University of Padova

Department of Mathematics "Tullio Levi-Civita"

Master Thesis in Data Science

The Hodge Laplacian in Higher-Order Data Analysis

Supervisor

Dr. Ludovico Bruni Bruno

Master Candidate

Luca Bogani

Student ID

2097389

Academic Year

2024-2025

Abstract

Graph theory is a well-established and widely used framework for modeling problems in data analysis, and one of the most helpful tools in this context is the graph Laplacian. The capacity to represent pairwise relations is indeed crucial, but is often not enough. If the underlying structure of the problem deals with relations between more than two entities, or even if the information we are interested in flows on the edges rather than on the vertices, we need a more complex instrument to properly capture these nuances. For this reason we introduce simplicial complexes, a higher-order generalization of the concept of graph that can encode more of the subtleties of the problem at hand. As simplicial complexes act as the natural evolution of graphs, the Hodge Laplacian replaces the usual graph Laplacian as the corresponding operator acting on simplicial complexes. This operator captures relations of simplicial complexes of different dimensions, extending the concept of adjacency that is classically only defined between vertices and edges. To illustrate the potential of the Hodge Laplacian, we show data analysis applications in which the limits faced by graphs and graph Laplacian are overcome through the use of this new, more sophisticated operator.

Contents

Abstract iii

List of figures vii

List of definitions ix

1 Introduction 1

2 Graphs 3
2.1 Graphs . . . 4
2.2 Special Families of Graphs . . . 5
2.2.1 Paths and cycles . . . 7
2.2.2 Trees and forests . . . 7
2.3 Incidence and Adjacency Matrices . . . 8
2.4 Vertex Degree . . . 9
2.5 Isomorphisms and Automorphisms . . . 10
2.6 Planar Graphs . . . 13
2.7 Constructing Graphs from Other Graphs . . . 13
2.8 Directed Graphs . . . 15
2.9 Hypergraphs . . . 17

3 Graph Laplacian 19
3.1 Graph Laplacian . . . 19
3.1.1 Some Basic Properties . . . 20
3.1.2 Laplacian eigenvectors . . . 22
3.2 Denoising of graph signals . . . 22
3.2.1 Graph Fourier Transform . . . 22
3.2.2 Convolution on graphs . . . 23
3.2.3 Filters . . . 24
3.3 Spectral Clustering . . . 25
3.3.1 Similarity Graphs . . . 25
3.3.2 Idea behind the clustering . . . 26

3.3.3 Example of Clustering Algorithm . . . 27
3.4 Dimensionality Reduction . . . 29
3.4.1 Embedding graph data . . . 29

4 Simplicial Complexes 31
4.1 Δ-Complexes . . . 32
4.2 Simplicial Homology . . . 34
4.2.1 A Theoretical Digression . . . 36
4.3 Simplicial Complex . . . 37
4.3.1 Conventions and orientation . . . 38

5 Hodge Laplacian 41
5.1 Hodge Laplacian . . . 42
5.2 Hodge Decomposition . . . 48
5.3 Denoising . . . 49
5.3.1 Line-graph approach . . . 50
5.3.2 Edge-space approach . . . 51
5.4 Spectral Clustering . . . 51
5.4.1 Normalized Hodge Laplacian . . . 52
5.4.2 Embedding a flow . . . 53
5.4.3 Embedding trajectories . . . 54
5.5 Community detection . . . 55
5.5.1 Up- and down- Laplacian . . . 55
5.5.2 The Algorithm . . . 57
5.5.3 Communities in Language . . . 58

6 Conclusions 59

References 61

List of figures

2.1 First example of graph 5
2.2 Notable families of graphs 6
2.3 Connected and disconnected graphs 6
2.4 Paths and cycles 7
2.5 Trees 8
2.6 Incidence and Adjacency matrices 9
2.7 Isomorphic graphs 11
2.8 Special notations for common graphs 12
2.9 Petersen graph 12
2.10 Petersen graph is not planar 13
2.11 Union and intersection of two graphs 14
2.12 Cartesian product of graphs 15
2.13 Prism graphs 15
2.14 Directed graphs 16
2.15 Hypergraph 17

3.1 The graph Laplacian 20
3.2 Noise on the Bunny graph 23
3.3 Fourier coefficients of a noisy signal 24
3.4 Bunny graph denoised 25
3.5 Similarity graphs 26
3.6 Similarity graph of the two-moons 28
3.7 Eigenvectors of the two-moons 28
3.8 The Swiss Roll 29
3.9 Eigenvectors of the Swiss Roll 30

4.1 n-simplices 32
4.2 All the parts of a 2-simplex 33
4.3 Unexpected representation of a torus 34
4.4 n-chains of simplices 35
4.5 First example of simplicial complex 38

5.1 Numerical computation of the Hodge Laplacian 42

5.2 Hodge decomposition 49
5.3 Noisy flow on the edges 49
5.4 Denoised flow with graph Laplacian 50
5.5 Denoised flow with Hodge Laplacian 51
5.6 Flow on a punctured simplicial complex 53
5.7 Flow projection onto harmonic functions 54
5.8 Trajectories on a simplicial complex 54
5.9 Community detection on simplicial complex 56
5.10 3-simplicial community detected 56

List of Definitions

2.1 Graph . . . 4
2.2 Graph nomenclature . . . 4
2.5 Notable families of graphs . . . 5
2.6 Connectivity . . . 6
2.7 Paths and cycles . . . 7
2.8 Trees and forests . . . 7
2.9 Incidence and Adjacency matrix . . . 8
2.10 Vertex degree . . . 9
2.13 k-regular graph . . . 10
2.14 Isomorphisms between graphs . . . 10
2.15 Automorphisms of graphs . . . 12
2.16 Planar graphs . . . 13
2.19 Union and intersection of graphs . . . 14
2.20 Cartesian product of graphs . . . 14
2.21 Directed graph . . . 15
2.22 In-degree and out-degree . . . 16
2.23 Adjacency matrix for directed graphs . . . 16
2.25 Hypergraph . . . 17

3.4 High-pass and low-pass filters . . . 24

4.1 n-simplex . . . 32
4.2 Geometry of a simplex . . . 32
4.3 Δ-complex . . . 33
4.4 n-chain . . . 34
4.5 Boundary homomorphism . . . 34
4.10 Simplicial complex . . . 37
4.12 Skeleton of a simplicial complex . . . 38
4.13 Upper and Lower adjacency . . . 39

5.1 Matrix representation of the boundary operator . . . 42
5.2 Hodge Laplacian . . . 42
5.6 Up and Down Laplacian . . . 44

5.12 Line-graph 50
5.13 Edge Laplacian 51
5.14 Normalized Hodge Laplacian 52

1 Introduction

One of the challenges in data science is understanding the structure and dynamics of complex systems. We are often interested in how information flows in such systems, and how in turn the flow is shaped by the underlying structure. Graph theory is a first approach that provides a rather intuitive foundation by modeling pairwise interactions between entities.

The graph structure can emerge from multiple real-world phenomena; it can model a social network and the relations between its users, or represent neurons and axons in the brain as vertices and edges. As such, applications of graph theory arise in many different fields, from computer science to biology and medicine.

However, such a simple framework fails when dealing with systems that exhibit higher-order interactions. When groups of more than two entities are involved, their relations cannot be fully captured by graphs alone and the usual techniques employed don't achieve satisfying results.

We approach this problem through a natural progression that starts from graphs and proceeds to higher-order structures known as simplicial complexes, which can model interactions between more than two entities. Motivated by the work of Bick et al. [1], we see how the properties of this new structure allow us to better face more complex situations.

To aid us in this endeavour, in a parallel to what is done in graph theory, we introduce the Hodge Laplacian, the higher-order generalization of the well-known graph Laplacian. Throughout this path, we deal with the theoretical footings found in algebraic topology and provide examples on some of the many applications that these powerful tools allow.

The thesis is structured in four main chapters:

- In chapter 2, we introduce graphs and lay the foundation of all the following work. We start with the most basic concepts and definitions, we underline some of the nuances of their representation and conclude with a remark on directed graphs that will come in handy at a later time.
- In chapter 3, we define the Graph Laplacian, an operator of the highest relevance in spectral graph theory. We examine its properties, especially the ones related to its spectrum, and show some of the applications that this operator allows, namely denoising, spectral clustering and dimensionality reduction.
- In chapter 4, we extend the notion of graph, by looking at a higher order structure that can model more complex interactions, the simplicial complex. We develop the theoretical groundwork in algebraic topology and introduce concepts such as chains and boundary operators.
- In chapter 5, we introduce the concept of Hodge Laplacian as a generalization of the graph Laplacian to simplicial complexes. We show how it is defined and how the Hodge decomposition can be leveraged. In a parallel to chapter 3, we show applications on denoising, spectral clustering and community detection, underlining the shortcomings of the graph Laplacian and justifying the use of the Hodge Laplacian.

2

Graphs

As with everything, it is always better to start from the beginning, by laying the theoretical foundations on which all the successive work will be built on. In our case, the starting point is graph theory. A powerful abstraction to represent interconnected entities, graph theory was first formalized by Leonhard Euler in 1736. Walking is a good way of clearing one's head and exercising, and the city of Königsberg had a nice path across two islands on the Pregel river, connected to each other and to the mainland by seven bridges. Euler couldn't just enjoy his strolls, and instead he wondered whether it was possible to walk through the city while crossing each bridge exactly once. Since Euler not only had good legs but was also a great mathematician, he solved the problem and laid the foundation of graph theory as a whole.

Nowadays, graph theory is used for much more than planning the best route for a walk, and the applications are numerous and span multiple fields of knowledge. Graphs are used to model complex networks in computer science, the structure of molecules in chemistry, the syntax and grammar in linguistic studies and so on and so forth. In this chapter, we will introduce graphs as a building block to develop more complex structures that manage what graphs cannot. While many theoretical problems are surely interesting and deserving of attention, we will just focus on the main characteristics of graphs in a way to give the reader the necessary knowledge for the chapters to come.

2.1 Graphs

We begin with some basic definitions that are needed to then further explore the more complex topics. While there are many ways to talk about the topic at hand, in this chapter we chose to follow the definitions and conventions used by Bondy and Murty [2] and Diestel [3].

Definition 2.1 (Graph)**.** A ***graph*** G is an ordered pair $(V(G), E(G))$, often also written as $G = (V, E)$ for brevity, consisting of a set V of *vertices* (or *nodes*/*points*) and a set E of *edges* (or *lines*) such that $E \subseteq [V]^2$; that is, there is a so-called *incidence function* ψ_G that associates with each edge an unordered pair of vertices.

Definition 2.2 (Nomenclature)**.** If e is an edge and u, v are vertices such that $\psi_G(e) = \{u, v\}$, then e is said to *join* u and v, and the vertices u and v are called the *ends* of e. For simplicity, we will often write uv for the unordered pair $\{u, v\}$. The ends of an edge are said to be ***incident*** with the edge, and vice versa.
Two vertices which are incident to a common edge are called ***adjacent***, as are two edges adjacent to the same vertex.
Two distinct adjacent vertices are called ***neighbours***, and the set of neighbours of a vertex v in a graph G is denoted by $N_G(v)$.
An edge with identical ends is called a ***loop***, and an edge with distinct ends a ***link***.
Two or more links with the same pair of ends are said to be ***parallel edges***.
The graph with no vertices is called the ***null graph***, and any graph with just one vertex is referred to as *trivial*; all other graphs are *nontrivial*.
A graph is ***simple*** if it has no loops or parallel edges.

Remark 2.3. A set V, together with a set E of two-element subsets of V, defines a simple graph $G(V, E)$, where the ends of an edge uv are precisely the vertices u and v. In any simple graph we may dispense with the incidence function ψ by renaming each edge as the unordered pair of its ends. In a diagram of such a graph, the labels on the edges may then be omitted. When there is no chance for ambiguity, we will omit the letter G and write, for example, V and E instead of $V(G)$ and $E(G)$. In such instances, we will denote the numbers of vertices and edges of G by n and m, respectively.

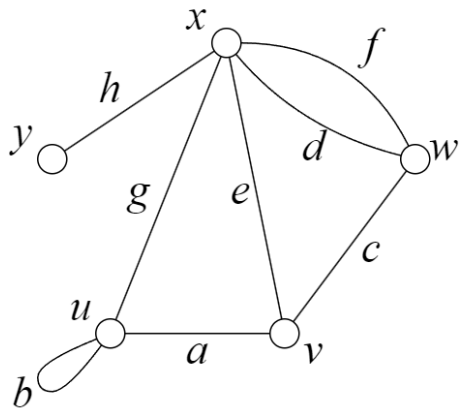
Example 2.4. Define the graph $G = (V, E)$

$$\text{where} \quad V = \{u, v, w, x, y\}\ ,\ E = \{a, b, c, d, e, f, g, h\}$$

$$\text{and } \psi_G \text{ is defined by}$$

$$\psi_G(a) = uv \quad \psi_G(b) = uu \quad \psi_G(c) = vw \quad \psi_G(d) = wx$$
$$\psi_G(e) = vx \quad \psi_G(f) = wx \quad \psi_G(g) = ux \quad \psi_G(h) = xy$$

The graph G can then be represented graphically as



A representation of the graph G

2.2 Special Families of Graphs

Certain types of graphs play prominent roles in graph theory, and as such we felt it was necessary to introduce and define them once and for all. Similarly, notions such as connectivity are fundamental to the understanding of the topic and cannot be left undefined.

Definition 2.5 (Notable families of graphs)**.** A ***complete*** graph is a simple graph in which any two vertices are adjacent; an ***empty*** graph one in which no two vertices are adjacent (the edge set is empty).

A graph is ***bipartite*** if its vertex set can be partitioned into two subsets X and Y so that every edge has one end in X and one end in Y; such a partition (X, Y) is called a *bipartition* of the graph, and X and Y its *parts*. We denote a bipartite graph G with bipartition (X, Y) by $G[X, Y]$. If $G[X, Y]$ is simple and every vertex in X is joined to every vertex in Y, then G is called a *complete bipartite graph*.
A ***star*** is a complete bipartite graph $G[X, Y]$ with $|X| = 1$ or $|Y| = 1$.

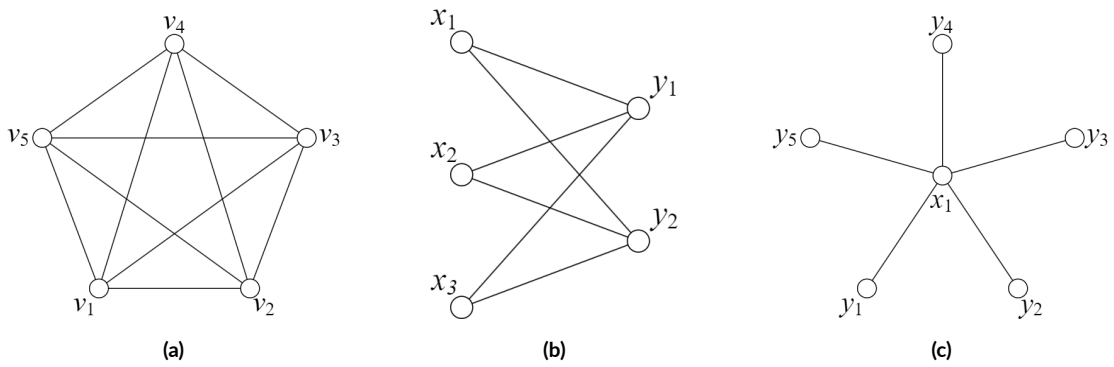


Figure 2.2: **(a)** A complete graph, **(b)** a complete bipartite graph and **(c)** a star

Definition 2.6 (Connectivity)**.** A graph is ***connected*** if, for every partition (X, Y) of its vertex set, there is an edge with one end in X and one end in Y; otherwise, the graph is ***disconnected***. In other words, a graph is disconnected if its vertex set can be partitioned in two nonempty subsets X and Y such that no edge has one end in X and one end in Y.

