IEEE Transactions on Signal Processing*, 68:2992–3007, 2020. Cited on pages 4, 8, 12, 13, 14, 15, 20, 21, 22, 25, 31, 32, 50, 51, 69, 98, and 130.

Sergio Barbarossa, Stefania Sardellitti, and Elena Ceci. Learning from signals defined over simplicial complexes. In *2018 IEEE Data Science Workshop (DSW)*, pp. 51–55. IEEE, 2018. Cited on page 4.

Claudio Battiloro, Lucia Testa, Lorenzo Giusti, Stefania Sardellitti, Paolo Di Lorenzo, and Sergio Barbarossa. Generalized simplicial attention neural networks. *arXiv preprint arXiv:2309.02138*, 2023. Cited on page 74.

Federico Battiston, Giulia Cencetti, Iacopo Iacopini, Vito Latora, Maxime Lucas, Alice Patania, Jean-Gabriel Young, and Giovanni Petri. Networks beyond pairwise interactions: structure and dynamics. *Physics Reports*, 874:1–92, 2020. Cited on page 49.

Jean-David Benamou and Yann Brenier. A computational fluid mechanics solution to the Monge-Kantorovich mass transfer problem. *Numerische Mathematik*, 84(3):375–393, January 2000. Cited on pages 130 and 145.

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. Cited on pages 49, 50, and 88.

Austin R Benson, David F Gleich, and Desmond J Higham. Higher-order network analysis takes off, fueled by classical ideas and new data. *arXiv preprint arXiv:2103.05031*, 2021. Cited on page 49.

Renato Berlinghieri, Brian L. Trippe, David R. Burt, Ryan Giordano, Kaushik Srinivasan, Tamay Özgökmen, Junfei Xia, and Tamara Broderick. Gaussian Processes at the Helm(Holtz): A More Fluid Model for Ocean Currents. In *Proceedings of the 40th International Conference on Machine Learning*, ICML'23. JMLR.org, 2023. Cited on page 107.

Maciej Besta, Florian Scheidl, Lukas Gianinazzi, Shachar Klaiman, Jürgen Müller, and Torsten Hoefler. Demystifying Higher-Order Graph Neural Networks. *arXiv preprint arXiv:2406.12841*, 2024. Cited on pages 9, 50, 69, and 178.

Arne Beurling. An automorphism of product measures. *Annals of Mathematics*, 72(1): 189–200, 1960. Cited on page 133.

Christian Bick, Elizabeth Gross, Heather A Harrington, and Michael T Schaub. What are higher-order networks? *SIAM Review*, 65(3):686–731, 2023. Cited on pages 12, 50, 128, and 130.

Alberto Bietti and Julien Mairal. Group invariance, stability to deformations, and complexity of deep convolutional representations. *arXiv preprint arXiv:1706.03078*, 2017. Cited on page 61.

Cristian Bodnar, Fabrizio Frasca, Nina Otter, Yu Guang Wang, Pietro Liò, Guido F Montufar, and Michael Bronstein. Weisfeiler and lehman go cellular: Cw networks. *Advances in Neural Information Processing Systems*, 34, 2021a. Cited on pages 70 and 75.

Cristian Bodnar, Fabrizio Frasca, Yuguang Wang, Nina Otter, Guido F Montufar, Pietro Lio, and Michael Bronstein. Weisfeiler and lehman go topological: Message passing simplicial networks. In *International Conference on Machine Learning*, pp. 1026–1037. PMLR, 2021b. Cited on pages 5, 19, 50, 51, 54, 59, 64, 69, 73, 75, 86, 88, and 91.

Cristian Bodnar, Francesco Di Giovanni, Benjamin Chamberlain, Pietro Liò, and Michael Bronstein. Neural sheaf diffusion: A topological perspective on heterophily and oversmoothing in gnns. *Advances in Neural Information Processing Systems*, 35:18527–18541, 2022. Cited on pages 70 and 88.

Raicho Bojilov and Alfred Galichon. Matching in closed-form: equilibrium, identification, and comparative statics. *Economic Theory*, 61(4):587–609, 2016. Cited on page 143.

Viacheslav Borovitskiy, Alexander Terenin, Peter Mostowsky, et al. Matérn Gaussian processes on Riemannian manifolds. *Advances in Neural Information Processing Systems*, 33:12426–12437, 2020. Cited on pages 97 and 98.

Viacheslav Borovitskiy, Iskander Azangulov, Alexander Terenin, Peter Mostowsky, Marc Deisenroth, and Nicolas Durrande. Matérn Gaussian processes on graphs. In *International Conference on Artificial Intelligence and Statistics*, pp. 2593–2601. PMLR, 2021. Cited on pages 97, 98, 99, 118, 126, and 131.

Viacheslav Borovitskiy, Mohammad Reza Karimi, Vignesh Ram Somnath, and Andreas Krause. Isotropic Gaussian Processes on Finite Spaces of Graphs, February 2023. Cited on page 98.

Mario Botsch, Leif Kobbelt, Mark Pauly, Pierre Alliez, and Bruno Lévy. *Polygon mesh processing*. CRC press, 2010. Cited on page 13.

Michael M Bronstein, Joan Bruna, Taco Cohen, and Petar Veličković. Geometric deep learning: Grids, groups, graphs, geodesics, and gauges. *arXiv preprint arXiv:2104.13478*, 2021. Cited on page 49.

Joan Bruna and Stéphane Mallat. Invariant scattering convolution networks. *IEEE transactions on pattern analysis and machine intelligence*, 35(8):1872–1886, 2013. Cited on page 61.

Joan Bruna, Wojciech Zaremba, Arthur Szlam, and Yann LeCun. Spectral networks and locally connected networks on graphs. *arXiv preprint arXiv:1312.6203*, 2013. Cited on page 3.

Eric Bunch, Qian You, Glenn Fung, and Vikas Singh. Simplicial 2-Complex Convolutional Neural Networks. In *TDA & Beyond*, 2020. Cited on pages 5, 50, 53, 55, 60, 64, 69, 73, 74, 75, 76, 86, 88, 92, and 93.

Charlotte Bunne, Laetitia Papaxanthos, Andreas Krause, and Marco Cuturi. Proximal optimal transport modeling of population dynamics. In *International Conference on Artificial Intelligence and Statistics*, pp. 6511–6528. PMLR, 2022. Cited on page 141.

Charlotte Bunne, Ya-Ping Hsieh, Marco Cuturi, and Andreas Krause. The Schrödinger Bridge between Gaussian Measures has a Closed Form. In *Proceedings of The 26th International Conference on Artificial Intelligence and Statistics*, pp. 5802–5833. PMLR, April 2023. Cited on pages 128, 134, 136, 137, 140, 141, 155, and 170.

Donald Bures. An extension of Kakutani's theorem on infinite product measures to the tensor product of semifinite w*-algebras. *Transactions of the American Mathematical Society*, 135:199–212, 1969. Cited on page 167.

Chen Cai and Yusu Wang. A note on over-smoothing for graph neural networks. *arXiv preprint arXiv:2006.13318*, 2020. Cited on page 55.

Kenneth F Caluya and Abhishek Halder. Wasserstein proximal algorithms for the Schrödinger bridge problem: Density control with nonlinear drift. *IEEE Transactions on Automatic Control*, 67(3):1163–1178, 2021. Cited on pages 133, 134, 135, 152, 153, and 163.

Andrew Campbell, Jason Yim, Regina Barzilay, Tom Rainforth, and Tommi Jaakkola. Generative Flows on Discrete State-Spaces: Enabling Multimodal Flows with Applications to Protein Co-Design, February 2024. Cited on page 141.