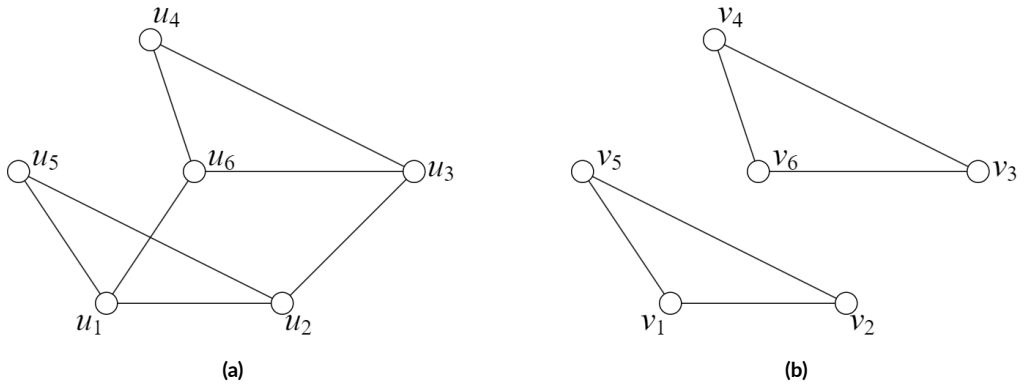


Figure 2.3: **(a)** A connected graph and **(b)** a disconnected graph

2.2.1 Paths and cycles

Definition 2.7 (Paths and Cycles)**.** A ***path*** is a simple graph whose vertices can be arranged in a linear sequence in such a way that two vertices are adjacent if they are consecutive in the sequence, and are otherwise non-adjacent. A ***cycle*** on three or more vertices is a simple graph whose vertices can be arranged in a cyclic sequence in such a way that two vertices are adjacent if they are consecutive in the sequence, and are otherwise non-adjacent. The ***length*** of a path or cycle is the number of its edges; a path or cycle of length k is called a $k-$ *path* or $k-$ *cycle*, respectively, and it is *odd* or *even* according to the parity of k. A 3-cycle is often called a *triangle*, a 4-cycle a *quadrilateral*, and so on.

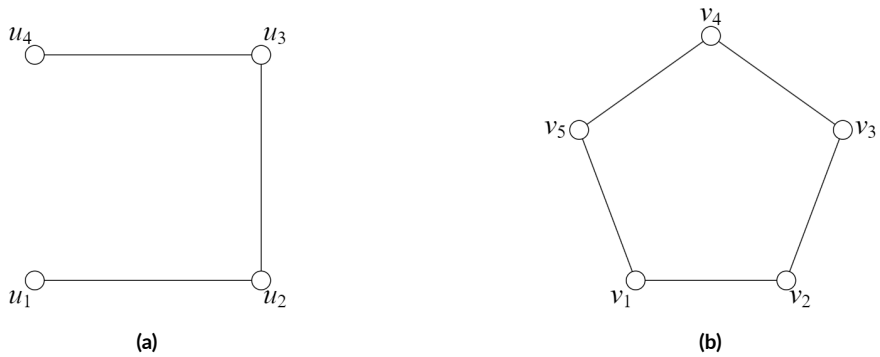


Figure 2.4: (a) A 3-path and **(b)** a 5-cycle

The existence of an edge joining two vertices and the existence of a path joining two vertices are both concepts of distance on a graph; however, the strength of such concepts is clearly different. A large part of the forthcoming chapters may be interpreted as a further strengthening of the idea of connectedness of graphs.

2.2.2 Trees and forests

Definition 2.8 (Trees and Forests)**.** An acyclic graph (one not containing any cycles) is called a ***forest***; a connected forest is called a ***tree***. Thus, a forest is a graph whose components are trees. The vertices of degree 1 in a tree are its ***leaves*** (with the caveat that one vertex of degree 1 is given the name of *root*), the others are its *inner vertices*. Every non-trivial tree has a leaf, a fact that often comes handy during proofs about trees.

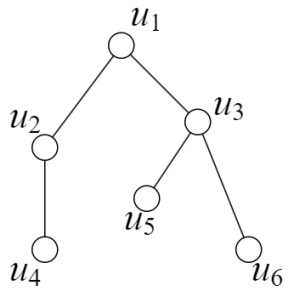


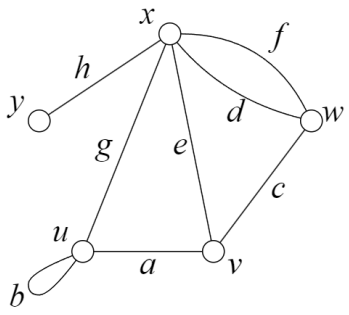
Figure 2.5: An example of tree graph

While the thesis will not focus on trees and their many applications, we found it was nonetheless important to introduce them. Furthermore, in the next chapters we will deal with the concept of graph Laplacian, which first application by Gustav Kirchhoff in 1847 dealt with spanning trees [4].

2.3 Incidence and Adjacency Matrices

Although drawings are a convenient means of specifying graphs, they are not suitable for storing graphs in computers or for applying mathematical methods to study their properties. Thus, we consider two matrices associated with a graph: its incidence matrix and its adjacency matrix.

Definition 2.9 (Incidence and Adjacency Matrix)**.** Let G=(V,E) be a graph. The ***incidence matrix*** of G is the nxm matrix $M_G := (m_{ve})$, where m_ve is the number of times (0,1 or 2) that vertex v and edge e are incident. The ***adjacency matrix*** of G is the nxn matrix $A_G := (a_{uv})$, where a_uv is the number of edges joining vertices u and v, each loop counting as two edges. Both of these matrices are ways of specifying the graph.



	a	b	c	d	e	f	g	h
u	1	2	0	0	0	0	1	0
v	1	0	1	0	1	0	0	0
w	0	0	1	1	0	1	0	0
x	0	0	0	1	1	1	1	1
y	0	0	0	0	0	0	0	1

	u	v	w	x	y
u	2	1	0	1	0
v	1	0	1	1	0
w	0	1	0	2	0
x	1	1	2	0	1
y	0	0	0	1	0

G (a) M_G (b) A_G (c)

Figure 2.6: **(a)** A graph, **(b)** its incidence matrix and **(c)** its adjacency matrix

Since most graphs have many more edges than vertices,the adjacency matrix of a graph is generally much smaller than its incidence matrix and thus requires less storage space. When dealing with simple graphs, we can further compact the information by using lists. For each vertex v, the neighbours of v are listed in some order. A list $(N(v) : v \in V)$ of these lists is called an *adjacency list* of the graph.

2.4 Vertex Degree

The concept of vertex degree is as fundamental as it gets, and a multitude of theorems exist that deal with just this facet of a graph. Here, we will show just the most basic ones as an introduction, knowing well that much more can be built from these simple theorems.

Definition 2.10 (Vertex Degree)**.** The ***degree*** of a vertex v in a graph G, denoted by $d_G(v)$, is the number of edges of G incident with v (with loops counting as two edges). If G is a simple graph, $d_G(v)$ is the number of neighbours of v in G. A vertex of degree zero is called an *isolated vertex*. We will denote by $\delta(G)$ and $\Delta(G)$ the minimum and maximum degrees of the vertices of G, and by d(G) their average degree (that is, $\frac{1}{n}\sum_{v\in V} d_G(v)$).

Theorem 2.11. For any graph G=(V,E),

$$\sum_{v\in V} d_G(v) = 2m$$

where $m = |E|$

Proof. Consider the incidence matrix M of G. The sum of the entries in the row corresponding to vertex v is precisely $d_G(v)$. Therefore $\sum_{v \in V} d_G(v)$ is just the sum of all the entries in M. But this sum is also 2m, because each of the m column sums of M is 2, each edge having two ends. □

Theorem 2.12. In any graph, the number of vertices of odd degree is even

Proof. Consider the equation of theorem 2.11, modulo 2. For each vertex, we have that

$$d(v) = \begin{cases} 1 \,(mod\, 2) & \text{if d(v) is odd} \\ 0 \,(mod\, 2) & \text{if d(v) is even} \end{cases}$$

Thus, modulo 2, the left-hand side is congruent to the number of vertices of odd degree, and the right-hand side is zero. The number of vertices of odd degree is therefore congruent to zero modulo 2 (that is, even). □

Definition 2.13 (Regularity)**.** A graph G is ***k-regular*** if d(v) = k $\forall v \in V$; a regular graph is one that is k-regular for some k. For instance, the complete graph on n vertices is (n-1)-regular, and the complete bipartite graph with k vertices in each part is k-regular.

2.5 Isomorphisms and Automorphisms

Definition 2.14 (Isomorphism)**.** Two graphs G and H are *identical* (written $G = H$) if $V(G) = V(H)$, $E(G) = E(H)$ and $\psi_G = \psi_H$. If two graphs are identical, they can be represented by identical diagrams. However, it is also possible for graphs that are not identical to have what is essentially the same diagram. As we can see in Figure 2.7, the only difference lies in the label of vertices and edges.
In general we say that two graphs G and H are ***isomorphic*** , and write $G \cong H$, if there are bijections $\theta : V(G) \to V(H)$ and $\varphi : E(G) \to E(H)$ such that $\psi_G(e) = uv \iff \psi_H(\varphi(e)) = \theta(u)\theta(v)$. Such a pair of mappings is called an ***isomorphism*** between G and H.

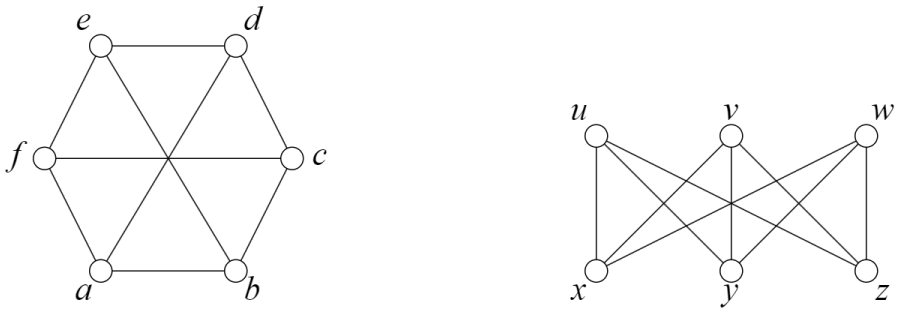
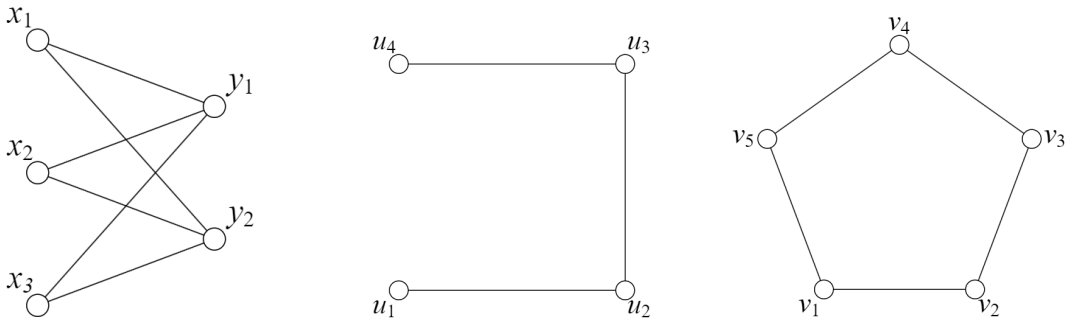


Figure 2.7: Two isomorphic simple graphs

In order to show that two graphs are isomorphic, we must indicate an isomorphism between them. In Figure 2.7, we can show that the two graphs are isomorphic by providing the mapping $\theta := \begin{pmatrix} a & b & c & d & e & f \\ u & x & v & z & w & y \end{pmatrix}$. Isomorphic graphs clearly have the same numbers of vertices and edge, but that alone is not a sufficient condition for isomorphism.

It is natural then to wonder how to test for isomorphism. Suppose we have two simple graphs on n vertices; a brute force approach could need to check for all $n!$ bijections to verify that the two graphs are not isomorphic. This approach is obviously unfeasible, and unfortunately there are no known efficient generally applicable procedures for testing isomorphism. While the details go beyond the scope of this work, we thought interesting to at least mention that in the literature exist efficient isomorphism-testing algorithms for cubic graphs that employ group theory methods [5].

Since we are mainly interested in the structural properties of graphs, we will often omit labels when drawing graphs. We might still assign labels for the purpose of referring to them, such as in proofs. Up to isomorphism, we denote some classes of graphs in unique ways; for example, the complete graph on n vertices is denoted K_n. Given two positive integers m and n, the unique complete bipartite graph with parts of sizes m and n is denoted $K_{m,n}$. Given a positive integer n, the unique path on n vertices is denoted P_n and the unique cycle on n vertices is denoted C_n.



$K_{3,2}$ **(a)** P_4 **(b)** C_5 **(c)**

Figure 2.8: (a) A complete bipartite graph, **(b)** a path and **(c)** a cycle

Definition 2.15 (Automorphism)**.** An ***automorphism*** of a graph is an isomorphism of the graph to itself. In the case of a simple graph, it is just a permutation α of its vertex set which preserves adjacency: if uv is an edge then so is $\alpha(u)\alpha(v)$.

The automorphisms of a graph reflect its symmetries. For example, if u and v are two vertices of a simple graph, and if there is an automorphism α which maps u to v, then u and v are alike in the graph and are referred to as *similar* vertices.

Graphs in which all vertices are similar (such as the complete graph K_n or the complete bipartite graph $K_{n,n}$) are called ***vertex-transitive***. Graphs in which no two vertices are similar are called ***asymmetric***, and in this case they have only the identity permutation as automorphism.

In Figure 2.9 we give three drawings of the *Petersen graph*, often cited due to its many properties.

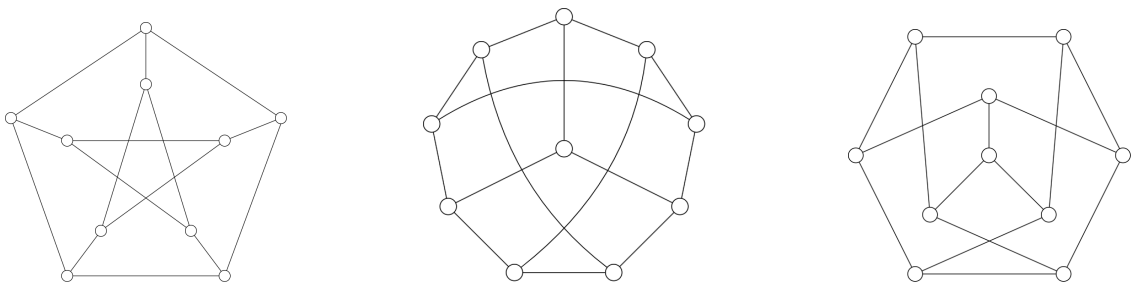


Figure 2.9: The first drawing shows that vertices of the outer pentagon are similar under rotational simmetry, as are the inner vertices. In the third drawing the vertices of the outer hexagon are similar under reflective or rotational simmetry. Combining these two observations we conclude that all ten vertices of the Petersen graph are similar, and the graph is thus vertex-transitive.

2.6 Planar Graphs

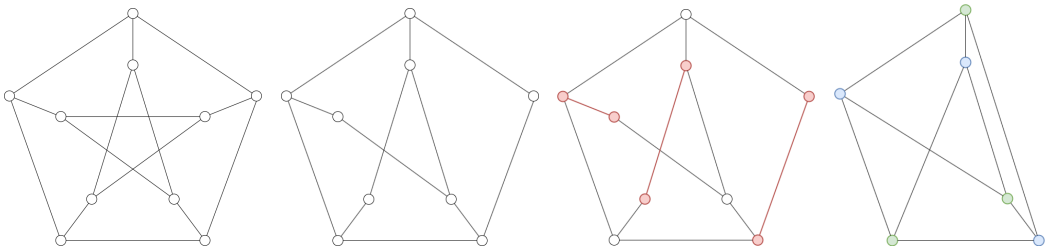
Now that we have given some examples and have seen how the same graph can be drawn in multiple ways, one question that might come natural is whether a graph can be drawn without intersecting the edges.

Definition 2.16 (Planar Graph)**.** A graph $G = (V, E)$ is ***planar*** if there exists an embedding of G in the plane such that no edges of E intersect, except possibly at a common vertex.

Theorem 2.17 (Kuratowski's Theorem)**.** Recall that the subdivision of an edge $e = \{uv\}$ generates a new vertex w and the two edges $e_1 = \{uw\}$ and $e_2 = \{wv\}$; subsequently, the subdivision of a graph G is a graph resulting from the subdivision of edges in G.
A finite graph is planar if and only if it does not contain a subgraph that is a subdivision of K_5 or $K_{3,3}$.

Proof. The proof can be found in the original work from Kuratowski [6]. □

Example 2.18. The Petersen graph is not planar, as it contains a subgraph that is a subdivision of $K_{3,3}$, as we can see in the following figure:



We can see that by choosing the appropriate subgraph and contracting the right edges, we obtain the complete bipartite graph $K_{3,3}$

2.7 Constructing Graphs from Other Graphs

Suppose we have two graphs G and H; a new graph can then be defined in various ways. For ease of notation, we assume that G and H are simple, so that each edge is an unordered pair of vertices.

Definition 2.19 (Union and Intersection)**.** Two graphs are *disjoint* if they have no vertex in common, and *edge − disjoint* if they have no edge in common. The most basic ways to combine graphs are by union and by intersection.
The ***union*** of G and H is the graph $G \cup H$ with vertex set $V(G) \cup V(H)$ and edge set $E(G) \cup E(H)$. If G and H are disjoint, we refer to their union as a *disjoint union* and denote it by $G + H$. This operation is associative and commutative, and can be extended to an arbitrary number of graphs. It can be seen that every graph G may be expressed uniquely as a disjoint union of connected graphs. These graphs are called the *connected components* (or just *components*) of G, and we denote the number of connected components of G by $c(G)$.
The ***intersection*** of G and H is the graph $G \cap H$ and is defined analogously, with the caveat that when G and H are disjoint, their intersection is the null graph.

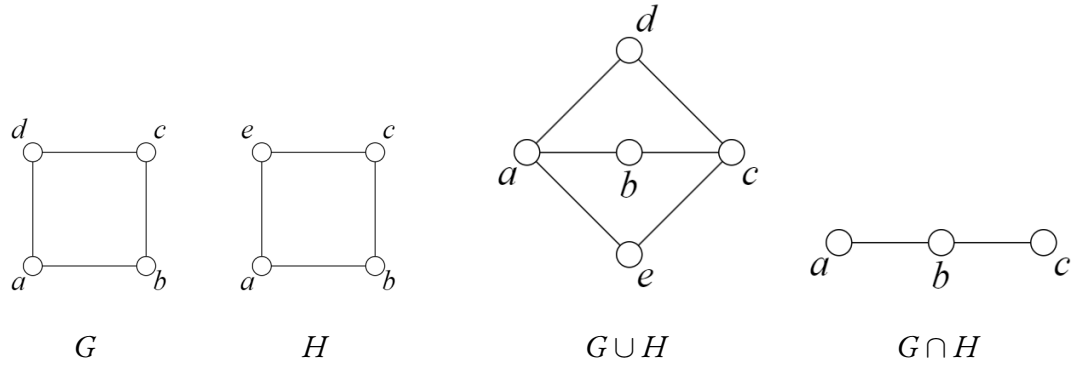


Figure 2.11: Union and intersection of two graphs

Definition 2.20 (Cartesian Product)**.** Another way to construct a graph is by way of products between graphs. One such operation is the ***cartesian product***, that in case of two simple graphs G and H is the graph $G \square H$ whose vertex set is $V(G) \times V(H)$ and whose edge set is the set of all pairs $(u_1, v_1)(u_2, v_2)$ such that either $u_1u_2 \in E(G)$ and $v_1 = v_2$, or $v_1v_2 \in E(G)$ and $u_1 = u_2$. Thus, for each edge u_1u_2 of G and each edge v_1v_2 of H, there are four edges in $G \square H$.

The cartesian product $P_m \square P_n$ of two paths is the $(mxn) - grid$. For $n \geq 3$, the cartesian product $C_n \square K_2$ is a polyhedral graph, the $n - prism$; the 3-prism, 4-prism and 5-prism are commonly called *triangular prism*, *cube* and *pentagonal prism* respectively. In Figure 2.12 and Figure 2.13 we show some examples of cartesian products between common graphs.

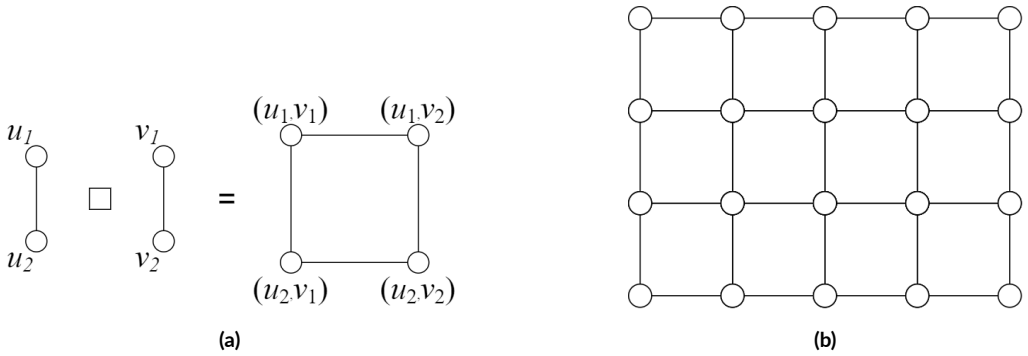


Figure 2.12: (a) The product $K_2 \square K_2$ and **(b)** the product $P_4 \square P_5$, the $(4x5)$-grid

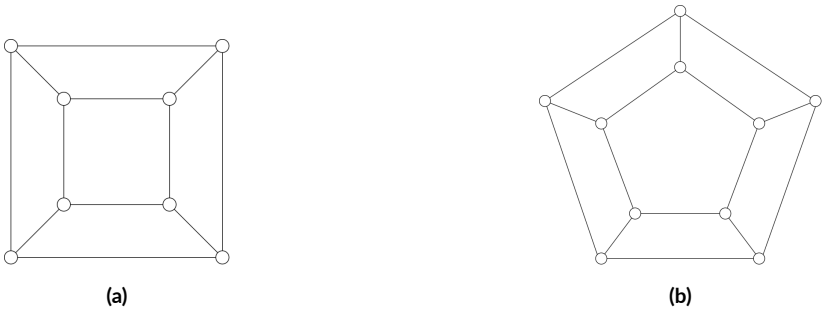


Figure 2.13: (a) The cube and **(b)** the pentagonal prism

2.8 Directed Graphs

Up until now we talked about graphs as just a couple of sets representing vertices and edges, and while many problems can be approached by this kind of graph formulation, sometimes it is not enough. For example, when dealing with traffic flow, a graph of the network is not very useful; we would like to know which roads are one-way and in which direction the traffic is permitted. To deal with these kinds of problems, we need a graph in which each edge has an assigned orientation.

Definition 2.21 (Directed Graph)**.** A ***directed graph*** D is an ordered pair $(V(D), E(D))$ consisting of a set $V := V(D)$ of vertices and a set $E := E(D)$ of arcs, together with an incidence function ψ_D that associates each arc of D to an *ordered pair* of vertices of D. If a is an arc and $\psi_D(a) = (u, v)$, then a is said to *join* u and v; the vertex u will be called the *tail* of a, while the vertex v will be called the *head* of a.

All concepts that are valid for graphs also apply to directed graphs. For example, the degree of

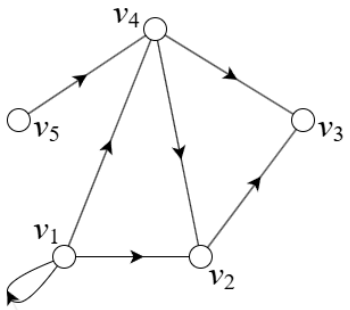
a vertex v in a directed graph D is simply the degree of v in the underlying graph G of D (that is, the graph obtained by replacing arcs with undirected edges). In the same vein, a directed graph is connected if the underlying graph is. However, there are some concepts in which orientations play an essential role.

Definition 2.22 (In-degree and Out-degree)**.** Let $D = (V, E)$ be a directed graph, and let $v \in V$. The ***in-degree*** of v is the number of arcs that have v as head, and is denoted by $d_D^-(v)$. Similarly, the ***out-degree*** of v is the number of arcs that have v as tail, and is denoted by $d_D^+(v)$.

Definition 2.23 (Adjacency Matrix for Directed Graphs)**.** While many more definitions can be given about directed graphs, we will focus on the one example that is most likely to help in the following chapters: the adjacency matrix of a directed graph. Let $G = (V, E)$ be a directed graph, and let $V = \{v_1, ..., v_n\}$. The ***adjacency matrix*** of D is the $n \times n$ matrix A_D where

$$(A_D)_{i,j} = \begin{cases} 1 & \text{if there is a directed edge from i to j} \\ 0 & \text{otherwise} \end{cases}$$

Example 2.24. In the following figure, we give an example of a directed graph and the relative adjacency matrix.



D

(a)

$$\begin{bmatrix} 1 & 1 & 0 & 1 & 0 \\ 0 & 0 & 1 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 \\ 0 & 1 & 1 & 0 & 0 \\ 0 & 0 & 0 & 1 & 0 \end{bmatrix}$$

A_D

(b)

(a) A directed graph and **(b)** its adjacency matrix

2.9 Hypergraphs

Since in the following chapters we will deal with a generalization of the concept of graphs, we offer a small preview in order to ease the reader into this topic.

Definition 2.25 (Hypergraph)**.** A ***set system*** is an ordered pair $(V, \mathcal{F})$ where V is a set of elements and $\mathcal{F}$ is a family of subsets of V. When $\mathcal{F}$ consists of pairs of elements of V, the set system $(V, \mathcal{F})$ is a loopless graph. Set systems can thus be thought of as a generalization of graphs, and are usually referred to as ***hypergraphs***. The elements of V are then called the *vertices* of the hypergraph, and the elements of $\mathcal{F}$ its *edges* or *hyperedges*. A hypergraph is $k-$*uniform* if each edge is a set of k elements ($k-$*set*).

Example 2.26. In this example, we have $V = \{v_1, v_2, v_3, v_4, v_5\}$ as the set of vertices and $\mathcal{F} = \{\{v_1, v_2, v_4, v_5\}, \{v_2, v_3, v_4\}, \{v_3, v_4\}\}$ as the set of hyperedges.

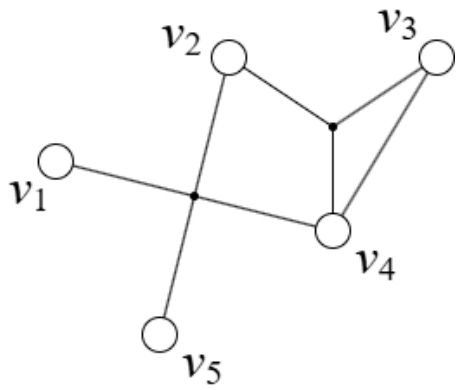


Figure 2.15: The hyperedges can connect more than two vertices at the same time, contrary to the dyadic relationships on which graphs are based on.

In chapter 4 we will see how there exists another similar generalization of graphs, the simplicial complex. Both structures replace the set of edges with something more general, but in the case of the simplicial complexes their additional constraints will be helpful to develop higher-order concepts.

3

Graph Laplacian

When talking about Laplacian, the first thing that comes to mind is the Laplace operator, especially dear to physicists. In the continuous domain, the Laplacian plays a central role in modeling heat flow, gravitational potential and many others. While the introduction of this operator was due to Laplace in the 18th century, only in the following centuries a discrete analogous was first approximately theorized (Kirchhoff, 1847) and later on properly exploited (Fiedler, 1973). In this chapter we will introduce the concept of graph Laplacian and see how, especially through its eigenvalues and eigenvectors, it proves its worth as a powerful and versatile tool, allowing us to employ techniques and algorithms in many different fields. For some definitions and examples, we followed the lecture notes of professor Wolfgang Erb [7].

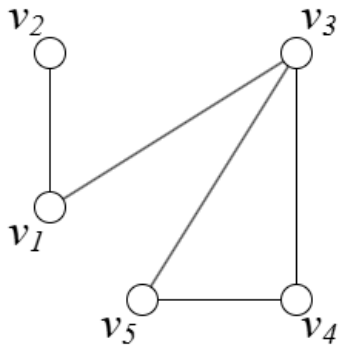
3.1 Graph Laplacian

Given a simple graph G with n vertices $v_1, \ldots, v_n$, consider its Degree matrix D and its Adjacency matrix A. The Laplacian matrix L is then defined as

$$L = D - A$$

or element-wise as

$$L_{i,j} = \begin{cases} d(v_i) & \text{if i = j} \\ -1 & \text{if i} \neq j \text{ and } \mathrm{v}_i \text{ is adjacent to } \mathrm{v}_j \\ 0 & \text{otherwise} \end{cases} \qquad \text{or even} \quad L_{i,j} = \begin{cases} D_{i,i} & \text{if } i = j \\ -A_{i,j} & \text{if } i \neq j \end{cases}$$



G

(a)

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

D_G

(b)

$$\begin{bmatrix} 0 & 1 & 1 & 0 & 0 \\ 1 & 0 & 0 & 0 & 0 \\ 1 & 0 & 0 & 1 & 1 \\ 0 & 0 & 1 & 0 & 1 \\ 0 & 0 & 1 & 1 & 0 \end{bmatrix}$$

A_G

(c)

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

$L_G = D_G - A_G$

(d)

Figure 3.1: (a) A graph, **(b)** its degree matrix, **(c)** its adjacency matrix and **(d)** its Laplacian matrix

3.1.1 Some Basic Properties

Let $x \in \mathcal{L}(V)$. We can write the operation Lx as

$$Lx(v_i) = D_{i,i}x(v_i) - \sum_{j=1}^{n} A_{i,j}x(v_j) = \sum_{j=1}^{n} A_{i,j}x(v_i)x(v_j)$$

Theorem 3.1. For the graph Laplacian L of a graph $G = (V, E)$, we have that:

1. L is symmetric.

2. All the rows and columns sum up to 0. Writing it as $Le = 0$ for $e = (1, ..., 1)^T$, it implies that L has an eigenvalue $\lambda_1 = 0$ with eigenvector $e \in \mathbb{R}^n$.
3. For every $x \in \mathbb{R}^n$, we have $x^T Lx = \frac{1}{2} \sum_{(v_i,v_j)\in E} A_{i,j}(x(v_i) - x(v_j))^2$. This implies that L is positive semidefinite.

Proof. Properties 1) and 2) follow from the definition. We can prove 3) by observing that

$$\begin{aligned}
\sum_{i,j=1}^{n} A_{i,j}(x(v_i) - x(v_j))^2 &= \sum_{i,j=1}^{n} A_{i,j}x(v_i)^2 + \sum_{i,j=1}^{n} A_{i,j}x(v_j)^2 - 2\sum_{i,j=1}^{n} A_{i,j}x(v_i)x(v_j) \\
&= \sum_{i=1}^{n} D_{i,i}x(v_i)^2 + \sum_{j=1}^{n} D_{j,j}x(v_j)^2 - 2\sum_{i,j=1}^{n} A_{i,j}x(v_i)x(v_j) \\
&= 2x^T Dx - 2x^T Ax \\
&= 2x^T(D - A)x \\
&= 2x^T Lx
\end{aligned}$$

This concludes the proof. □

Theorem 3.2. The multiplicity of the zero eigenvalue of the Laplacian L is equal to the number of connected components of graph G.

Proof. We will first show that the number of zero eigenvalues is at least the number of connected components of G. To do so, assume that G has k connected components, corresponding to the partition of V into disjoint sets $V_1, ..., V_k$. Define then k vectors $w_1, ..., w_k$ such that

$$w_i(v_j) = \begin{cases} \frac{1}{\sqrt{|V_i|}} & \text{if } v_j \in V_i \\ 0 & \text{otherwise.} \end{cases}$$

For $i \in \{1, ..., k\}$ we have $\|w_i\|_2 = 1$, $\langle w_i, w_j \rangle = 0$ for $i \neq j$ and $Lw_i = 0$. Thus, there is a set of k orthonormal vectors that are all eigenvectors of L (with respect to the eigenvalue 0). To see that the number of 0 eigenvalues is at most the number of connected components of G, consider $x^T Lx = \frac{1}{2} \sum_{(v_i,v_j)\in E} A_{i,j}(x(v_i) - x(v_j))^2$. This quantity can only be zero if x is constant on every connected component. We ask whether there is another vector y, orthogonal to $w_1, ..., w_k$ that is also a 0 eigenvector. Any eigenvector y must be non-zero on some node; we assume that this holds on a node in V_i and thus non-zero and constant on all nodes in V_i. In this case y cannot be orthogonal to w_i, proving that there cannot be more than k eigenvectors with eigenvalue 0. □

3.1.2 Laplacian eigenvectors

We have seen that the Laplacian L is symmetric, and thus its eigenvalues can be ordered as $0 = \lambda_1 \leq \lambda_2 \leq ... \leq \lambda_n$. The spectral decomposition for symmetric matrices yields the following result:

Theorem 3.3. Given the graph Laplacian L, there exists a set $\{u_1, ..., u_n\}$ of real-valued orthonormal eigenvectors with respect to the eigenvalues $0 = \lambda_1 \leq ... \leq \lambda_n$. Furthermore, this set is an orthonormal basis of the space $\mathcal{L}(V)$ endowed with the inner product $\langle x, y \rangle = \sum_{i=1}^{n} x(v_i)\overline{y(v_i)}$.