Ozan Candogan, Ishai Menache, Asuman Ozdaglar, and Pablo A Parrilo. Flows and decompositions of games: Harmonic and potential games. *Mathematics of Operations Research*, 36(3):474–503, 2011. Cited on pages 4, 12, 32, 50, and 98.

Benjamin Paul Chamberlain, James Rowbottom, Davide Eynard, Francesco Di Giovanni, Xiaowen Dong, and Michael M. Bronstein. Beltrami Flow and Neural Diffusion on Graphs, October 2021a. Cited on page 179.

Benjamin Paul Chamberlain, James Rowbottom, Maria Gorinova, Stefan Webb, Emanuele Rossi, and Michael M. Bronstein. GRAND: Graph Neural Diffusion, September 2021b. Cited on page 179.

Qi Chen, Yifei Wang, Yisen Wang, Jiansheng Yang, and Zhouchen Lin. Optimization-Induced Graph Implicit Nonlinear Diffusion. In *Proceedings of the 39th International Conference on Machine Learning*, pp. 3648–3661. PMLR, June 2022a. Cited on page 179.

Ricky TQ Chen, Yulia Rubanova, Jesse Bettencourt, and David K Duvenaud. Neural ordinary differential equations. *Advances in neural information processing systems*, 31, 2018. Cited on page 137.

Tianrong Chen, Guan-Horng Liu, and Evangelos A Theodorou. Likelihood training of Schrödinger bridge using forward-backward sdes theory. *International Conference on Learning Representation (ICLR)*, 2022b. Cited on pages 8, 128, 129, 134, 136, 137, 138, 140, 142, 152, 164, 170, and 171.

Yongxin Chen, Tryphon T Georgiou, and Michele Pavon. Optimal transport over a linear dynamical system. *IEEE Transactions on Automatic Control*, 62(5):2137–2152, 2016. Cited on pages 136 and 151.

Yongxin Chen, Tryphon T Georgiou, and Michele Pavon. Optimal transport in systems and control. *Annual Review of Control, Robotics, and Autonomous Systems*, 4(1):89–113, 2021a. Cited on page 144.

Yongxin Chen, Tryphon T Georgiou, and Michele Pavon. Stochastic control liaisons: Richard sinkhorn meets gaspard monge on a schrodinger bridge. *Siam Review*, 63(2): 249–313, 2021b. Cited on pages 143 and 145.

Yu-Chia Chen and Marina Meila. The decomposition of the higher-order homology embedding constructed from the k-Laplacian. *Advances in Neural Information Processing Systems*, 34:15695–15709, 2021. Cited on pages 123 and 169.

Yu-Chia Chen, Marina Meilă, and Ioannis G Kevrekidis. Helmholtzian Eigenmap: Topological feature discovery & edge flow learning from point cloud data. *arXiv preprint arXiv:2103.07626*, 2021c. Cited on pages 109, 110, 123, 138, 167, and 169.

Yuzhou Chen, Yulia Gel, and H Vincent Poor. Time-Conditioned Dances with Simplicial Complexes: Zigzag Filtration Curve based Supra-Hodge Convolution Networks for Time-series Forecasting. *Advances in Neural Information Processing Systems*, 35:8940–8953, 2022c. Cited on pages 88, 139, and 168.

Yuzhou Chen, Yulia R. Gel, and H. Vincent Poor. BScNets: Block Simplicial Complex Neural Networks. *Proceedings of the AAAI Conference on Artificial Intelligence*, 36(6):6333–6341, 2022d. Cited on pages 50, 53, 69, and 74.

Shui-Nee Chow, Wuchen Li, Chenchen Mou, and Haomin Zhou. Dynamical Schrödinger Bridge Problems on Graphs. *Journal of Dynamics and Differential Equations*, 34(3): 2511–2530, September 2022. Cited on page 141.

Fan RK Chung. *Spectral graph theory*, volume 92. American Mathematical Soc., 1997. Cited on pages 3, 99, and 130.

Soon-Yeong Chung, Yun-Sung Chung, and Jong-Ho Kim. Diffusion and elastic equations on networks. *Publications of the Research Institute for Mathematical Sciences*, 43(3):699–726, 2007. Cited on page 151.

Paolo Dai Pra. A stochastic control approach to reciprocal diffusion processes. *Applied mathematics and Optimization*, 23(1):313–329, 1991. Cited on pages 128, 133, and 145.

Graeme Dandy. 06 Zhi Jiang. International Systems., 2016. Cited on pages 110 and 123.

Valentin De Bortoli, James Thornton, Jeremy Heng, and Arnaud Doucet. Diffusion schrödinger bridge with applications to score-based generative modeling. *Advances in Neural Information Processing Systems*, 34:17695–17709, 2021. Cited on pages 8, 128, 134, 136, 137, and 142.

Michaël Defferrard, Xavier Bresson, and Pierre Vandergheynst. Convolutional Neural Networks on Graphs with Fast Localized Spectral Filtering. In *Advances in Neural Information Processing Systems*, volume 29, 2016. Cited on pages 3, 64, 68, 74, and 88.

Eustasio del Barrio and Jean-Michel Loubes. The statistical effect of entropic regularization in optimal transportation. *arXiv preprint arXiv:2006.05199*, 2020. Cited on page 143.

Mauricio Delbracio and Peyman Milanfar. Inversion by direct iteration: An alternative to denoising diffusion for image restoration. *arXiv preprint arXiv:2303.11435*, 2023. Cited on page 166.

Jean-Charles Delvenne, Renaud Lambiotte, and Luis EC Rocha. Diffusion on networked systems is a question of time or structure. *Nature communications*, 6(1):7366, 2015. Cited on page 132.

Wei Deng, Weijian Luo, Yixin Tan, Marin Biloš, Yu Chen, Yuriy Nevmyvaka, and Ricky T. Q. Chen. Variational Schrödinger Diffusion Models. In *Proceedings of the 41st International Conference on Machine Learning*, volume 235, pp. 10506–10529. PMLR, 2024. Cited on pages 128 and 140.

Lee DeVille. Consensus on Simplicial Complexes: Results on Stability and Synchronization. *Chaos: An Interdisciplinary Journal of Nonlinear Science*, 31(2):023137, February 2021. Cited on page 105.

Mehdi Dini and Massoud Tabesh. A new method for simultaneous calibration of demand pattern and Hazen-Williams coefficients in water distribution systems. *Water resources management*, 28:2021–2034, 2014. Cited on pages 110 and 126.

T. A Driscoll, N. Hale, and L. N. Trefethen. *Chebfun Guide*. Pafnuty Publications, 2014. Cited on page 31.

Vladimir L Druskin and Leonid A Knizhnerman. Two polynomial methods of calculating functions of symmetric matrices. *USSR Computational Mathematics and Mathematical Physics*, 29(6):112–121, 1989. Cited on page 29.

David Duvenaud. *Automatic model construction with Gaussian processes*. PhD thesis, University of Cambridge, 2014. Cited on pages 97, 99, and 117.

David K Duvenaud, Dougal Maclaurin, Jorge Iparraguirre, Rafael Bombarell, Timothy Hirzel, Alán Aspuru-Guzik, and Ryan P Adams. Convolutional networks on graphs for learning molecular fingerprints. *Advances in neural information processing systems*, 28, 2015. Cited on page 3.