The previous spectral theorem for the Laplacian tells us that we have a spectral decomposition of the form $L = U\Lambda U^T$, where $\Lambda = diag(\lambda_1, ..., \lambda_n)$ is a diagonal matrix with the ordered eigenvalues of L on the diagonal, and U is a unitary matrix given by $U = [u_1, ..., u_n]$.

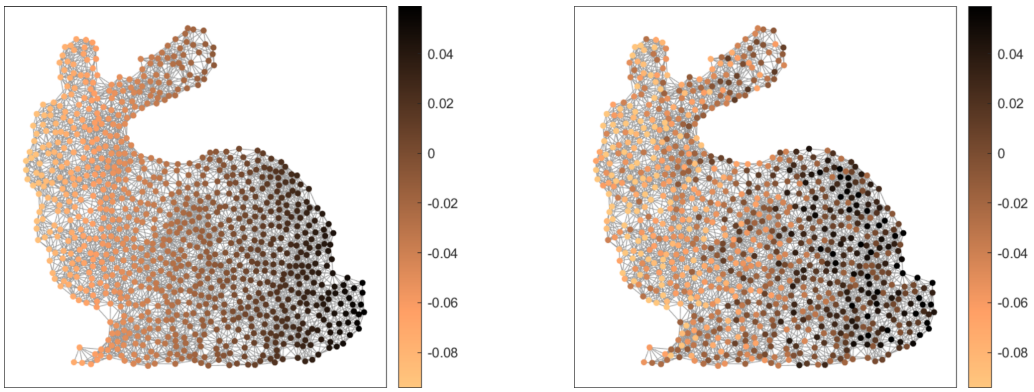
Now that we laid the necessary theoretical foundations, we will give some examples of the possible applications of the graph Laplacian, particularly through the use of the eigenvalues and eigenvectors of the Laplacian matrix. Some concepts will be expanded on and others will be introduced as needed, but we will see how powerful and versatile this tool can be, while exploiting the relatively simple structure of the graph.

3.2 Denoising of graph signals

Suppose we have a graph signal, that is, a function that assigns a value to each node in a graph and is commonly represented as a vector. If we have noise on the signal (Figure 3.2) and are interested in reducing it, the Laplacian plays a critical role.

3.2.1 Graph Fourier Transform

Recall that the Fourier transform on $\mathbb{R}$ is $\hat{x}(\omega) = \int_{\mathbb{R}} x(t)e^{-2\pi it\omega}dt$.
The functions $u_\omega(t) = e^{2\pi it\omega}$ are the eigenfunctions of the Laplace operator $-\Delta = -\frac{\partial^2}{\partial t^2}$, that is, $-\Delta u_\omega(t) = \omega^2 u_\omega(t)$. The signal $x(t)$ can then be recovered from the Fourier coefficients $\hat{x}(\omega)$ using the inverse Fourier transform $x(t) = \int_{\mathbb{R}} \hat{x}(\omega)e^{2\pi it\omega}d\omega$.



(a) (b)

Figure 3.2: **(a)**The original signal and **(b)** the noisy signal on the bunny graph

If we are dealing with graphs the idea is to use the eigenvectors u_k of the graph Laplacian L as a Fourier basis and set

$$\widehat{x_k} = \langle x, u_k \rangle = \sum_{i=1}^{n} x(v_i)\overline{u_k(v_i)}$$

The signal x can the be recovered by the sum

$$x(v_i) = \sum_{k=1}^{n} \widehat{x_k} u_k(v_i)$$

If we consider the spectral decomposition of L as $L = U\Lambda U^T$, we can then write the Graph Fourier transform of a signal x as $\hat{x} = U^T x$, with the k-th entry $\widehat{x_k} = u_k^T x = \langle x, u_k \rangle$ The inverse Fourier transform is then $x = U\hat{x}$

3.2.2 Convolution on graphs

Recall that the convolution operator $*$ on $\mathbb{R}$ is defined as $(x * y)(s) = \int_{\mathbb{R}} x(t)\, y(s-t)\, dt$.
In the Fourier domain, we also have that $\widehat{x * y}\,(\omega) = \hat{x}(\omega)\,\hat{y}(\omega)$.
While there is no direct match in the graph case, the idea is to define convolution via graph Fourier transform:

$$\widehat{x * y}_k = \widehat{x}_k \widehat{y}_k$$

We then define the graph convolution as

$$x * y := U\,diag(\hat{x})\,\hat{y} = U\,diag(\hat{x})\,U^T y$$

and also define the convolution matrix $C_x \in \mathbb{R}^{n\times n}$ as

$$C_x = U\,diag(\hat{x})\,U^T$$

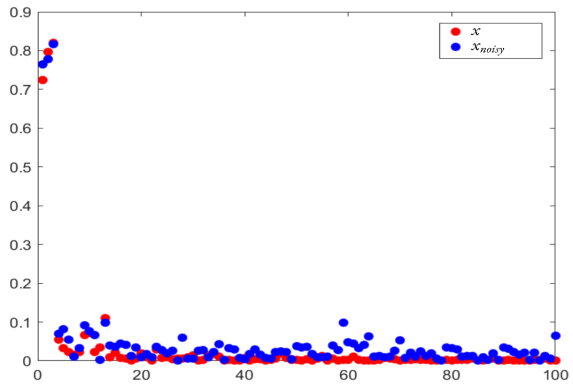
3.2.3 Filters

We can now use the graph convolution to define filters for signals on graphs. For a filter $f \in \mathcal{L}(V)$ and a signal $x \in \mathcal{L}(V)$, the filtered signal $f * x$ is given by

$$(f * x)(v_i) = C_f x(v_i) = \sum_{k=1}^{n} \widehat{f}_k\, \widehat{x}_k\, u_k(v_i)$$

Definition 3.4 (Filters). A ***high-pass filter*** is a function f with $\widehat{f}_k = 0$ for $k < N < n$. Such a filter preserves the larger eigenvalues of L (the "higher frequencies") and enhances local variations; this is useful, for example, for anomaly detection.
A ***low-pass filter*** is a function f with $\widehat{f}_k = 0$ for $k \geq N < n$. Such a filter preserves the smaller eigenvalues of L (the "lower frequencies") and smooths the signal on the graph; this is good, among other things, for denoising.

Example 3.5. Consider once again the case of the bunny graph of Figure 3.2. Let us call x the original signal and x_{noisy} the noisy signal. We start by computing the graph Fourier transform of x and x_{noisy} and we consider the first 100 Fourier coefficients of both signals.

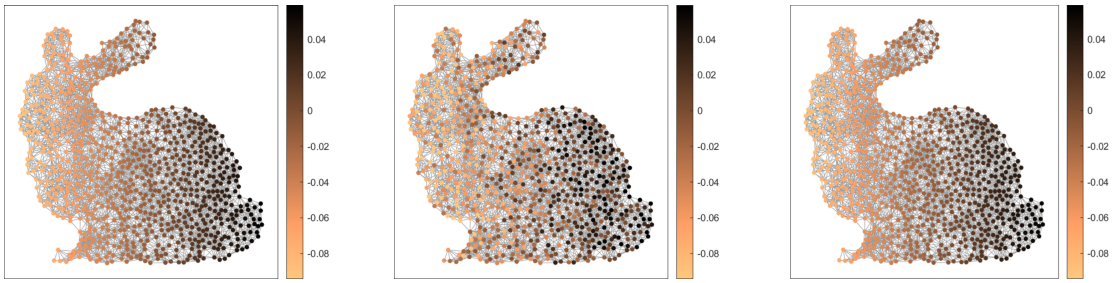


First 100 Fourier coefficients of x and x_{noisy}

We notice that around $k \approx 15$ the frequency components $(\hat{x}_{noisy})_k$ are larger than the components $\hat{x}_k$. Thus, we apply a low-pass filter f for frequencies with k>20 that cuts off all frequencies with k larger than 20. What we get is a denoised signal $x_{denoised} = f * x_{noisy}$, whose components are

$$(\hat{x}_{denoised})_k = \begin{cases} \hat{x}_k & \text{for } k \leq 20 \\ 0 & \text{for } k > 20 \end{cases}$$

What we get in the end is the following result:



(a) **(b)** **(c)**

(a) The original signal, **(b)** the noisy signal, and **(c)** the denoised signal on the bunny graph

3.3 Spectral Clustering

When dealing with graphs, we can use the spectrum of the Laplacian to capture the structure of the data and divide the nodes into clusters in such a way that nodes within the same cluster are "more similar" to each other.

3.3.1 Similarity Graphs

Suppose we have a data set comprised of n elements, and the distances, directions or relations between these elements might be relevant. To properly model these dependencies, we can construct a graph in which the vertices $v_1, \ldots, v_n$ represent the data points: that takes the name of ***similarity graph***.

Aside from the case in which the edges between nodes are previously given, there is a multitude of ways to build a similarity graph. Some examples are:

- A ***complete*** graph K_n; this would then result in a dense graph Laplacian and might be too computationally expensive.

- The ***k-nearest neighbour*** graph (or "kNN graph"), in which we create an edge $e = (v_i, v_j)$ if v_i is among the k vertices nearest to v_j or if v_j is among the k vertices nearest to v_i.
- The ε ***- ball*** graph, built by connecting all vertices whose pairwise distances are smaller than a previously chosen $\varepsilon > 0$.

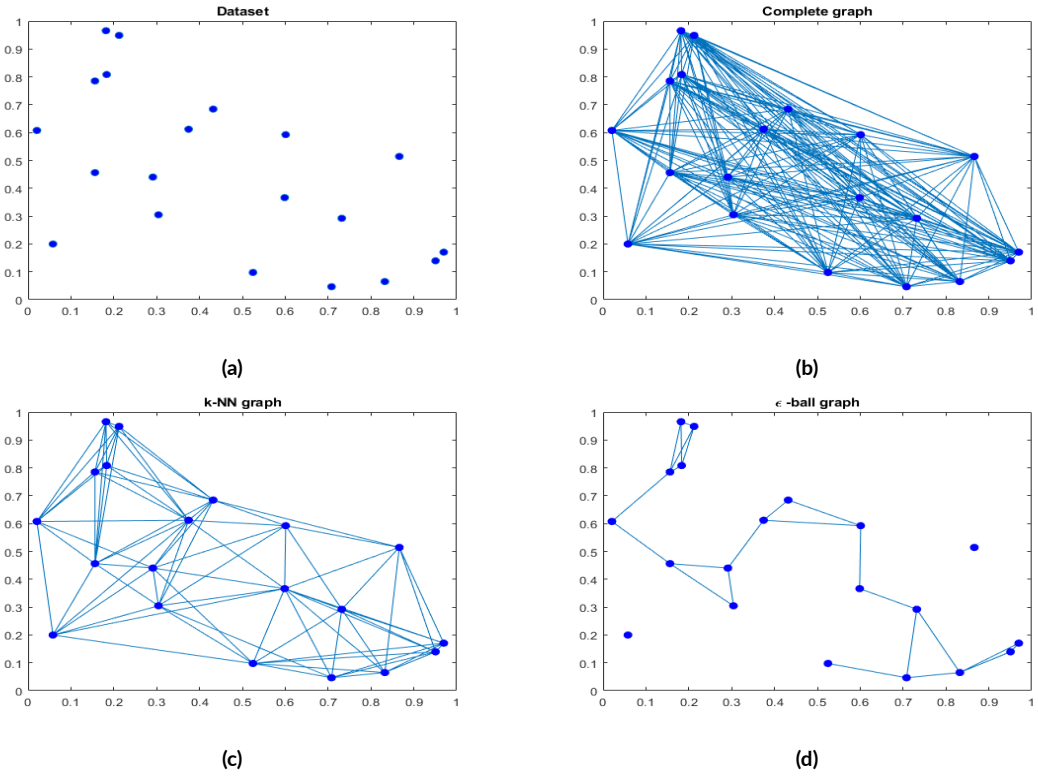


Figure 3.5: **(a)**The dataset and **(b), (c), (d)** three possible similarity graphs

A thing to note is that the choice of which graph to build, as well as which parameters to choose, is inherently subjective. A good knowledge of the domain is thus expected to achieve satisfying results.

3.3.2 Idea behind the clustering

As we have seen in *Theorem* 3.2, the number of connected components of a graph is equal to the multiplicity of the eigenvalue $\lambda_1 = 0$. Furthermore, the eigenvectors corresponding to λ_1 are piecewise constant on the connected components. The idea is that if data points are

strongly clustered a corresponding similarity graph could be split into a number k of connected components, and the eigenvectors corresponding to the k smallest eigenvalues would be almost constant on those components.

While spectral clustering doesn't make assumptions on the form of the clusters, it strongly depends on the similarity graph we build from the data.

3.3.3 Example of Clustering Algorithm

While a thorough analysis of the clustering algorithms is beyond the scope of this work, we give an example (in pseudocode) of one such algorithm by Shi and Malik [8] :

Algorithm 3.1 Shi-Malik Normalized Spectral Clustering

Data: A point set $V = \{v_1, \ldots, v_n\}$ and a number $k \geq 1$ of clusters

1. Construct a similarity graph G from the data.
2. Compute the graph Laplacian $L = D - A$.
3. Compute the k smallest eigenvectors $u_1, \ldots, u_k$ given by

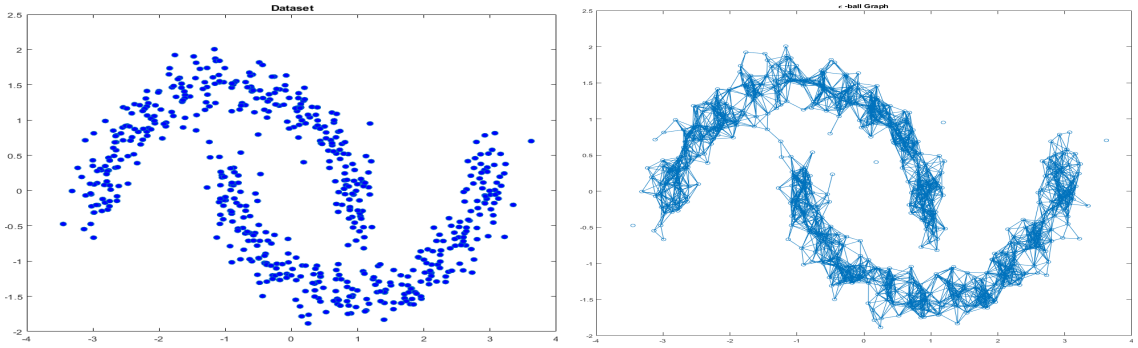
$$Lu_i = \lambda_i D u_i \,, \quad i \in \{1, \ldots, k\}$$

   and store them in a matrix $U_k = \{u_1, \ldots, u_k\}$.
4. Let $y_i \in \mathbb{R}, i \in \{1, \ldots, n\}$ be the vector corresponding to the i-th row of U_k. Cluster the set $\{y_1, \ldots, y_n\} \subset \mathbb{R}^k$ with the *k-means algorithm* into the clusters $C_1, \ldots, C_k$.

Result: Clusters $A_1, \ldots, A_k$ with $A_i = \{U_j \in U : y_j \in C_i\}$

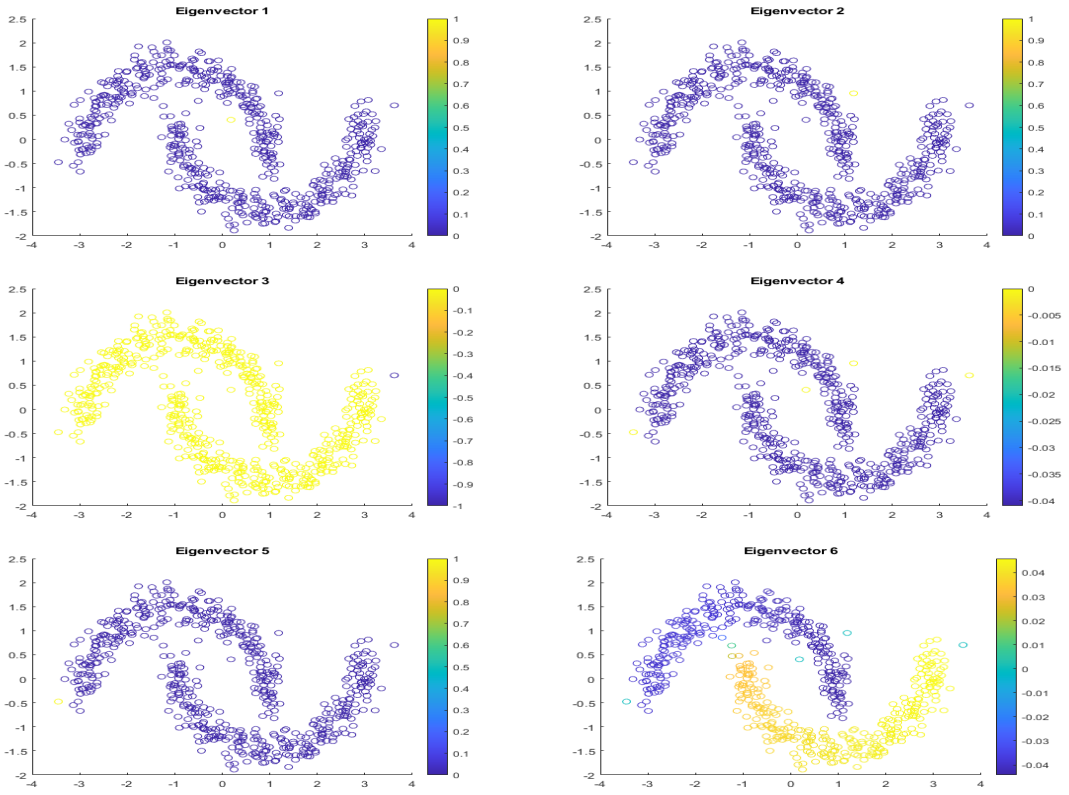
Here we mentioned the k-means algorithm, which partitions a set of data points into k clusters such that each point belongs to the cluster with the nearest mean. More informations can be found in the work of MacQueen [9].

Example 3.6. In this example we will use the so-called "two-moons" dataset, a synthetic dataset often used in examples and tests. As we can see from the following figure, the points are quite clearly separated in an upper and a lower moon. A thing to note is that one of the connected components includes the majority of the nodes, in a way that the two moons are part of the same component. We decide to build an ε-ball similarity graph (with $\varepsilon = 0.3$):



(a) The "two-moons" dataset and **(b)** its ε-ball similarity graph

The graph we built has 5 connected components, and thus the eigenvalue λ_1 has multiplicity 5. The eigenvector relative to the 6-th smaller eigenvalue should then tell us something and help us cluster the data.



The values of the first 6 eigenvectors

We can see that the first five eigenvectors show piecewise constant values on the connected components; this doesn't help at all with the clustering. The sixth eigenvector, on the other hand, shows a clear distinction between the two moons: negative values for the upper moon and positive values for the lower moon.

3.4 Dimensionality Reduction

In many fields we are often dealing with high-dimensional data and for a number of reasons we would like to reduce the number of variables, all the while preserving the underlying structure of the data and the information it brings. To give the idea in a more formalized way, suppose we have a set of points in a high dimensional Euclidean space sampled from a manifold: $v_1, \ldots, v_n \in \mathcal{M} \subset \mathbb{R}^d$. We want to find a set of vectors in a low-dimensional Euclidean space, $y_1, \ldots, y_n \in \mathbb{R}^k$ for some $k < d$ such that y_i "represents" v_i.

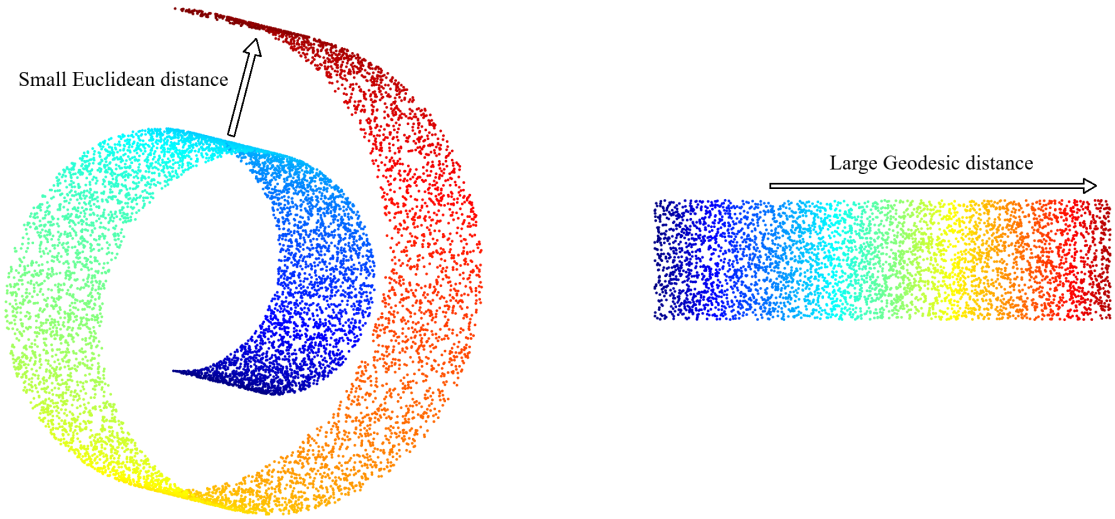


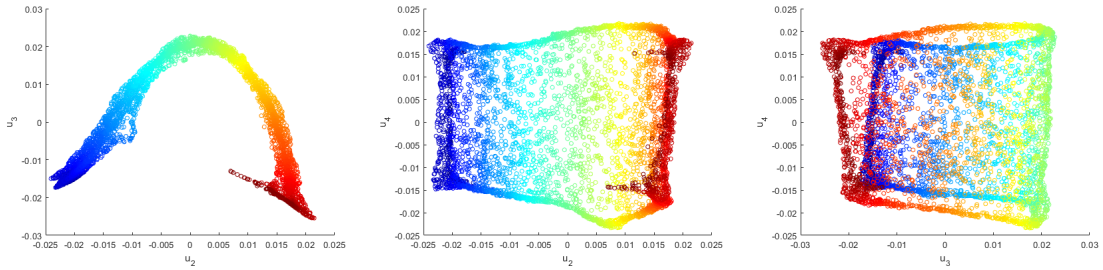
Figure 3.8: An example of dimensionality reduction of the so-called *Swiss Roll*

3.4.1 Embedding graph data

To obtain a lower-dimension embedding of the nodes of a connected graph G with adjacency matrix A and degree matrix D, we consider the so-called *random-walk Laplacian*, defined as $L_{rw} = D^{-1}L$. The random-walk Laplacian is a normalized version of the standard Laplacian we have come to know, but considers relative degrees of nodes instead of their degree (and is thus connected to transition probabilities and random walks, hence the name).

If we now consider the eigenvectors $u_2, \ldots, u_{k+1}$ of L_{rw}, we can form an embedding matrix $X = [u_2, \ldots, u_{k+1}] \in \mathbb{R}^{n \times k}$. The rows of X are the new coordinates of the original data points $v_i \in \mathbb{R}^d$.

Example 3.7. Consider once again the swiss roll: a 3D data set that intrinsically lies on a 2D manifold. We build a similarity graph for the data set and then plot the eigenvectors of L_{rw} against each other:



Eigenvectors plotted against each other

We can see that using the eigenvectors u_2 and u_4 we obtain an embedding that captures the underlying structure of our data. The embedding matrix would then be $X = [u_2, u_4] \in \mathbb{R}^{n \times 2}$.

4

Simplicial Complexes

One of the merits of graphs and a reason for their success is their ease of interpretation and visualization. Relating to data analysis, for instance, one may think of each node of a graph as a single piece of information, and of any edge as a dyadic relation between such data. Thus, at a glance, we can see whether data is connected or not.

Of course, this interpretation is rather trivial, and more refined ideas may be developed considering, for instance, paths in graphs. There is, however, a whole facet of the story that is not detected by the simple use of graphs. We are talking about all the features known as the *topology* of the data, by which we mean the underlying (and often hidden) geometry of point cloud data.

This aspect has come prominently at play in recent years, and has lead to the definition and the study of *high-order* networks, with the aim of identifying stronger relations between points of the cloud [1]. In an unusual but appreciated twist, rather theoretical tools such as persistent homology proved to have an incredibly broad spectrum of applications, from dynamical networks to mathematical biology, and from data analysis to signal processing.

Most of the features of a simplicial complex may be seen as a generalization of properties of graphs; in fact, as we will see, graphs are nothing but a very simple instance of simplicial complexes.

4.1 Δ-Complexes

In order to properly define the simplicial complex and the homology theory it requires, a first step is to define a broader and more generalized class of spaces called Δ-complexes. The definitions used in this section are taken from the work of Allen Hatcher [10].

Definition 4.1 (n-simplex)**.** A ***n-simplex*** is the smallest convex set in a Euclidean space $\mathbb{R}^m$ containing $n+1$ points $v_0, \ldots, v_n$ that do not lie in a hyperplane of dimension less than n. The points v_i are called the **vertices** of the simplex, and the simplex itself is denoted $[v_0, \ldots, v_n]$. The ***standard n-simplex***, whose vertices are the unit vectors among the coordinate axes, is denoted by

$$\Delta^n = \left\{ (t_0, \ldots, t_n) \in \mathbb{R}^{n+1} \mid \sum_i t_i = 1 \text{ and } t_i \geq 0 \ \forall i \right\}$$

When dealing with homology, it will be important to keep track of the order of the vertices of a simplex, so when saying "n-simplex" we will really mean "n-simplex with an ordering of its vertices".

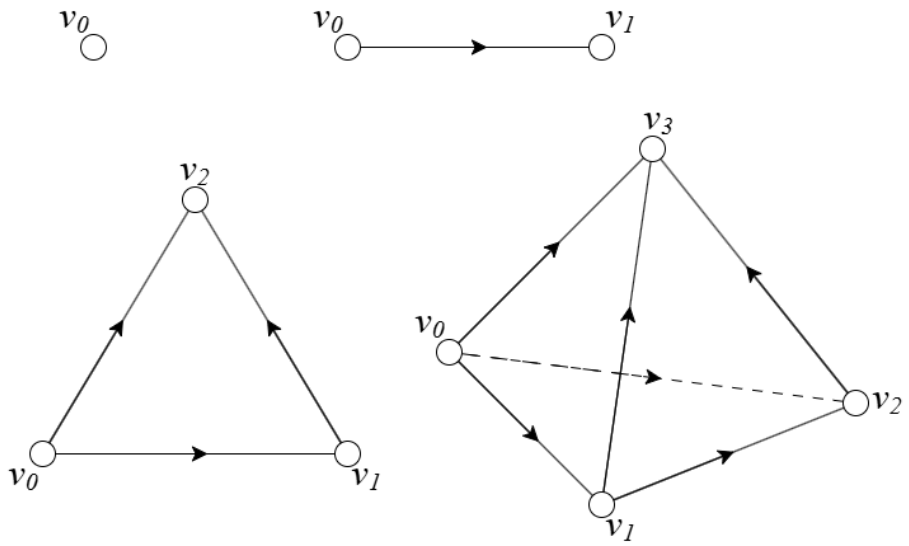


Figure 4.1: n-simplices for n=0,1,2,3

Definition 4.2 (Geometry of a Simplex)**.** If we delete one of the $n+1$ vertices of an n-simplex $[v_0, ..., v_n]$, the remaining n vertices span an $(n-1)$-simplex called a ***face*** of $[v_0, ..., v_n]$. As a convention, the vertices of a face will always be ordered according to their order in the larger simplex.
The union of all the faces of Δ^n is called the ***boundary*** of Δ^n and is written as $\partial\Delta^n$.
The ***open simplex*** $\mathring{\Delta}^n$ is $\Delta^n - \partial\Delta^n$, the interior of Δ^n.

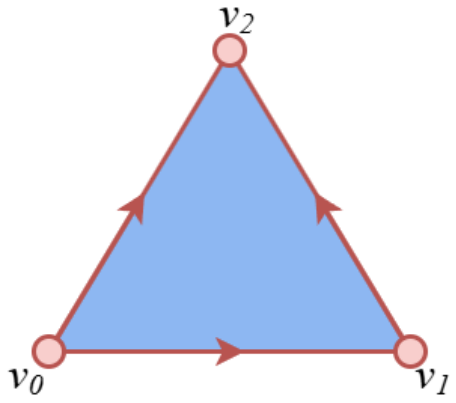


Figure 4.2: In the 2-simplex case, each edge is a face and its border is indicated in red; the interior is indicated in blue.

Definition 4.3 (Δ-complex)**.** A Δ-***complex*** structure on a space X is a collection of maps $\sigma_\alpha : \Delta^n \to X$, with n depending on the index α, such that:

1. The restriction $\sigma_\alpha|\mathring{\Delta}^n$ is injective, and each point of X is in the image of exactly one such restriction.

2. Each restriction of σ_α to a face of Δ^n is one of the maps $\sigma_\beta : \Delta^{n-1} \to X$. Here we are identifying the face of Δ^n with Δ^{n-1} by the canonical linear homeomorphism between them that preserves the ordering of the vertices.

3. A set $A \subset X$ is open $\iff \sigma_\alpha^{-1}$ is open in Δ^n for each σ_α.

In general, Δ-complexes can be built from collections of disjoint simplices by identifying various subsimplices spanned by subsets of the vertices. For example, the Δ-complex structure on a torus can be obtained by identifying pairs of edges of two 2-simplices.

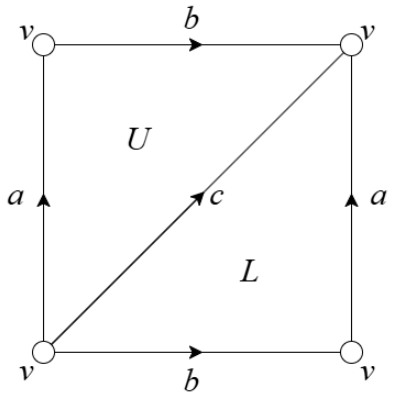


Figure 4.3: Δ-complex representation of a torus

4.2 Simplicial Homology

The goal is now to define the simplicial homology groups of a Δ-complex X.

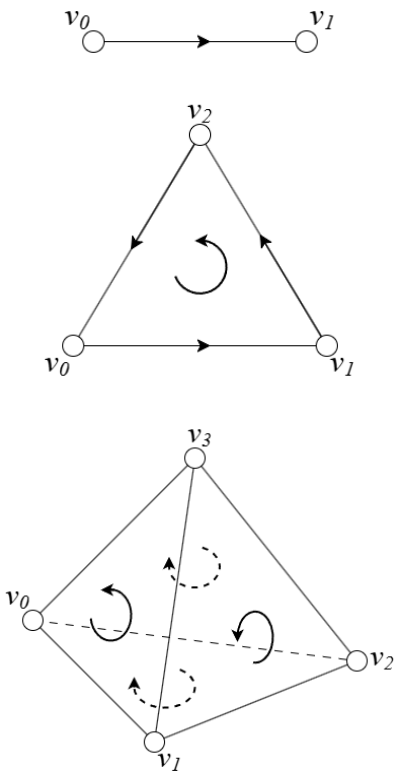
Definition 4.4 (n-chain)**.** Let $\Delta_n(X)$ be the free abelian group with basis the open n-simplices e^n_α of X. Elements of $\Delta_n(X)$, called ***n-chains***, can be written as finite formal sums $\sum_\alpha n_\alpha e^n_\alpha$ with coefficients $n_\alpha \in \mathbb{Z}$. Equivalently, we can write $\sum_\alpha n_\alpha \sigma_\alpha$ where $\sigma_\alpha : \Delta^n \to X$ is the characteristic map of e^n_α, with image the closure of e^n_α as described above. Such a sum can then be thought of as a finite collection, or "*chain*", of n-simplices in X with integer multiplicities (the coefficients n_α).

As we can see in the previous figure, the boundary of the n-simplex $[v_0, \ldots, v_n]$ consists of the various $(n-1)$-dimensional simplices $[v_0, \ldots, \widehat{v_i}, \ldots, v_n]$, where $\widehat{v_i}$ indicates that this vertex is deleted from the sequence. Under this convention, the boundary of $[v_0, \ldots, v_n]$ can then be written as $\sum_i (-1)^i [v_0, \ldots, \widehat{v_i}, \ldots, v_n]$. The signs are conventionally inserted to take orientations into account, so that all the faces of a simplex are coherently oriented.

Definition 4.5 (Boundary Homomorphism)**.** For a general Δ-complex X we can define a ***boundary homomorphism*** $\partial_n : \Delta_n(X) \to \Delta_{n-1}(X)$ by specifying its values on basis elements:

$$\partial_n(\sigma_\alpha) = \sum_i (-1)^i \sigma_\alpha | [v_0, \ldots, \widehat{v_i}, \ldots, v_n]$$

The right side of the equation lies in $\Delta_{n-1}(X)$, since each restriction $\sigma_\alpha | [v_0, \ldots, \widehat{v_i}, \ldots, v_n]$ is the



$$\partial[v_0, v_1] = [v_1] - [v_0]$$

$$\partial[v_0, v_1, v_2] = [v_1, v_2] - [v_0, v_2] + [v_0, v_1]$$

$$\partial[v_0, v_1, v_2, v_3] = [v_1, v_2, v_3] - [v_0, v_2, v_3] + [v_0, v_1, v_3] - [v_0, v_1, v_2]$$

Figure 4.4: n-chains for n= 1,2,3

characteristic map of an $(n-1)$-simplex of X.