Stefania Ebli, Michaël Defferrard, and Gard Spreemann. Simplicial Neural Networks. *arXiv preprint arXiv:2010.03633*, 2020. Cited on pages 5, 50, 53, 59, 64, 66, 68, 69, 73, 74, 86, 87, 88, and 92.

Moshe Eliasof, Eldad Haber, and Eran Treister. PDE-GCN: Novel Architectures for Graph Neural Networks Motivated by Partial Differential Equations, October 2021. Cited on page 179.

Wolfgang Erb. Krylov subspace methods to accelerate kernel machines on graphs. *arXiv preprint arXiv:2301.06384*, 2023. Cited on page 118.

Joshua Faskowitz, Richard F Betzel, and Olaf Sporns. Edges in brain networks: Contributions to models of structure and function. *Network Neuroscience*, 6(1):1–28, 2022. Cited on pages 2, 98, and 128.

Lucas CF Ferreira and Julio C Valencia-Guevara. Gradient flows of time-dependent functionals in metric spaces and applications to PDEs. *Monatshefte für Mathematik*, 185: 231–268, 2018. Cited on page 153.

H Föllmer and A Wakolbinger. Time reversal of infinite-dimensional diffusions. *Stochastic processes and their applications*, 22(1):59–77, 1986. Cited on page 156.

Hans Föllmer. Random fields and diffusion processes. *Lect. Notes Math*, 1362:101–204, 1988. Cited on pages 128, 133, 135, and 144.

Robert Fortet. Résolution d'un système d'équations de M. Schrödinger. *Journal de Mathématiques Pures et Appliquées*, 19(1-4):83–105, 1940. Cited on pages 128 and 133.

Yoshi Fujiwara and Rubaiyat Islam. Hodge decomposition of bitcoin money flow. *Advanced Studies of Financial Technologies and Cryptocurrency Markets*, pp. 117–137, 2020. Cited on pages 4 and 98.

Jason Gaitonde, Jon Kleinberg, and Éva Tardos. Polarization in geometric opinion dynamics. In *Proceedings of the 22nd ACM Conference on Economics and Computation*, pp. 499–519, 2021. Cited on page 132.

Fernando Gama, Antonio G Marques, Geert Leus, and Alejandro Ribeiro. Convolutional graph neural networks. In *2019 53rd Asilomar Conference on Signals, Systems, and Computers*, pp. 452–456. IEEE, 2019a. Cited on page 74.

Fernando Gama, Alejandro Ribeiro, and Joan Bruna. Stability of graph scattering transforms. *Advances in Neural Information Processing Systems*, 32, 2019b. Cited on page 61.

Fernando Gama, Joan Bruna, and Alejandro Ribeiro. Stability properties of graph neural networks. *IEEE Transactions on Signal Processing*, 68:5680–5695, 2020a. Cited on pages 61 and 73.

Fernando Gama, Elvin Isufi, Geert Leus, and Alejandro Ribeiro. Graphs, convolutions, and neural networks: From graph filters to graph neural networks. *IEEE Signal Processing Magazine*, 37(6):128–138, 2020b. Cited on pages 3 and 74.

Jacob Gardner, Geoff Pleiss, Kilian Q Weinberger, David Bindel, and Andrew G Wilson. Gpytorch: Blackbox matrix-matrix gaussian process inference with gpu acceleration. *Advances in neural information processing systems*, 31, 2018. Cited on pages 108 and 121.

Thomas Gebhart, Xiaojun Fu, and Russell J Funk. Go with the Flow? A Large-Scale Analysis of Health Care Delivery Networks in the United States Using Hodge Theory. In *2021 IEEE International Conference on Big Data (Big Data)*, pp. 3812–3823. IEEE, 2021. Cited on pages 4, 12, and 32.

Lorenzo Giambagli, Lucille Calmon, Riccardo Muolo, Timoteo Carletti, and Ginestra Bianconi. Diffusion-Driven Instability of Topological Signals Coupled by the Dirac Operator. *Physical Review E*, 106(6):064314, December 2022. Cited on page 179.

Lorenzo Giusti, Claudio Battiloro, Paolo Di Lorenzo, Stefania Sardellitti, and Sergio Barbarossa. Simplicial Attention Neural Networks. *arXiv preprint arXiv:2203.07485*, 2022. Cited on pages 50, 68, and 74.

Matthew F Glasser, Timothy S Coalson, Emma C Robinson, Carl D Hacker, John Harwell, Essa Yacoub, Kamil Ugurbil, Jesper Andersson, Christian F Beckmann, Mark Jenkinson, et al. A multi-modal parcellation of human cerebral cortex. *Nature*, 536(7615):171–178, 2016. Cited on pages 138 and 169.

Chris Godsil and Gordon Royle. *Algebraic Graph Theory*, volume 207 of *Graduate Texts in Mathematics*. Springer New York, New York, NY, 2001. Cited on pages 98, 108, and 130.

Christopher Wei Jin Goh, Cristian Bodnar, and Pietro Lio. Simplicial Attention Networks. In *ICLR 2022 Workshop on Geometrical and Topological Representation Learning*, 2022. Cited on pages 68 and 74.

Kiya W Govek, Venkata S Yamajala, and Pablo G Camara. Spectral Simplicial Theory for Feature Selection and Applications to Genomics. *arXiv preprint arXiv:1811.03377*, 2018. Cited on page 69.

Leo J Grady and Jonathan R Polimeni. *Discrete calculus: Applied analysis on graphs for computational science*, volume 3. Springer, 2010. Cited on pages 4, 12, 13, 50, 59, 60, 70, 72, 88, 98, 100, 102, and 130.

Nicola Guglielmi, Anton Savostianov, and Francesco Tudisco. Quantifying the structural stability of simplicial homology. *arXiv preprint arXiv:2301.03627*, 2023. Cited on page 60.

Nikita Gushchin, Sergei Kholkin, Evgeny Burnaev, and Alexander Korotin. Light and Optimal Schrödinger Bridge Matching. In *Forty-first International Conference on Machine Learning*, 2024. Cited on page 140.

Kilian Konstantin Haefeli, Karolis Martinkus, Nathanaël Perraudin, and Roger Wattenhofer. Diffusion Models for Graphs Benefit From Discrete State Spaces, August 2023. Cited on page 141.

Mustafa Hajij, Kyle Istvan, and Ghada Zamzmi. Cell Complex Neural Networks. In *NeurIPS 2020 Workshop on Topological Data Analysis and Beyond*, 2020. Cited on pages 70 and 75.

Mustafa Hajij, Ghada Zamzmi, Theodore Papamarkou, Vasileios Maroulas, and Xuanting Cai. Simplicial complex representation learning. *arXiv preprint arXiv:2103.04046*, 2021. Cited on pages 69 and 75.

Mustafa Hajij, Ghada Zamzmi, Theodore Papamarkou, Nina Miolane, Aldo Guzmán-Sáenz, and Karthikeyan Natesan Ramamurthy. Higher-Order Attention Networks. *arXiv preprint arXiv:2206.00606*, 2022. Cited on pages 70 and 75.

David K Hammond, Pierre Vandergheynst, and Rémi Gribonval. Wavelets on graphs via spectral graph theory. *Applied and Computational Harmonic Analysis*, 30(2):129–150, 2011. Cited on page 31.

Jakob Hansen and Robert Ghrist. Toward a spectral theory of cellular sheaves. *Journal of Applied and Computational Topology*, 3(4):315–358, 2019. Cited on pages 70 and 75.

Jeremy Heng, Valentin De Bortoli, Arnaud Doucet, and James Thornton. Simulating diffusion bridges with score matching. *arXiv preprint arXiv:2111.07243*, 2021. Cited on page 166.