Theorem 4.6. The composition $\Delta_n(X) \xrightarrow{\partial_n} \Delta_{n-1}(X) \xrightarrow{\partial n-1} \Delta_{n-2}(X)$ is zero.

Proof. We have $\partial_n(\sigma) = \sum_i (-1)^i \sigma\,|[v_0, ..., \widehat{v_i}, ..., v_n]$, and thus

$$\begin{aligned}\partial_{n-1}\partial_n(\sigma) = &\sum_{j<i} (-1)^i (-1)^j \sigma\,|[v_0, ..., \widehat{v_j}, ..., \widehat{v_i}, ..., v_n] \\ &+ \sum_{j>i} (-1)^i (-1)^{j-1} \sigma\,|[v_0, ..., \widehat{v_i}, ..., \widehat{v_j}, ..., v_n]\end{aligned}$$

Now, if we switch i and j in the second sum, it becomes the negative of the first sum and the total summation equals to zero. □

Example 4.7. Consider the 2-simplex in Figure 4.4. Then:

$$\begin{aligned}\partial_1\partial_2([v_0,v_1,v_2]) &= \partial_1([v_1,v_2]-[v_0,v_2]+[v_0,v_1])\\ &= \partial_1([v_1,v_2]+[v_2,v_0]+[v_0,v_1])\\ &= [v_2]-[v_1]+[v_0]-[v_2]+[v_1]-[v_0]\\ &= 0\end{aligned}$$

4.2.1 A Theoretical Digression

What we have now is a sequence of homomorphisms of abelian groups

$$\cdots \longrightarrow C_{n+1} \xrightarrow{\partial_{n+1}} C_n \xrightarrow{\partial_n} C_{n-1} \longrightarrow \cdots \longrightarrow C_1 \xrightarrow{\partial_1} C_0 \xrightarrow{\partial_0} 0$$

with $\partial_n\partial_{n+1} = 0$ for each n. Such a sequence is called a ***chain complex***. The equation $\partial_n\partial_{n+1} = 0$ is equivalent to the inclusion $Im\,\partial_{n+1} \subset Ker\,\partial_n$; we can thus define the n^{th} ***homology group*** of the chain complex to be the quotient group $H = Ker\,\partial_n / Im\,\partial_{n+1}$. Elements of $Ker\,\partial_n$ are called ***cycles*** and elements of $Im\,\partial_{n+1}$ are called ***boundaries***. Elements of H_n are cosets of $Im\,\partial_{n+1}$ and are called ***homology classes***. Two cycles representing the same homology class are said to be ***homologous***, and this means that their difference is a boundary. When $C_n = \Delta_n(X)$, the homology group will be denoted H_n^Δ and called the n^{th} ***simplicial homology group*** of X.

Example 4.8. Let $X = S^1$, with one vertex v and one edge e. Then $\Delta_0(S^1)$ and $\Delta_1(S^1)$ are both $\mathbb{Z}$ and the boundary map ∂_1 is zero, since $\partial e = v - v$. The groups $\Delta_n(S^1)$ for $n \geq 2$ are 0 since there are no simplices in those dimensions. We have thus

$$H_n^\Delta(S^1) \approx \begin{cases}\mathbb{Z} & \text{for } n = 0,1\\ 0 & \text{for } n \geq 2\end{cases}$$

Generally speaking, if the boundary maps in a chain complex are all zero, then the homology groups of the complex are isomorphic to the chain groups themselves.

Example 4.9. Let $X = T$, the torus with the Δ-complex structure that we previously saw. Here we have one vertex v, three edges a,b and c, and two 2-simplices U and L. As in the previous example, $\partial_1 = 0$ so $H_0^\Delta(T) = \mathbb{Z}$. $\partial_2 U = a + b - c = \partial_2 L$, and $\{a, b, a + b - c\}$ is a basis for $\Delta_1(T)$, so it follows that $H_1^\Delta(T) \approx \mathbb{Z} \oplus \mathbb{Z}$ with basis the homology classes $[a]$ and $[b]$. Since there are no 3-simplices, $H_2^\Delta(T)$ is equal to $Ker\, \partial_2$, which is infinite cyclic generated by $U - L$ since $\partial(pU - qL) = (p + q)(a + b - c) = 0 \iff p = -q$. We have thus

$$H_n^\Delta(T) \approx \begin{cases} \mathbb{Z} \oplus \mathbb{Z} & \text{for } n = 1 \\ \mathbb{Z} & \text{for } n = 0, 2 \\ 0 & \text{for } n \geq 2 \end{cases}$$

Homology is a cornerstone of algebraic topology [10]: it allows us to categorize objects in terms of topological invariants which represent, roughly speaking, n-dimensional "holes" of the underlying manifold. A crucial property of homology is that it is invariant under the action of appropriate mappings, generally known as *homeomorphisms* (which means "similar shape").

From the perspective of homology, then, a torus and a mug are the same object. As a consequence, it is unquestionably convenient to reduce an object to its simplest shape before studying it from the homological point of view. It is here where simplicial complexes display all their power.

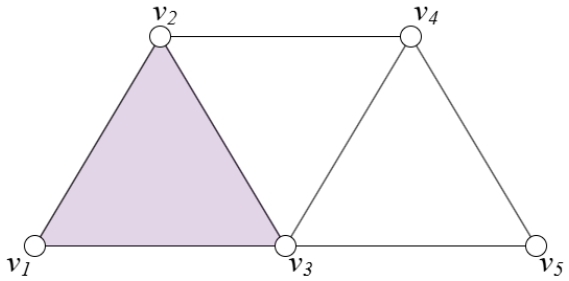
4.3 Simplicial Complex

By leveraging what we previously saw, we could say that simplicial complexes are Δ-complexes whose simplices are uniquely determined by their vertices. This means that each n-simplex has $n + 1$ distinct vertices, and no other n-simplex has the same set of vertices. Nonetheless, let us see a more rigorous definition.

Definition 4.10 (Simplicial Complex)**.** Consider a finite set of vertices $\mathcal{V} = \{1, ..., N\}$. Given an abstract *n-simplex* σ, let $\tau \subset \sigma$ be a *face* of σ. An *abstract* ***simplicial complex*** $\mathcal{K}$ is a finite collection of such simplices such that

- $\tau \subset \sigma, \sigma \in \mathcal{K} \implies \tau \in \mathcal{K}$
- $\{v\} \in \mathcal{K}$ for all $v \in \mathcal{V}$

Example 4.11. Given $\mathcal{V} = \{v_1, ..., v_5\}$, consider the following abstract simplicial complex $\mathcal{K}$:



$$\mathcal{K} = \Big\{\{v_1\}, \{v_2\}, \{v_3\}, \{v_4\}, \{v_5\}, \{v_1, v_2\}, \{v_1, v_3\}, \{v_2, v_3\}$$
$$\{v_2, v_4\}, \{v_3, v_4\}, \{v_3, v_5\}, \{v_4, v_5\}, \{v_1, v_2, v_3\}\Big\}$$

There's a total of five 0-simplices (vertices), seven 1-simplices (edges) and one 2-simplex. The simplicial complex $\mathcal{K}$ has thus dimension 2, as {1,2,3} is the largest simplex in $\mathcal{K}$.

Definition 4.12 (Skeleton)**.** The dimension of an abstract simplicial complex $\mathcal{K}$ is defined as the largest dimension of any simplex in $\mathcal{K}$. The ***n-skeleton*** $\mathcal{K}_n$ of the abstract simplicial complex $\mathcal{K}$ is the union of the simplices of dimensions $0, 1, ..., n$. The $0 - skeleton$ $\mathcal{K}_0 = \mathcal{V}$ are the vertices of $\mathcal{V}$, and the $1 - skeleton$ is the graph associated to the simplicial complex.

4.3.1 Conventions and orientation

We will number the vertices and set $\mathcal{V} = \{1, ..., N\}$ so that the maximal simplex dimension is $N-1$. We use n to refer to a generic n-simplex of arbitrary dimension n (n+1 vertices), whereas we use N as a fixed constant describing the total number of vertices.

To refer to simplices and their faces, as well as to facilitate computations with simplicial complexes, we introduce an orientation for each simplex in $\mathcal{K}$. Consider an n-simplex $\{v_0, ..., v_n\}$, $v_i \in \mathcal{V}$. An *oriented simplex* associated to this n-simplex is denoted by an ordered tuple $\sigma = [v_0, ..., v_n]$. There are now two possible orientations: if two oriented simplices associated to the same underlying simplex differ by an even permutation of the tuple, they are equivalent

(and thus have the same orientation). If they instead differ by an odd permutation of the tuple, they have an opposite orientation. For example, the oriented 1-simplex $[v_i, v_j]$ from v_i to v_j (that is, an edge) has an opposite orientation to $[v_j, v_i]$. From here on, we will say that the oriented n-simplex $\sigma = [v_0, ..., v_n]$ has a positive orientation if the tuple is ordered equivalently to the standard ordering according to increasing vertex labels.

We will use the notation $\sigma_i^{k-1} \subset \sigma_j^k$ to indicate that σ_i^{k-1} is a boundary element of σ_j^k. Given a simplex $\sigma_i^{k-1} \subset \sigma_j^k$, we will use the notation $\sigma_i^{k-1} \sim \sigma_j^k$ to indicate that the orientation of σ_i^{k-1} is coherent with that of σ_j^k, while we will write $\sigma_i^{k-1} \nsim \sigma_j^k$ to indicate that the two have opposite orientations.

Definition 4.13 (Upper and Lower Adjacency)**.** The following definitions come from [11] and will help us in the next chapter to deal with more complex topics. Let $\mathcal{K}$ be a finite simplicial complex. Two n-simplices σ_1, σ_2 in $\mathcal{K}$ are called ***upper adjacent*** (denoted by $\sigma_1 \sim_U \sigma_2$) if both simplices are faces of a $(n+1)$-simplex τ in $\mathcal{K}$, called their ***common upper simplex***. The ***upper degree*** of an n-simplex σ in $\mathcal{K}$ (denoted by $deg_U(\sigma)$) is the number of $(n+1)$-simplices in $\mathcal{K}$ of which σ is a face.
Two n-simplices σ_1, σ_2 are called ***lower adjacent*** (denoted by $\sigma_1 \sim_L \sigma_2$) if both simplices contain a nonempty $(n-1)$-simplex η in $\mathcal{K}$ as a face, called their ***common lower simplex***. The ***lower degree*** of an n-simplex σ in $\mathcal{K}$ (denoted by $deg_L(\sigma)$) is the number of nonempty $(n-1)$-simplices in $\mathcal{K}$ that are faces of σ.

Proposition 4.14 (Uniqueness of Common Upper Simplex)**.** Let $\mathcal{K}$ be a finite simplicial complex, and let σ_1, σ_2 be two distinct n-simplices in $\mathcal{K}$. If σ_1 and σ_2 are upper adjacent, then their common upper $(n+1)$-simplex is unique.

Proof. Suppose τ_1 and τ_2 are both $(n+1)$-simplices in $\mathcal{K}$ containing σ_1 and σ_2 as faces. Then $\sigma_1 \cup \sigma_2 \subseteq \tau_1 \cap \tau_2$, and $\tau_1 \cap \tau_2$ must be a face of both τ_1 and τ_2. Since τ_1 is an $(n+1)$-simplex, the only face of τ_1 that can contain two distinct n-simplices (σ_1 and σ_2) is τ_1 itself. This implies that $\tau_1 = \tau_1 \cap \tau_2$, that is, τ_1 is a face of τ_2. Since τ_1 and τ_2 are both $(n+1)$-simplices, this means that $\tau_1 = \tau_2$. □

Proposition 4.15 (Uniqueness of Common Lower Simplex)**.** Let $\mathcal{K}$ be a finite simplicial complex, and let σ_1, σ_2 be two distinct n-simplices in $\mathcal{K}$. If σ_1 and σ_2 are lower adjacent, then their common lower $(n-1)$-simplex is the intersection of the two simplices. If it exists, it is unique.

Proof. Suppose η is a common lower simplex of σ_1 and σ_2; then $\eta \subseteq \sigma_1 \cap \sigma_2$. From the definition of simplicial complex, we know that $\sigma_1 \cap \sigma_2$ is a face of σ_1, and this face must contain η. Since η is a $(n-1)$-simplex, there can be only two faces of σ_1 that contain η: η and σ_1 itself. If $\sigma_1 \cap \sigma_2 = \sigma_1$, then σ_1 is a face of σ_2, implying that $\sigma_1 = \sigma_2$ and contradicting the assumption that σ_1 and σ_2 are distinct. It follows that $\eta = \sigma_1 \cap \sigma_2$. We have seen that any common lower simplex of two simplices is always the intersection of the two, so any other common lower simplex would be identical to the first, hence the uniqueness. □

5

Hodge Laplacian

One of the most important tasks in mathematics and physics is to understand how information flows through a space, whose geometrical and topological constraints heavily influence these flows. Graphs have been used for a relatively long time to model such spaces, but there are problems that cannot be solved with just such simple abstractions.

The graph Laplacian is widely used to analyze functions on the nodes of a graph, but lacks the same capacity when dealing with edges or surfaces. For example, look at theorem 3.2: it correlates the number of connected components to the multiplicity of the zero eigenvalue of the graph Laplacian. That is surely helpful, but if we stop at the graph-level we cannot gather other informations about the geometrical structure of the space, like the presence of holes. This is just one of the examples that we will see in this chapter and that propelled us to look for a more complex and richer structure, namely the simplicial complex that we previously introduced.

In a parallel to chapter 3, we will introduce the concept of Hodge Laplacian, a generalization to higher-dimensional structures of the graph Laplacian that in the last few years has gained relevance in applied mathematics, especially in topological data analysis. We will look into examples and problems that underline the shortcomings of the graph Laplacian, that become tractable and approachable when seen through the lenses of the Hodge Laplacian. When possible, we will use the same fields of application seen in chapter 3 to better emphasize the utility and necessity of the higher-order operator.

5.1 Hodge Laplacian

Definition 5.1 (Boundary Operator)**.** Given an orientation of the simplicial complex $\mathcal{K}$, the entries of its incidence matrices B_n establish which n-simplices are incident to which $(n-1)$-simplices, akin to the adjacency matrix for graphs. The matrix representation is thus:

$$B_n(i,j) = \begin{cases} 0 & \text{if } \sigma_i^{n-1} \not\subset \sigma_j^n \\ 1 & \text{if } \sigma_i^{n-1} \subset \sigma_j^n \text{ and } \sigma_i^{n-1} \sim \sigma_j^n \\ -1 & \text{if } \sigma_i^{n-1} \subset \sigma_j^n \text{ and } \sigma_i^{n-1} \nsim \sigma_j^n \end{cases}$$

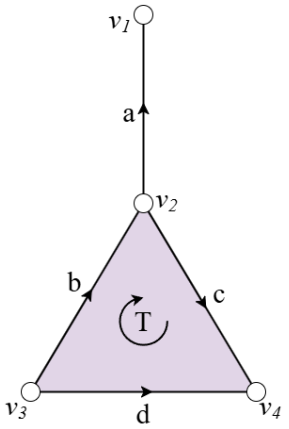
Definition 5.2 (Hodge Laplacian)**.** To gain information on a simplicial complex of order N by using boundary operators we build, for $n = 0, 1, ..., N$, its ***Hodge Laplacians*** [1]:

$$L_n = W_{n+1} B_n^T W_n^{-1} B_n + B_{n+1} W_{n+2} B_{n+1}^T W_{n+1}^{-1}$$

where B_n are the matrix representations of the boundary operators ∂_n, and W_n are diagonal weight matrices. We will often consider $W_n = \mathbb{I}_n$, in which case:

$$L_n = B_n^T B_n + B_{n+1} B_{n+1}^T$$

Example 5.3. Let $\mathcal{K}$ be the simplicial complex of order 2 given in the following figure:



An example of a simplicial complex $\mathcal{K}$ with 4 vertices (v_1, v_2, v_3, v_4), 4 edges (a, b, c, d) and one 2-simplex (T)

By following the previous definitions, we can compute the boundary maps:

$$B_1 = \begin{bmatrix} 1 & 0 & 0 & 0 \\ -1 & 1 & -1 & 0 \\ 0 & -1 & 0 & -1 \\ 0 & 0 & 1 & 1 \end{bmatrix} \qquad B_2 = \begin{bmatrix} 0 \\ 1 \\ 1 \\ -1 \end{bmatrix}$$

Then

$$L_0 = B_0^T B_0 + B_1 B_1^T = 0 + \begin{bmatrix} 1 & -1 & 0 & 0 \\ -1 & 3 & -1 & -1 \\ 0 & -1 & 2 & -1 \\ 0 & -1 & -1 & 2 \end{bmatrix}$$

and

$$\begin{aligned} L_1 = B_1^T B_1 + B_2 B_2^T &= \begin{bmatrix} 1 & 0 & 0 & 0 \\ -1 & 1 & -1 & 0 \\ 0 & -1 & 0 & -1 \\ 0 & 0 & 1 & 1 \end{bmatrix} \begin{bmatrix} 1 & 0 & 0 & 0 \\ -1 & 1 & -1 & 0 \\ 0 & -1 & 0 & -1 \\ 0 & 0 & 1 & 1 \end{bmatrix} + \begin{bmatrix} 0 \\ 1 \\ 1 \\ -1 \end{bmatrix} \begin{bmatrix} 0 & 1 & 1 & -1 \end{bmatrix} \\ &= \begin{bmatrix} 2 & -1 & 1 & 0 \\ -1 & 2 & -1 & 1 \\ 1 & -1 & 2 & 1 \\ 0 & 0 & 1 & 2 \end{bmatrix} + \begin{bmatrix} 0 & 0 & 0 & 0 \\ 0 & 1 & 1 & -1 \\ 0 & 1 & 1 & -1 \\ 0 & -1 & -1 & 1 \end{bmatrix} \\ &= \begin{bmatrix} 2 & -1 & 1 & 0 \\ -1 & 3 & 0 & 0 \\ 1 & 0 & 3 & 0 \\ 0 & 0 & 0 & 3 \end{bmatrix} \end{aligned}$$

Proposition 5.4. Suppose we have a simplicial complex G of order 1 (a graph). Then $L_0 = B_1 B_1^T = D - A$, where D and A are the degree matrix and adjacency matrix of G, respectively.

Proof. For ease of notation, since there can be no confusion we write $B_1 = B$ and $L_0 = L$. Let n be the number of vertices and m the number of edges. For each edge $e_k = (a, b)$, we choose an arbitrary orientation. Then

$$B_{i,k} = \begin{cases} 1 & \text{if } i = a \\ -1 & \text{if } i = b \\ 0 & \text{otherwise} \end{cases}$$

If we compute BB^T we can see that the diagonal entries of L are $L_{i,i} = \sum_{k=1}^{m} B_{i,k}^2$. Every time the vertex i is part of an edge, it contributes 1 to this sum, so $L_{i,i} = (\text{degree of vertex i}) = D_{i,i}$.

For $i \neq j$, we have that $L_{i,j} = \sum_{k=1}^{m} B_{i,k} B_{j,k}$. Now, each component of the sum only contributes something if $B_{i,k}$ and $B_{j,k}$ are adjacent, in which case one term is $+1$ and the other is -1; the product is then always -1. We have thus

$$L_{i,j} = \begin{cases} -1 & \text{if } B_{i,k} \text{ and } B_{j,k} \text{ are adjacent} \\ 0 & \text{otherwise} \end{cases}$$

This means that $L_{i \neq j} = -A_{i,j}$, and thus $L = D - A$. □

Remark 5.5. All Laplacian matrices of intermediate order (i.e. $n = 1, \ldots, N-1$) contain two terms: the first term, known as *lower/down Laplacian*, expresses the lower adjacency of n-order simplices; the second term, known as *upper/up Laplacian*, expresses the upper adjacency of n-order simplices. So, for example, two edges are lower adjacent if they share a common vertex, whereas they are upper adjacent if they are faces of a common triangle. Note that the vertices of a graph can only be upper adjacent (if they are incident to the same edge). That is why the graph Laplacian L_0 contains only one term.
This distinction between upper and lower Laplacian motivates the following definition.

Definition 5.6 (Up and Down Laplacian)**.** Let the n-th Hodge Laplacian be $L_n = B_n^T B_n + B_{n+1} B_{n+1}^T$. We define

- $L_n^{down} = L_n^d = B_n^T B_n$ as ***n-down Laplacian***
- $L_n^{up} = L_n^u = B_{n+1} B_{n+1}^T$ as ***n-up Laplacian***

Proposition 5.7. Let $\mathcal{K}$ be a finite oriented simplicial complex, and let k be an integer with $0 \leq k \leq dim(\mathcal{K})$. Let $\{ \sigma_1, ..., \sigma_n \}$ be the k-simplices of $\mathcal{K}$, and let $\{ \tau_1, ..., \tau_m \}$ be the $(k+1)$-simplices of $\mathcal{K}$. Let $i, j \in \{1, ..., n\}$. Then

$$(L_k^u)_{i,j} = \begin{cases} deg_U(\sigma_i) & \text{if } i = j \\ 1 & \text{if } i \neq j, \ \ \sigma_i \text{ and } \sigma_j \text{ are upper adjacent and } \sigma_i \sim \sigma_j \\ -1 & \text{if } i \neq j, \ \ \sigma_i \text{ and } \sigma_j \text{ are upper adjacent and } \sigma_i \nsim \sigma_j \\ 0 & \text{if } i \neq j, \ \ \sigma_i \text{ and } \sigma_j \text{ are not upper adjacent} \end{cases}$$

Proof. If $k = dim(\mathcal{K})$, then $\partial_{k+1} : \mathcal{C}_{k+1} \to \mathcal{C}_k$ is the zero map from the trivial vector space to another vector space, and L_k^u is the zero matrix. Since in this case there are no $(k+1)$-simplices in $\mathcal{K}$, no k-simplices are upper adjacent in $\mathcal{K}$ and the proposition follows.

Now suppose that $k < dim(\mathcal{K})$. Since $L_k^u = B_{k+1} B_{k+1}^T$, the component (i, j) of L_k^u is the standard dot product of the i-th and j-th rows of B_{k+1}. Let X and Y denote these rows, respectively.

Suppose $i = j$. Let τ_h be a $(k+1)$-simplex in $\mathcal{K}$. If σ_i is a face of τ_h, then the h-th component of X is either 1 or -1, depending on the orientation of σ_i, so the h-th summand in $X \cdot X$ will always be 1. If σ_i is not a face of τ_h, then the h-th component of X is 0, so the h-th summand of $X \cdot X$ will be 0. It follows that $X \cdot X$ is then equal to the number of $(k+1)$-simplices in $\mathcal{K}$ of which σ_i is a face, which is the given definition of upper degree of σ_i.

Suppose now $i \neq j$. Let τ_h be a $(k+1)$-simplex in $\mathcal{K}$. Suppose both σ_i and σ_j are faces of τ_h. If σ_i and σ_j have coherent orientation, then the h-th components of X and Y are either both 1 or -1, and the h-th summand in $X \cdot Y$ will be 1. If σ_i and σ_j have opposite orientation, then between the h-th components of X and Y, one is 1 and the other -1, so the h-th summand in $X \cdot Y$ will be -1. If σ_i and σ_j are not both faces of τ_h, say σ_i isn't one, then the h-th component of X is 0. The same is true for σ_j and Y, therefore the k-th summand in $X \cdot Y$ will be 0.

From proposition 4.14 we know that there is at most one $(k+1)$-simplex in $\mathcal{K}$ containing both σ_i and σ_j as faces. Therefore, if σ_i and σ_j are upper adjacent, then a single summand in $X \cdot Y$ is either 1 or -1 and all other summands are 0, so $(L_k^u)_{i,j} = X \cdot Y$ is either 1 or -1, depending on the two simplices' orientations. If σ_i and σ_j are not upper adjacent, then all the summands of $X \cdot Y$ are 0, and $(L_k^u)_{i,j} = X \cdot Y = 0$. □

Proposition 5.8. Let $\mathcal{K}$ be a finite oriented simplicial complex, and let k be an integer with $0 \leq k \leq dim(\mathcal{K})$. Let $\{\sigma_1, ..., \sigma_n\}$ be the k-simplices of $\mathcal{K}$, and let $i, j \in \{1, ..., n\}$. Then

$$(L_k^d)_{i,j} = \begin{cases} deg_L(\sigma_i) & \text{if } i = j \\ 1 & \text{if } i \neq j \text{ and } \sigma_i, \sigma_j \text{ have similar common lower simplex} \\ -1 & \text{if } i \neq j \text{ and } \sigma_i, \sigma_j \text{ have dissimilar common lower simplex} \\ 0 & \text{if } i \neq j \text{ and } \sigma_i, \sigma_j \text{ are not lower adjacent} \end{cases}$$

Proof. If $k = 0$, then $\partial_0 : C_0 \to C_{-1}$ is the zero map from a finite dimensional vector space to the trivial vector space, and L_k^d is thus the zero matrix. Since no vertices of a simplicial complex can be lower adjacent, the proposition follows.

Now suppose $k > 0$. Since $L_k^d = B_k^T B_k$, the component (i, j) of L_k^d is the standard dot product of the i-th and j-th columns of B_k. Let X and Y denote these columns, respectively, and let $\{\eta_1, ..., \eta_m\}$ be the $(k-1)$-simplices of $\mathcal{K}$.

Suppose $i = j$. Let η_h be a $(k-1)$-simplex of $\mathcal{K}$. If σ_i contains η_h as a face, the the h-th component of X is either 1 or -1, depending on the orientation of η_h, so the h-th summand of $X \cdot X$ will be 1. If σ_i does not contain η_h as a face, then the h-th component of X is 0, and the h-th summand of $X \cdot X$ will be 0. It follows that $X \cdot X$ is then equal to the number of $(k-1)$-faces of σ_i, which is the given definition of lower degree of σ_i.

Suppose now $i \neq j$. Let η_h be a $(k-1)$-simplex of $\mathcal{K}$. Suppose σ_i and σ_j are lower adjacent with $\sigma_i \cap \sigma_j = \eta_h$. If η_h is a similar common lower simplex, that is, if it has the same orientation when traversed through both σ_i and σ_j, then the h-th components of X and Y are thus both either 1 or -1, and the h-th summand in $X \cdot Y$ will be 1. If η_h is a dissimilar common lower simplex, that is, if when traversed through σ_i and σ_j it has opposite orientations, then of the h-th components of X and Y, one will be 1 and the other -1. The h-th summand in $X \cdot Y$ will thus be -1. If η_h is not a face of both σ_i and σ_j, then the h-th component of X or the h-th component of Y is 0. Thus, if η_h is not a common lower simplex of σ_i and σ_j, the h-th summand in $X \cdot Y$ is

0.

From proposition 4.15 we know that there is at most one $(k-1)$-simplex in $\mathcal{K}$ that is a face of both σ_i and σ_j. Therefore, if σ_i and σ_j have opposite orientation, then a single summand of $X \cdot Y$ is 1 and all other summands are 0, so $(L_k^d)_{i,j} = X \cdot Y = 1$. If σ_i and σ_j have coherent orientation, then a single summand of $X \cdot Y = -1$ and all the other summands are 0, so $(L_k^d)_{i,j} = X \cdot Y = -1$. If σ_i and σ_j are not lower adjacent, then all the summands of $X \cdot Y$ are 0, so $(L_k^d)_{i,j} = X \cdot Y = 0$. □

Theorem 5.9. Let $\mathcal{K}$ be a finite oriented simplicial complex, and let k be an integer with $0 \leq k \leq dim(\mathcal{K})$. Let $\{\sigma_1, ..., \sigma_n\}$ be the k-simplices of $\mathcal{K}$, and let $i, j \in \{1, ..., n\}$.

1. If $k = 0$, then

$$(L_k)_{i,j} = \begin{cases} deg_U(\sigma_i) & \text{if } i = j \\ -1 & \text{if } \sigma_i, \sigma_j \text{ are distinct and upper adjacent} \\ 0 & \text{if } \sigma_i, \sigma_j \text{ are distinct and not upper adjacent} \end{cases}$$

2. If $k > 0$, then

$$(L_k)_{i,j} = \begin{cases} deg_U(\sigma_i) + k + 1 & \text{if } i = j \\ 1 & \text{if } i \neq j \text{ and } \sigma_i, \sigma_j \text{ are not upper adjacent but have a similar common lower simplex} \\ -1 & \text{if } i \neq j \text{ and } \sigma_i, \sigma_j \text{ are not upper adjacent but have a dissimilar common lower simplex} \\ 0 & \text{if } i \neq j \text{ and either } \sigma_i, \sigma_j \text{ are upper adjacent or they are not lower adjacent} \end{cases}$$

Proof. (1) Suppose $k = 0$. We have seen in proposition 5.4 that $L_0 = B_1 B_1^T$ and how it reflects in the components of L_0. In fact, this part of the proof follows directly from proposition 5.4. (2) From now on, let $k > 0$. Suppose $i = j$, then from proposition 5.7 and proposition 5.8 we know that $(L_k)_{i,i} = (L_k^u)_{i,i} + (L_k^d)_{i,i} = deg_U(\sigma_i) + deg_L(\sigma_i)$. Since every k-simplex has exactly $k+1$ $(k-1)$-faces, we see that $(L_k)_{i,i} = deg_U(\sigma_i) + k + 1$.

Suppose now $i \neq j$. If σ_i and σ_j are not upper adjacent but have a similar common lower simplex, then from proposition 5.7 and proposition 5.8 we know that $(L_k)_{i,j} = (L_k^u)_{i,j} + (L_k^d)_{i,j} =$

$0+1=1$. If σ_i and σ_j are not upper adjacent but have a dissimilar common lower simplex, then from the same propositions we know that $(L_k)_{i,j} = (L_k^u)_{i,j} + (L_k^d)_{i,j} = 0 + (-1) = -1$.

Suppose now that σ_i and σ_j are upper adjacent. If they have coherent orientation, then they have a dissimilar common lower simplex (this is a general fact, proven e.g. in [11, Lemma 3.2.6]); by the same propositions used above, we have that $(L_k)_{i,j} = (L_k^u)_{i,j} + (L_k^d)_{i,j} = 1 + (-1) = 0$. If instead they have opposite orientation, then they have a similar common lower simplex, so $(L_k)_{i,j} = (L_k^u)_{i,j} + (L_k^d)_{i,j} = -1 + 1 = 0$.

Finally, if σ_i and σ_j are not lower adjacent, then they are also not upper adjacent, and from the same propositions above we have that $(L_k)_{i,j} = (L_k^u)_{i,j} + (L_k^d)_{i,j} = 0 + 0 = 0$. □

5.2 Hodge Decomposition

From the definition we gave, it follows that the Hodge Laplacian is real, symmetric and positive semidefinite (it is the sum of two positive semidefinite operators). Moreover,

$$\mathrm{Im}(L_n^d) \subseteq \ker(L_n^u), \qquad \mathrm{Im}(L_n^u) \subseteq \ker(L_n^d), \qquad \ker(L_n) = \ker(L_n^u) \cap \ker(L_n^d)$$

This means that an eigenvector of L_n relative to a non-zero eigenvalue is either a non-zero eigenvector of L_n^d or a non-zero eigenvector of L_n^u.

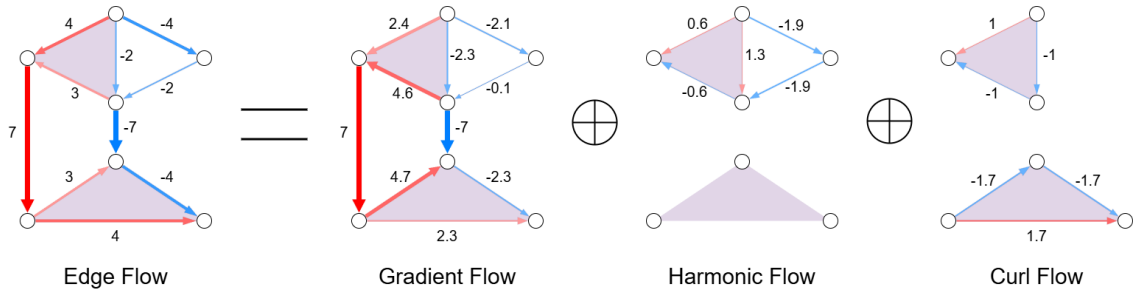
We can in fact decompose the space on which L_n, L_n^u and L_n^d act, that is the space C_n of all n-chains (check fig. 4.4 for a reminder). We have that $C_n = \ker(L_n) \oplus \mathrm{Im}(L_n)$, and it's easy to see that $\mathrm{Im}(L_n) \subseteq \mathrm{Im}(B_n^T) \cup \mathrm{Im}(B_{n+1})$. Moreover, since $\mathrm{Im}(B_{n+1}) \subset \ker(B_n^T)$, we obtain $\mathrm{Im}(L_n) \subseteq \mathrm{Im}(B_n^T) \oplus (B_{n+1})$. Since the dimension of the right-hand side cannot exceed that of $\mathrm{Im}(L_n)$, the subspaces must be equal, and we can decompose C_n as follows:

$$C_n = \mathrm{Im}(B_n^T) \oplus \ker(L_n) \oplus \mathrm{Im}(B_{n+1})$$

This is known as ***Hodge Decomposition***, a powerful concept with applications in many fields.