Nicholas J Higham. Functions of matrices: theory and computation, 2008. Cited on pages 119 and 136.

Jonathan Ho, Ajay Jain, and Pieter Abbeel. Denoising diffusion probabilistic models. *Advances in neural information processing systems*, 33:6840–6851, 2020. Cited on pages 128 and 132.

William Vallance Douglas Hodge. *The theory and applications of harmonic integrals*. CUP Archive, 1989. Cited on page 69.

Danijela Horak and Jürgen Jost. Spectra of combinatorial Laplace operators on simplicial complexes. *Advances in Mathematics*, 244:303–336, 2013. Cited on pages 60 and 88.

Roger A Horn and Charles R Johnson. *Matrix analysis*. Cambridge university press, 2012. Cited on pages 27, 57, and 77.

Weiyu Huang and Alejandro Ribeiro. Metrics in the space of high order networks. *IEEE Transactions on Signal Processing*, 64(3):615–629, 2015. Cited on page 12.

Weiyu Huang, Thomas AW Bolton, John D Medaglia, Danielle S Bassett, Alejandro Ribeiro, and Dimitri Van De Ville. A graph signal processing perspective on functional brain imaging. *Proceedings of the IEEE*, 106(5):868–885, 2018. Cited on page 12.

Guillaume Huguet, Alexander Tong, María Ramos Zapatero, Christopher J. Tape, Guy Wolf, and Smita Krishnaswamy. Geodesic Sinkhorn for Fast and Accurate Optimal Transport on Manifolds, September 2023. Cited on page 136.

Michael F Hutchinson. A stochastic estimator of the trace of the influence matrix for Laplacian smoothing splines. *Communications in Statistics-Simulation and Computation*, 18(3):1059–1076, 1989. Cited on pages 165 and 171.

Elvin Isufi and Maosheng Yang. Convolutional Filtering in Simplicial Complexes. In *ICASSP 2022 - 2022 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, pp. 5578–5582, May 2022. Cited on pages 6, 71, and 98.

Elvin Isufi, Fernando Gama, David I Shuman, and Santiago Segarra. Graph filters for signal processing and machine learning on graphs. *IEEE Transactions on Signal Processing*, 72: 4745–4781, 2024. Cited on pages 3, 9, and 11.

Benton Jamison. The markov processes of schrödinger. *Zeitschrift für Wahrscheinlichkeitstheorie und verwandte Gebiete*, 32(4):323–331, 1975. Cited on pages 133, 144, and 151.

Hicham Janati, Boris Muzellec, Gabriel Peyré, and Marco Cuturi. Entropic optimal transport between unbalanced gaussian measures has a closed form. *Advances in neural information processing systems*, 33:10468–10479, 2020. Cited on pages 135, 143, and 155.

Junteng Jia, Michael T Schaub, Santiago Segarra, and Austin R Benson. Graph-based semi-supervised & active learning for edge flows. In *Proceedings of the 25th ACM SIGKDD International Conference on Knowledge Discovery & Data Mining*, pp. 761–771, 2019. Cited on pages 2, 5, 13, 34, 35, 50, 59, 64, 98, 108, and 122.

Xiaoye Jiang, Lek-Heng Lim, Yuan Yao, and Yinyu Ye. Statistical ranking and combinatorial Hodge theory. *Mathematical Programming*, 127(1):203–244, 2011. Cited on pages 2, 4, 12, 31, 32, 34, 35, 36, 50, 59, 64, 98, and 108.

Jaehyeong Jo, Seul Lee, and Sung Ju Hwang. Score-based generative modeling of graphs via the system of stochastic differential equations. In *International conference on machine learning*, pp. 10362–10383. PMLR, 2022. Cited on page 141.

Richard Jordan, David Kinderlehrer, and Felix Otto. The Variational Formulation of the Fokker–Planck Equation. *SIAM Journal on Mathematical Analysis*, 29(1):1–17, January 1998. Cited on pages 141 and 153.

Henry Kenlay, Dorina Thano, and Xiaowen Dong. On the stability of graph convolutional neural networks under edge rewiring. In *ICASSP 2021-2021 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, pp. 8513–8517. IEEE, 2021. Cited on page 61.

Bulat Kerimov, Riccardo Taormina, and Franz Tscheikner-Gratl. Towards transferable metamodels for water distribution systems with edge-based graph neural networks. *Water Research*, 261:121933, 2024. Cited on page 179.

Bulat Kerimov, Maosheng Yang, Riccardo Taormina, and Franz Tscheikner-Gratl. State estimation in water distribution system via diffusion on the edge space. *Water Research*, 274:122980, 2025. Cited on pages 9 and 179.

Alexandros D Keros, Vidit Nanda, and Kartic Subr. Dist2Cycle: A Simplicial Neural Network for Homology Localization. *Proceedings of the AAAI Conference on Artificial Intelligence*, 36(7):7133–7142, 2022. Cited on pages 68 and 74.

Thomas N. Kipf and Max Welling. Semi-Supervised Classification with Graph Convolutional Networks. In *International Conference on Learning Representations (ICLR)*, 2017. Cited on pages 3, 55, 68, 74, 137, and 171.

Katherine A Klise, David Hart, Dylan Michael Moriarty, Michael Lee Bynum, Regan Murray, Jonathan Burkhardt, and Terra Haxton. Water network tool for resilience (WNTR) user manual. Technical report, Sandia National Lab.(SNL-NM), Albuquerque, NM (United States), 2017. Cited on pages 110 and 126.

Andrew V Knyazev. Toward the optimal preconditioned eigensolver: Locally optimal block preconditioned conjugate gradient method. *SIAM journal on scientific computing*, 23(2): 517–541, 2001. Cited on pages 109 and 123.

Risi Imre Kondor and John Lafferty. Diffusion kernels on graphs and other discrete structures. In *Proceedings of the 19th international conference on machine learning*, volume 2002, pp. 315–322, 2002. Cited on page 99.

Alexander Korotin, Nikita Gushchin, and Evgeny Burnaev. Light Schrödinger Bridge. In *The Twelfth International Conference on Learning Representations*, 2024. Cited on page 140.

Joshin Krishnan, Rohan Money, Baltasar Beferull-Lozano, and Elvin Isufi. Simplicial vector autoregressive models. *IEEE Transactions on Signal Processing*, 2024. Cited on page 178.

David Lee, Artur Palha, and Marc Gerritsma. Discrete conservation properties for shallow water flows using mixed mimetic spectral elements. *Journal of Computational Physics*, 357:282–304, 2018. Cited on page 179.

See Hian Lee, Feng Ji, and Wee Peng Tay. SGAT: Simplicial Graph Attention Network. *arXiv preprint arXiv:2207.11761*, 2022. Cited on page 68.

Kin K Leung, William A Massey, and Ward Whitt. Traffic models for wireless communication networks. *IEEE Journal on selected areas in Communications*, 12(8):1353–1364, 1994. Cited on page 12.

Geert Leus, Maosheng Yang, Mario Coutino, and Elvin Isufi. Topological Volterra Filters. In *ICASSP 2021-2021 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, pp. 5385–5399. IEEE, 2021. Cited on page 9.

Bo Li, Kaitao Xue, Bin Liu, and Yu-Kun Lai. Bbdm: Image-to-image translation with brownian bridge diffusion models. In *Proceedings of the IEEE/CVF conference on computer vision and pattern Recognition*, pp. 1952–1961, 2023. Cited on page 166.

Lek-Heng Lim. Hodge laplacians on graphs. *SIAM Review*, 62(3):685–715, 2020. Cited on pages 4, 8, 12, 14, 15, 20, 32, 38, 50, 54, 69, 98, 100, 102, 104, and 130.

Finn Lindgren, Håvard Rue, and Johan Lindström. An explicit link between Gaussian fields and Gaussian Markov random fields: the stochastic partial differential equation approach. *Journal of the Royal Statistical Society Series B: Statistical Methodology*, 73(4): 423–498, 2011. Cited on page 99.

Yaron Lipman, Ricky TQ Chen, Heli Ben-Hamu, Maximilian Nickel, and Matt Le. Flow matching for generative modeling. *arXiv preprint arXiv:2210.02747*, 2022. Cited on pages 8 and 128.

Chengyi Liu, Wenqi Fan, Yunqing Liu, Jiatong Li, Hang Li, Hui Liu, Jiliang Tang, and Qing Li. Generative Diffusion Models on Graphs: Methods and Applications, August 2023. Cited on page 141.

Guan-Horng Liu, Yaron Lipman, Maximilian Nickel, Brian Karrer, Evangelos Theodorou, and Ricky T. Q. Chen. Generalized Schrödinger Bridge Matching. In *The Twelfth International Conference on Learning Representations*, 2024. Cited on pages 128 and 141.

Xingchao Liu, Lemeng Wu, Mao Ye, and Qiang Liu. Let us Build Bridges: Understanding and Extending Diffusion Generative Models, August 2022. Cited on page 166.

László Lovász. Discrete analytic functions: an exposition. *Surveys in differential geometry*, 9(1):241–273, 2004. Cited on pages 4 and 98.

R Lumpkin and L Centurioni. Global Drifter Program quality-controlled 6-hour interpolated data from ocean surface drifting buoys. NOAA National Centers for Environmental Information. Dataset, 2019. Cited on page 109.

Albert T Lundell, Stephen Weingram, Albert T Lundell, and Stephen Weingram. Regular and Semisimplicial CW Complexes. *The Topology of CW Complexes*, pp. 77–115, 1969. Cited on page 70.

Christian Léonard. Lazy random walks and optimal transport on graphs, November 2013. Cited on page 141.

Christian Léonard. A survey of the Schrödinger problem and some of its connections with optimal transport. *Discrete & Continuous Dynamical Systems - A*, 34(4):1533–1574, 2014. Cited on pages 128, 129, 133, 135, 143, 144, 152, and 163.

Jan Maas. Gradient flows of the entropy for finite Markov chains. *Journal of Functional Analysis*, 261(8):2250–2292, 2011. Cited on page 141.

Anton Mallasto, Augusto Gerolin, and Hà Quang Minh. Entropy-regularized 2-Wasserstein distance between Gaussian measures. *Information Geometry*, 5(1):289–323, 2022. Cited on pages 143 and 155.

Sibylle Marcotte, Amélie Barbe, Rémi Gribonval, Titouan Vayer, Marc Sebban, Pierre Borgnat, and Paulo Gonçalves. Fast multiscale diffusion on graphs. In *ICASSP 2022-2022 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, pp. 5627–5631. IEEE, 2022. Cited on page 136.

John C Mason and David C Handscomb. *Chebyshev polynomials*. CRC press, 2002. Cited on page 29.

Hosein Masoomy, Behrouz Askari, Samin Tajik, Abbas K Rizi, and G Reza Jafari. Topological analysis of interaction patterns in cancer-specific gene regulatory network: persistent homology approach. *Scientific Reports*, 11(1):1–11, 2021. Cited on page 49.

Ben Mather, Sandra McLaren, David Taylor, Sukanta Roy, and Louis Moresi. Variations and controls on crustal thermal regimes in Southeastern Australia. *Tectonophysics*, 723: 261–276, 2018. Cited on page 167.

James McQueen, Marina Meilă, Jacob VanderPlas, and Zhongyue Zhang. Megaman: Scalable manifold learning in python. *The Journal of Machine Learning Research*, 17(1):5176–5180, 2016. Cited on page 123.

Andrea Mock and Ismar Volic. Political structures and the topology of simplicial complexes. *arXiv preprint arXiv:2104.02131*, 2021. Cited on pages 12 and 32.

Rohan Money, Joshin Krishnan, Baltasar Beferull-Lozano, and Elvin Isufi. Online Edge Flow Imputation on Networks. *IEEE Signal Processing Letters*, 2022. Cited on page 50.

Kevin R Moon, David Van Dijk, Zheng Wang, Scott Gigante, Daniel B Burkhardt, William S Chen, Kristina Yim, Antonia van den Elzen, Matthew J Hirn, Ronald R Coifman, et al. Visualizing structure and transitions in high-dimensional biological data. *Nature biotechnology*, 37(12):1482–1492, 2019. Cited on pages 138 and 169.

Louis Moresi and Ben Mather. Stripy: A Python module for (constrained) triangulation in Cartesian coordinates and on a sphere. *Journal of Open Source Software*, 4(38):1410, 2019. Cited on pages 139 and 168.

Abubakr Muhammad and Magnus Egerstedt. Control Using Higher Order Laplacians in Network Topologies. In *Proc. of 17th International Symposium on Mathematical Theory of Networks and Systems, Kyoto*, pp. 1024–1038, 2006. Cited on page 105.

James Raymond Munkres. *Elements of Algebraic Topology*. The Advanced Book Program. CRC Press, Boca Raton London New York, 2018. Cited on pages 54, 100, and 118.

Kirill Neklyudov, Rob Brekelmans, Daniel Severo, and Alireza Makhzani. Action Matching: Learning Stochastic Dynamics from Samples. In *Proceedings of the 40th International Conference on Machine Learning*, pp. 25858–25889. PMLR, July 2023. Cited on page 128.

Edward Nelson. *Dynamical theories of Brownian motion*, volume 101. Princeton university press, 2020. Cited on page 152.

Mark EJ Newman, Duncan J Watts, and Steven H Strogatz. Random graph models of social networks. *Proceedings of the national academy of sciences*, 99(suppl_1):2566–2572, 2002. Cited on page 49.

Eddie Nijholt and Lee DeVille. Dynamical systems defined on simplicial complexes: Symmetries, conjugacies, and invariant subspaces. *Chaos: An Interdisciplinary Journal of Nonlinear Science*, 32(9), 2022. Cited on page 179.

Alexander V. Nikitin, St John, Arno Solin, and Samuel Kaski. Non-Separable Spatio-temporal Graph Kernels via SPDEs. In *Proceedings of The 25th International Conference on Artificial Intelligence and Statistics*, pp. 10640–10660. PMLR, May 2022. Cited on pages 97, 99, 119, 132, 151, and 178.

Chenhao Niu, Yang Song, Jiaming Song, Shengjia Zhao, Aditya Grover, and Stefano Ermon. Permutation invariant graph generation via score-based generative modeling. In *International Conference on Artificial Intelligence and Statistics*, pp. 4474–4484. PMLR, 2020. Cited on page 141.

Hoang Nt and Takanori Maehara. Revisiting graph neural networks: All we have is low-pass filters. *arXiv preprint arXiv:1905.09550*, 2019. Cited on page 55.

Marco Nurisso, Alexis Arnaudon, Maxime Lucas, Robert L Peach, Paul Expert, Francesco Vaccarino, and Giovanni Petri. A unified framework for Simplicial Kuramoto models. *Chaos: An Interdisciplinary Journal of Nonlinear Science*, 34(5), 2024. Cited on page 179.

Oanda. Oanda Corporation. Foreign Exchange Data, 2018. Cited on page 108.

Bernt Oksendal. *Stochastic differential equations: an introduction with applications*. Springer Science & Business Media, 2013. Cited on page 146.

Antonio Ortega, Pascal Frossard, Jelena Kovačević, José MF Moura, and Pierre Vandergheynst. Graph signal processing: Overview, challenges, and applications. *Proceedings of the IEEE*, 106(5):808–828, 2018. Cited on pages 11 and 27.

Felix Otto. THE GEOMETRY OF DISSIPATIVE EVOLUTION EQUATIONS: THE POROUS MEDIUM EQUATION. *Communications in Partial Differential Equations*, 26(1-2):101–174, 2001. Cited on page 153.

Theodore Papamarkou, Tolga Birdal, Michael M Bronstein, Gunnar E Carlsson, Justin Curry, Yue Gao, Mustafa Hajij, Roland Kwitt, Pietro Lio, Paolo Di Lorenzo, et al. Position: Topological Deep Learning is the New Frontier for Relational Learning. In *Forty-first International Conference on Machine Learning*, 2024. Cited on pages 50, 69, 128, and 130.

Alejandro Parada-Mayorga, Zhiyang Wang, Fernando Gama, and Alejandro Ribeiro. Stability of Aggregation Graph Neural Networks. *arXiv preprint arXiv:2207.03678*, 2022. Cited on page 61.

Alice Patania, Giovanni Petri, and Francesco Vaccarino. The shape of collaborations. *EPJ Data Science*, 6:1–16, 2017. Cited on page 12.

Michele Pavon and Anton Wakolbinger. On free energy, stochastic control, and Schrödinger processes. In *Modeling, Estimation and Control of Systems with Uncertainty: Proceedings of a Conference held in Sopron, Hungary, September 1990*, pp. 334–348. Springer, 1991. Cited on pages 128, 133, and 145.

Michele Pavon, Giulio Trigila, and Esteban G Tabak. The Data-Driven Schrödinger Bridge. *Communications on Pure and Applied Mathematics*, 74(7):1545–1573, 2021. Cited on pages 128 and 134.

Stefano Peluchetti. Diffusion bridge mixture transports, Schrödinger bridge problems and generative modeling. *Journal of Machine Learning Research*, 24(374):1–51, 2023. Cited on page 140.

José Pereira, Morteza Ibrahimi, and Andrea Montanari. Learning networks of stochastic differential equations. *Advances in Neural Information Processing Systems*, 23, 2010. Cited on page 132.

Alessio Perinelli, Davide Tabarelli, Carlo Miniussi, and Leonardo Ricci. Dependence of connectivity on geometric distance in brain networks. *Scientific Reports*, 9(1):13412, 2019. Cited on page 169.

Michael Poli, Stefano Massaroli, Junyoung Park, Atsushi Yamashita, Hajime Asama, and Jinkyoo Park. Graph Neural Ordinary Differential Equations, June 2021. Cited on page 151.

Philip E. Protter. *Stochastic Differential Equations*, pp. 249–361. Springer Berlin Heidelberg, Berlin, Heidelberg, 2005. Cited on page 156.

Qiang Qiu, Xiuyuan Cheng, Guillermo Sapiro, et al. DCFNet: Deep neural network with decomposed convolutional filters. In *International Conference on Machine Learning*, pp. 4198–4207. PMLR, 2018. Cited on page 61.

Carl Edward Rasmussen and Christopher K. I. Williams. *Gaussian Processes for Machine Learning*. Adaptive Computation and Machine Learning. MIT Press, Cambridge, Mass, 2006. Cited on pages 97 and 99.

A Pérez Riascos and José L Mateos. Fractional dynamics on networks: Emergence of anomalous diffusion and Lévy flights. *Physical Review E*, 90(3):032809, 2014. Cited on pages 137 and 150.

Hannes Risken. *Fokker-planck equation*. Springer, 1996. Cited on page 145.

Daniel Robert-Nicoud, Andreas Krause, and Viacheslav Borovitskiy. Intrinsic Gaussian Vector Fields on Manifolds. In *International Conference on Artificial Intelligence and Statistics*, 2024. Cited on page 107.

T Mitchell Roddenberry and Santiago Segarra. HodgeNet: Graph neural networks for edge data. In *2019 53rd Asilomar Conference on Signals, Systems, and Computers*, pp. 220–224. IEEE, 2019. Cited on pages 5, 25, 50, 68, and 92.

T Mitchell Roddenberry, Nicholas Glaze, and Santiago Segarra. Principled simplicial neural networks for trajectory prediction. In *International Conference on Machine Learning*, pp. 9020–9029. PMLR, 2021. Cited on pages 19, 50, 53, 54, 60, 64, 68, 69, 73, 74, 86, 88, 92, 93, 98, 137, and 171.

T Mitchell Roddenberry, Michael T Schaub, and Mustafa Hajij. Signal processing on cell complexes. In *ICASSP 2022-2022 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, pp. 8852–8856. IEEE, 2022. Cited on pages 70 and 75.

Vincent Rouvreau. Alpha complex. In *GUDHI User and Reference Manual*. GUDHI Editorial Board, 2015. Cited on page 85.

T. Konstantin Rusch, Ben Chamberlain, James Rowbottom, Siddhartha Mishra, and Michael Bronstein. Graph-Coupled Oscillator Networks. In *Proceedings of the 39th International Conference on Machine Learning*, pp. 18888–18909. PMLR, June 2022. Cited on page 179.

T Konstantin Rusch, Michael M Bronstein, and Siddhartha Mishra. A Survey on Oversmoothing in Graph Neural Networks. *arXiv preprint arXiv:2303.10993*, 2023. Cited on page 55.

Aliaksei Sandryhaila and José MF Moura. Discrete signal processing on graphs. *IEEE Transactions on Signal Processing*, 61(7):1644–1656, 2013. Cited on pages 3, 11, 16, 27, 74, and 132.

Aliaksei Sandryhaila and José MF Moura. Discrete signal processing on graphs: Frequency analysis. *IEEE Transactions on Signal Processing*, 62(12):3042–3054, 2014. Cited on pages 3, 11, 16, 27, 74, and 132.

Augusto Santos, Diogo Rente, Rui Seabra, and José MF Moura. Learning the causal structure of networked dynamical systems under latent nodes and structured noise. In *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 38, pp. 14866–14874, 2024. Cited on page 132.

Stefania Sardellitti, Sergio Barbarossa, and Lucia Testa. Topological signal processing over cell complexes. In *2021 55th Asilomar Conference on Signals, Systems, and Computers*, pp. 1558–1562. IEEE, 2021. Cited on pages 70 and 75.

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)*, pp. 735–739. IEEE, 2018. Cited on pages 5, 12, 13, 31, 33, and 35.

Michael T Schaub, Jörg Lehmann, Sophia N Yaliraki, and Mauricio Barahona. Structure of complex networks: Quantifying edge-to-edge relations by failure-induced flow redistribution. *Network Science*, 2(1):66–89, 2014. Cited on pages 31, 37, and 98.

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. Cited on pages 15, 25, 36, 60, 68, 88, 92, 93, 105, and 130.

Michael T Schaub, Yu Zhu, Jean-Baptiste Seby, T Mitchell Roddenberry, and Santiago Segarra. Signal processing on higher-order networks: Livin'on the edge... and beyond. *Signal Processing*, 187:108149, 2021. Cited on pages 9, 12, 13, 14, 20, 22, 33, 35, 50, 53, 55, 71, 98, and 101.

Erwin Schrödinger. *Über die umkehrung der naturgesetze*. Verlag der Akademie der Wissenschaften in Kommission bei Walter De Gruyter u, 1931. Cited on pages 7 and 128.

Erwin Schrödinger. Sur la théorie relativiste de l'électron et l'interprétation de la mécanique quantique. In *Annales de l'institut Henri Poincaré*, volume 2, pp. 269–310, 1932. Cited on pages 7 and 128.

Nicholas Sharp, Yousuf Soliman, and Keenan Crane. The vector heat method. *ACM Transactions on Graphics (TOG)*, 38(3):1–19, 2019. Cited on page 105.

Yuyang Shi, Valentin De Bortoli, Andrew Campbell, and Arnaud Doucet. Diffusion Schrödinger Bridge Matching. In *Thirty-seventh Conference on Neural Information Processing Systems*, 2023. Cited on page 140.

David I Shuman, Sunil K Narang, Pascal Frossard, Antonio Ortega, and Pierre Vandergheynst. The emerging field of signal processing on graphs: Extending high-dimensional data analysis to networks and other irregular domains. *IEEE Signal Processing Magazine*, 30(3):83–98, 2013. Cited on pages 3, 11, 13, 16, and 130.

David I Shuman, Pierre Vandergheynst, Daniel Kressner, and Pascal Frossard. Distributed signal processing via Chebyshev polynomial approximation. *IEEE Transactions on Signal and Information Processing over Networks*, 4(4):736–751, 2018. Cited on pages 28, 29, and 31.

Gerard LG Sleijpen and Henk A Van der Vorst. A Jacobi–Davidson iteration method for linear eigenvalue problems. *SIAM review*, 42(2):267–293, 2000. Cited on page 28.

Alexander J Smola and Risi Kondor. Kernels and regularization on graphs. In *Learning theory and kernel machines*, pp. 144–158. Springer, 2003. Cited on page 97.

Jascha Sohl-Dickstein, Eric A. Weiss, Niru Maheswaranathan, and Surya Ganguli. Deep Unsupervised Learning using Nonequilibrium Thermodynamics, November 2015. Cited on page 132.

Justin Solomon. Optimal Transport on Discrete Domains, May 2018. Cited on page 141.

Yang Song and Stefano Ermon. Generative Modeling by Estimating Gradients of the Data Distribution, October 2020. Cited on page 132.

Yang Song, Sahaj Garg, Jiaxin Shi, and Stefano Ermon. Sliced score matching: A scalable approach to density and score estimation. In *Uncertainty in Artificial Intelligence*, pp. 574–584. PMLR, 2020a. Cited on page 165.

Yang Song, Jascha Sohl-Dickstein, Diederik P Kingma, Abhishek Kumar, Stefano Ermon, and Ben Poole. Score-based generative modeling through stochastic differential equations. *arXiv preprint arXiv:2011.13456*, 2020b. Cited on pages 8, 128, 129, 132, 136, 137, 141, 164, 165, 166, and 171.

Ben Stabler, Hillel Bar-Gera, and Elizabeth Sall. Transportation networks for research core team, 2018. Cited on page 39.

John Steenbergen. *Towards a spectral theory for simplicial complexes*. PhD thesis, Duke University, 2013. Cited on pages 51 and 69.

Simo Särkkä and Arno Solin. *Applied Stochastic Differential Equations*. Cambridge University Press, 1 edition, April 2019. Cited on pages 132, 135, 146, 147, 152, and 165.

T. Tarhasaari, L. Kettunen, and A. Bossavit. Some Realizations of a Discrete Hodge Operator: A Reinterpretation of Finite Element Techniques [for EM Field Analysis]. *IEEE Transactions on Magnetics*, 35(3):1494–1497, May 1999. Cited on page 179.

F. L. Teixeira. Geometric Aspects of the Simplicial Discretization of Maxwell's Equations. *Progress In Electromagnetics Research*, 32:171–188, 2001. Cited on page 179.

Alexander Tong. Processed Single-cell RNA Time Series Data, 2023. Cited on pages 138 and 169.

Alexander Tong, Kilian FATRAS, Nikolay Malkin, Guillaume Huguet, Yanlei Zhang, Jarrid Rector-Brooks, Guy Wolf, and Yoshua Bengio. Improving and generalizing flow-based generative models with minibatch optimal transport. *Transactions on Machine Learning Research*, 2024a. Cited on pages 128, 138, and 169.

Alexander Y. Tong, Nikolay Malkin, Kilian Fatras, Lazar Atanackovic, Yanlei Zhang, Guillaume Huguet, Guy Wolf, and Yoshua Bengio. Simulation-Free Schrödinger Bridges via Score and Flow Matching. In *Proceedings of The 27th International Conference on Artificial Intelligence and Statistics*, 2024b. Cited on pages 138, 140, and 169.

Leo Torres, Ann S Blevins, Danielle Bassett, and Tina Eliassi-Rad. The why, how, and when of representations for complex systems. *SIAM Review*, 63(3):435–485, 2021. Cited on page 49.

David C Van Essen, Stephen M Smith, Deanna M Barch, Timothy EJ Behrens, Essa Yacoub, Kamil Ugurbil, Wu-Minn HCP Consortium, et al. The WU-Minn human connectome project: an overview. *Neuroimage*, 80:62–79, 2013. Cited on pages 138 and 169.

Francisco Vargas, Pierre Thodoroff, Neil D. Lawrence, and Austen Lamacraft. Solving Schrödinger Bridges via Maximum Likelihood. *Entropy*, 23(9):1134, August 2021. Cited on pages 128, 134, and 136.

Arun Venkitaraman, Saikat Chatterjee, and Peter Handel. Gaussian processes over graphs. In *ICASSP 2020-2020 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, pp. 5640–5644. IEEE, 2020. Cited on page 97.

Prakhar Verma, Vincent Adam, and Arno Solin. Variational Gaussian Process Diffusion Processes. In *International Conference on Artificial Intelligence and Statistics*, pp. 1909–1917. PMLR, 2024. Cited on page 145.

D Vijay Anand, Soumya Das, and Moo K Chung. Hodge-Decomposition of Brain Networks. *arXiv e-prints*, pp. arXiv–2211, 2022. Cited on pages 4 and 98.

Cédric Villani. *Optimal transport: old and new*. Number 338 in Grundlehren der mathematischen Wissenschaften. Springer, Berlin Heidelberg, 2009. Cited on pages 8, 130, and 143.

S Vichy N Vishwanathan, Nicol N Schraudolph, Risi Kondor, and Karsten M Borgwardt. Graph kernels. *Journal of Machine Learning Research*, 11:1201–1242, 2010. Cited on page 97.

Tobias Wand, Oliver Kamps, and Hiroshi Iyetomi. Causal Hierarchy in the Financial Market Network—Uncovered by the Helmholtz–Hodge–Kodaira Decomposition. *Entropy*, 26 (10):858, 2024. Cited on page 4.

Gefei Wang, Yuling Jiao, Qian Xu, Yang Wang, and Can Yang. Deep generative learning via schrödinger bridge. In *International conference on machine learning*, pp. 10794–10804. PMLR, 2021. Cited on page 128.

Yuyang Wang, Jianren Wang, Zhonglin Cao, and Amir Barati Farimani. Molecular contrastive learning of representations via graph neural networks. *Nature Machine Intelligence*, 4(3):279–287, 2022. Cited on page 128.

Larry Wasserman. Topological data analysis. *Annual Review of Statistics and Its Application*, 5:501–532, 2018. Cited on page 13.

David S Watkins. *The matrix eigenvalue problem: GR and Krylov subspace methods*. SIAM, 2007. Cited on page 28.

Peter Whittle. Stochastic-processes in several dimensions. *Bulletin of the International Statistical Institute*, 40(2):974–994, 1963. Cited on page 99.

Felix Wu, Amauri Souza, Tianyi Zhang, Christopher Fifty, Tao Yu, and Kilian Weinberger. Simplifying graph convolutional networks. In *International conference on machine learning*, pp. 6861–6871. PMLR, 2019. Cited on page 74.

Keyulu Xu, Weihua Hu, Jure Leskovec, and Stefanie Jegelka. How powerful are graph neural networks? *arXiv preprint arXiv:1810.00826*, 2018a. Cited on page 69.

Keyulu Xu, Chengtao Li, Yonglong Tian, Tomohiro Sonobe, Ken-ichi Kawarabayashi, and Stefanie Jegelka. Representation learning on graphs with jumping knowledge networks. In *International conference on machine learning*, pp. 5453–5462. PMLR, 2018b. Cited on page 52.

Maosheng Yang. Topological Schrödinger Bridge Matching. In *The Thirteenth International Conference on Learning Representations*, 2025. Cited on pages 7 and 127.

Maosheng Yang, Elvin Isufi, Michael T. Schaub, and Geert Leus. Finite Impulse Response Filters for Simplicial Complexes. In *2021 29th European Signal Processing Conference (EUSIPCO)*, pp. 2005–2009, August 2021. Cited on pages 5, 11, 26, 51, 69, and 88.

Maosheng Yang, Elvin Isufi, and Geert Leus. Simplicial convolutional neural networks. In *ICASSP 2022-2022 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, pp. 8847–8851. IEEE, 2022a. Cited on pages 19, 49, 53, 64, 68, 69, 73, 74, 88, and 92.

Maosheng Yang, Elvin Isufi, Michael T Schaub, and Geert Leus. Simplicial convolutional filters. *IEEE Transactions on Signal Processing*, 70:4633–4648, 2022b. Cited on pages 5, 11, 50, 52, 53, 98, 132, and 151.

Maosheng Yang, Bishwadeep Das, and Elvin Isufi. Online edge flow prediction over expanding simplicial complexes. In *ICASSP 2023-2023 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, pp. 1–5. IEEE, 2023. Cited on page 9.

Maosheng Yang, Viacheslav Borovitskiy, and Elvin Isufi. Hodge-Compositional Edge Gaussian Processes. In *International Conference on Artificial Intelligence and Statistics*, volume 238, pp. 3754–3762. PMLR, 2024. Cited on pages 7, 64, 97, 131, 138, and 169.

Maosheng Yang, Geert Leus, and Elvin Isufi. Hodge-Aware Convolutional Learning on Simplicial Complexes. *Transactions on Machine Learning Research*, 2025. ISSN 2835-8856. Cited on pages 6 and 49.

Ruochen Yang, Frederic Sala, and Paul Bogdan. Efficient Representation Learning for Higher-Order Data with Simplicial Complexes. In *The First Learning on Graphs Conference*, 2022c. Cited on pages 53, 55, 60, 69, 74, and 75.

Mao Ye, Lemeng Wu, and Qiang Liu. First hitting diffusion models for generating manifold, graph and categorical data. *Advances in Neural Information Processing Systems*, 35: 27280–27292, 2022. Cited on page 141.

Hyejin Youn, Michael T Gastner, and Hawoong Jeong. Price of anarchy in transportation networks: efficiency and optimality control. *Physical review letters*, 101(12):128701, 2008. Cited on pages 31 and 37.

Anthony Zee. *Quantum field theory in a nutshell*, volume 7. Princeton university press, 2010. Cited on page 155.

Muhan Zhang and Yixin Chen. Link prediction based on graph neural networks. *Advances in neural information processing systems*, 31, 2018. Cited on pages 66 and 87.

Yin-Cong Zhi, Yin Cheng Ng, and Xiaowen Dong. Gaussian Processes on Graphs Via Spectral Kernel Learning. *IEEE Transactions on Signal and Information Processing over Networks*, 9:304–314, 2023. Cited on page 97.

Linqi Zhou, Aaron Lou, Samar Khanna, and Stefano Ermon. Denoising Diffusion Bridge Models. In *The Twelfth International Conference on Learning Representations*, 2024. Cited on pages 129, 137, and 166.

Xiao Zhou, Shuming Liu, Weirong Xu, Kunlun Xin, Yipeng Wu, and Fanlin Meng. Bridging hydraulics and graph signal processing: A new perspective to estimate water distribution network pressures. *Water Research*, 217:118416, 2022. Cited on pages 4, 98, and 110.

Cameron Ziegler, Per Sebastian Skardal, Haimonti Dutta, and Dane Taylor. Balanced Hodge Laplacians optimize consensus dynamics over simplicial complexes. *Chaos: An Interdisciplinary Journal of Nonlinear Science*, 32(2):023128, 2022. Cited on pages 56, 72, and 179.

Valentin Zobel, Jan Reininghaus, and Ingrid Hotz. Generalized Heat Kernel Signatures. *Journal of WSCG*, 2011. Cited on page 105.

Acknowledgments

This thesis project is the result of five years of collaboration. I want to say *thanks* to everyone who has supported me throughout the past five years. I would not have come this far without you.

To *Elvin Isufi* — thank you for all of your mentorship. You brought me into this PhD project and your advice has been invaluable. In the beginning years of my PhD, also the year of my master project, I have learned a lot from you — how to read papers, how to present papers, how to choose research topics, and how to work on them. You are a great mentor and I am grateful for your support.

To *Geert Leus* — thank you for the support during these years. When I had difficulities, about either technical questions or my PhD progress, I know I can trust you. From working with you, I realized what kind of researcher I want to be. It was my honor to have you as my promotor.

To *Michael T. Schaub* — I appreciate the two collaborations we had in the beginning of my PhD. Thank you for introducing me Lek-Heng Lim's work on the Hodge decomposition.

To *Slava* (Viacheslav Borovitskiy) — I am grateful for the collaboration we had on the Gaussian process work. It was a great beginning work for statistical learning on simplicial complexes and I was able to find new directions starting from this work. I also appreciate your help on introducing me, together with my work, to broder audiences and communities.

To *Alan Hanjalic* — thank you for raising the awareness of social safety in our department and taking actions to improve it. This is utterly important for the well-being of PhD students, including myself. I feel lucky to have you as the group leader.

I am grateful to all of my committee members for their time, insight, and expertise. I was fortunate to share the working space with Bishwadeep, Mohammad, Andrea, Tianqi, Li, Shilun, Alex, Roberto, Bulat, and Vimal. Thank you for the discussions and the fun times we had together. I also want to thank Saskia, Ruud, and the department staff for all of their help during these five years. To my friends, Yuanyuan, Ned, Tian and Yanbin, thank you for the fun times we had together. I cannot imagine my life in Delft without you.

To my family, thank you for everything. To Maarten, I am lucky to have you behind all this way.

Maosheng
Delft, May 2025

Curriculum Vitæ

Maosheng YANG

20/11/1996	Date of birth in Shanxi, China
2011–2014	Secondary education at No. 1 High School of Xinzhou, China
2014–2018	B.Sc. in Electrical Engineering, Beijing Jiaotong University, China
2018–2020	M.Sc. in Electrical Engineering, Delft University of Technology, the Netherlands
2020–2025	Ph.D., Delft University of Technology