Remark 5.10. For $k = 1$, the Hodge decomposition tells us that any 1-chain can be decomposed into the sum of three orthogonal components: a gradient (image of B_1), a curl (image of B_2^T) and a harmonic representative (kernel of L_1). There exist of course analogous examples for arbitrary n, where one can define higher-dimensional curl and gradient operators.

Example 5.11. Consider a simplicial complex of order 2 embedded with an edge flow. Hodge decomposition can then be applied to this flow:



Example of Hodge decomposition of an edge flow (courtesy of Schaub et al. [12])

5.3 Denoising

In chapter 3, we saw an application of the graph Laplacian for the denoising of a graph signal. What happens, then, when the data of interest is not located on the nodes of the graph but instead on its edges? Such cases arise, for example, when we are interested in is the flow on the edges, like in traffic or brain networks. For this application, we follow the work of Schaub and Segarra [13].

Suppose we have an unoriented graph that has been endowed with oriented flows on the edges, but we only get to observe a noisy version.

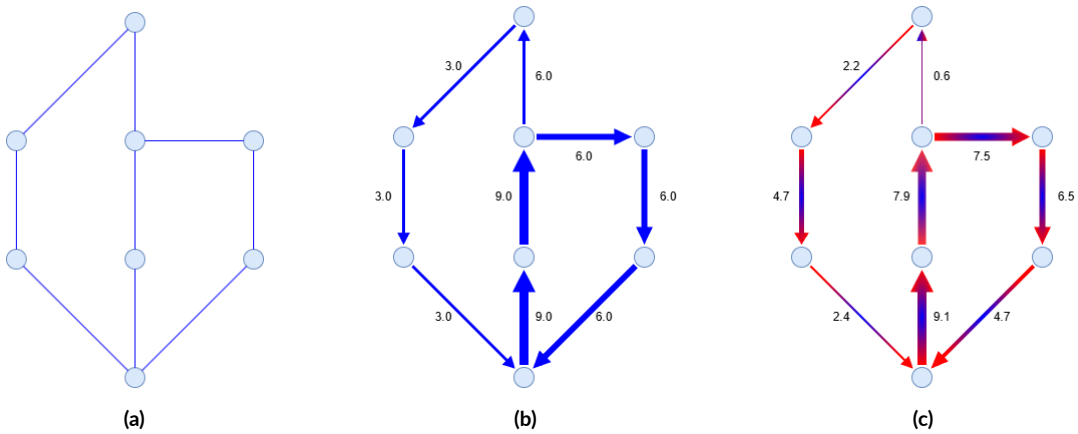


Figure 5.3: **(a)**The undirected graph, **(b)** the endowed flow and **(c)** the noisy flow

5.3.1 Line-graph approach

Definition 5.12 (Line-graph). Given a graph G we define its ***Line-graph*** G_{LG} as the graph whose nodes correspond to the edges of G, and two nodes are connected if the corresponding edges in G share an incident node. Given the incidence matrix of G M_G, the adjacency matrix of G_{LG} can be written as $A_{LG} = |M_G^T M_G - 2\mathbb{I}|$ (with the absolute value applied element-wise). The graph Laplacian of G_{LG} is then defined as $L_{LG} = diag(A_{LG}\mathbb{I}) - A_{LG} = M_{LG}M_{LG}^T$, where M_{LG} is the incidence matrix of G_{LG}.

In section 3.2 we talked about low-pass filters. Considering a signal x on the nodes, two examples of such procedures are given here by

$$\hat{x} = (\mathbb{I} + \alpha L_{LG})^{-1}x \quad \text{(node denoising)} \quad \text{and} \quad \hat{x} = (\mathbb{I} - \mu L_{LG})^{k}x \quad \text{(node smoothing)}$$

By applying these techniques, we can see that the signal is smoothed towards the global average Figure 5.4. While there can certainly be found interpretations about this behaviour, the results are definitely not satisfying. In fact, it can be computed that the error between the original and the filtered signal is even bigger than the error relative to the unfiltered noisy signal. We need a more powerful instrument to approach the problem, and to do so we look at the higher-order structure of the Hodge Laplacians.

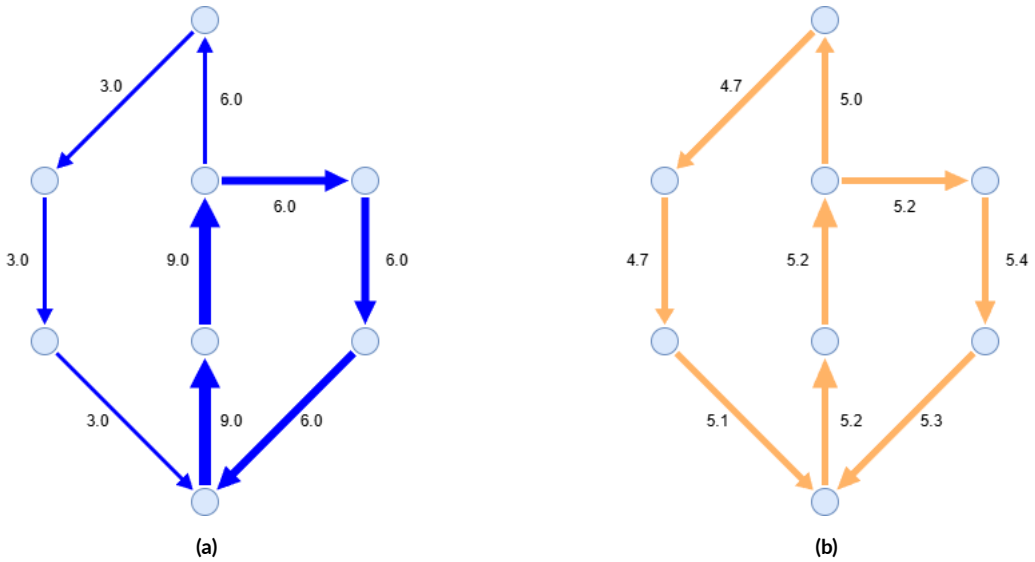


Figure 5.4: **(a)**The original flow and **(b)** the denoised flow obtained by using the line-graph

5.3.2 Edge-space approach

Definition 5.13 (Edge Laplacian)**.** Consider a simplicial complex G of order 1. Following the definition we gave in Theorem 5.2, its Hodge Laplacian are thus:

$$L_0 = B_1 B_1^T \qquad \text{and} \qquad L_1 = B_1^T B_1$$

We have seen that L_0 is just the graph Laplacian, while we will call L_1 the ***Edge Laplacian*** of G.

We can now proceed further by filtering the flow signal f on the edges by considering the operations

$$\hat{f} = (\mathbb{I} + \alpha L_1)^{-1} f \quad \text{(flow denoising)} \quad \text{and} \quad \hat{f} = (\mathbb{I} - \mu L_1)^k f \quad \text{(flow smoothing)}$$

By appropriately choosing values for α, μ and k, we obtain a much more satisfying result.

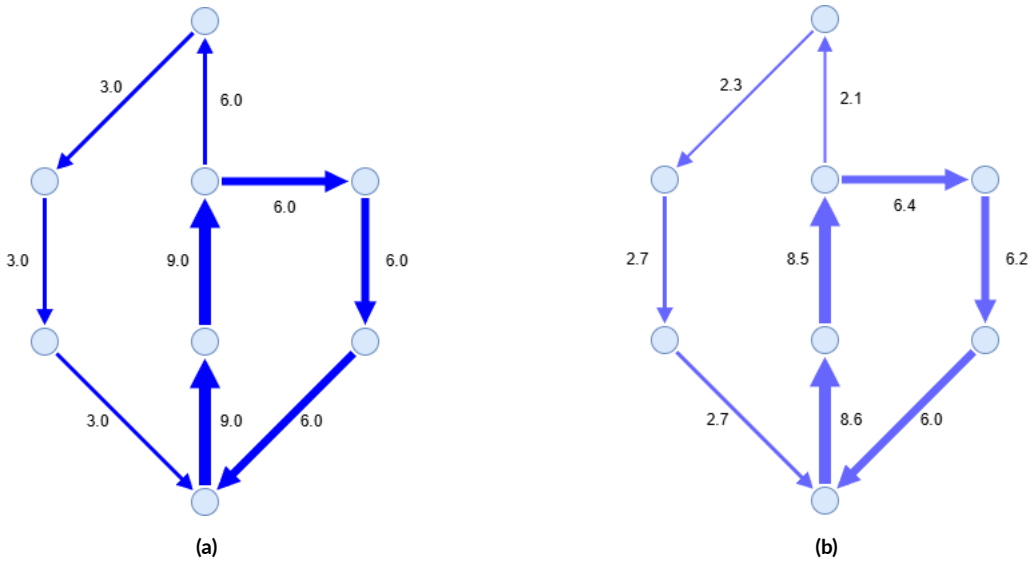


Figure 5.5: (a) The original flow and **(b)** the denoised flow obtained by using the edge-graph

5.4 Spectral Clustering

In section 3.3 we saw how the spectrum of the graph Laplacian could help us to identify clusters in an undirected graph. What if we were to embed such a graph with a flow, like we did in

section 5.3 ? We saw that the graph Laplacian is no longer a powerful enough instrument, and we have to consider a higher-order structure. We will now follow an example given by Shaub et al. [14] where the Hodge Laplacian is employed to exploit the underlying topological structure of a simplicial complex.

5.4.1 Normalized Hodge Laplacian

For the example at hand, we will use a slightly different formulation of the Hodge Laplacian by normalizing it, in a similar way to how it is often done for the graph Laplacian.

Definition 5.14 (Normalized Hodge 1-Laplacian)**.** Consider a simplicial complex $\mathcal{K}$ whose boundary operators are represented by the matrices B_0 and B_1. Here, we will use $|A|$ to denote the matrix whose elements are given by the absolute values of the entries of A. The ***normalized Hodge 1-Laplacian*** matrix is then defined as

$$\mathcal{L}_1 = D_1 B_0^T D_0^{-1} B_0 \; + B_1 D_2 B_1^T D_1^{-1}$$

where:

- $D_0 = 2\, diag(|B_0| D_1 1)$ is the diagonal matrix of weighted degrees of the nodes
- $D_1 = max\,(diag(|B_1|1), \mathbb{I})$ is the diagonal matrix of (adjusted) degrees of each edge
- $D_2 = \frac{1}{3}\mathbb{I}$

Note: D_0, D_1, D_2 all represent information of the weighted degrees of 0-, 1-, and 2-simplices respectively.

Remark 5.15. Since solutions to the Laplace equation $\Delta x = 0$ are called *harmonic functions* and L_n can be interpreted as a discretized version of the Laplace equation [15], elements $h \in ker(L_n)$ are also called harmonic functions, and they carry a topological meaning. In the following paragraphs we will consider the 0-eigenvalues of $\mathcal{L}_1$ and the associated harmonic functions.

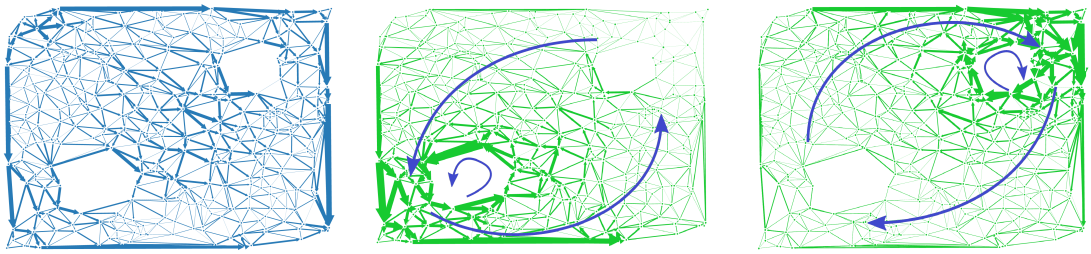
5.4.2 Embedding a flow

The simplicial complex we will use as example is built as follows:

- Draw 400 random points in the unit square
- Generate a triangular lattice via Delauney triangulation [16]
- Eliminate all the edges inside two predetermined regions (to create "holes")
- Define all triangles to be faces

The resulting simplicial complex has two holes and, accordingly to the theory, its normalized Hodge Laplacian $\mathcal{L}_1$ has two zero eigenvalues (more on Betti numbers and on the relationship between n-dimensional holes and the zero eigenvalues can be found in [17]). These eigenvalues can then be associated to two harmonic functions that encircle the two holes of the simplicial complex.

To avoid the differentiation of left and right eigenvectors, we will use the symmetrized Laplacian $\mathcal{L}_1^s = D_1^{-1/2}\mathcal{L}_1 D_1^{1/2}$ instead of $\mathcal{L}_1$. In a similar way to what we did for the graph Laplacian, we consider the spectral decomposition $\mathcal{L}_1^s = U\Lambda U^T$ and define $H := (h_1, \dots, h_k)$ to be the matrix collecting the harmonic functions associated to $\mathcal{L}_1^s$, where k is the multiplicity of the zero eigenvalue of $\mathcal{L}_1^s$. Now we denote the indicator vector of a positive flow on the edge $e = [i,j]$ by $\mathbf{e}$ (that is, $\mathbf{e}_{[i,j]} = 1$ and 0 otherwise), and define the harmonic embedding of and edge e through the mapping $e \mapsto \boldsymbol{l}_e = H^T\mathbf{e} \in \mathbb{R}^k$. The embedding measures how much a unit flow along $[i,j]$ contributes to the global circular flows. Consider a flow $\boldsymbol{f}$ on the simplicial complex:



(a) **(b)** **(c)**

Figure 5.6: (a) The flow embedded on the simplicial complex, with arrows that indicate the direction and magnitude of the flow. The harmonic functions h_1 **(b)** and h_2 **(c)** of $\mathcal{L}_1^s$, where the blue arrows indicate how the flows encircle the holes.

For each edge $[i,j]$ we construct the weighted indicator vector $\boldsymbol{f}_{[i,j]} = f([i,j])\mathbf{e}$, and then compute its projection onto the harmonic subspace $\boldsymbol{l}_f = H^T\boldsymbol{f}_{[i,j]}$

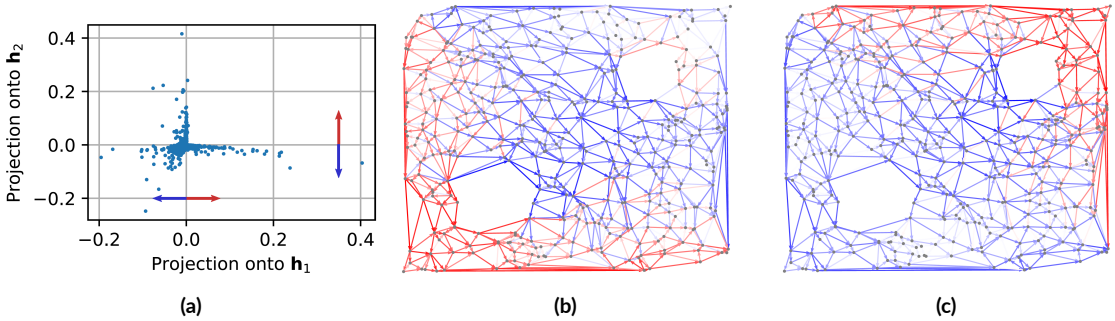


Figure 5.7: (a) Projection of each edge flow $f_{[i,j]}$ onto the harmonic functions, and **(b)**,**(c)** the projection of the edge flow onto harmonic functions h_1 and h_2, respectively. The colour red indicates a positive value, the colour blue a negative value.

5.4.3 Embedding trajectories

A second step taken by Schaub et al., inspired by the work of Gosh et al. [18], is to consider the embedding of trajectories of flow vectors defined on contiguous edges. They employ a set of 9 trajectories and project them onto the harmonic eigenvectors of $\mathcal{L}_1^s$.

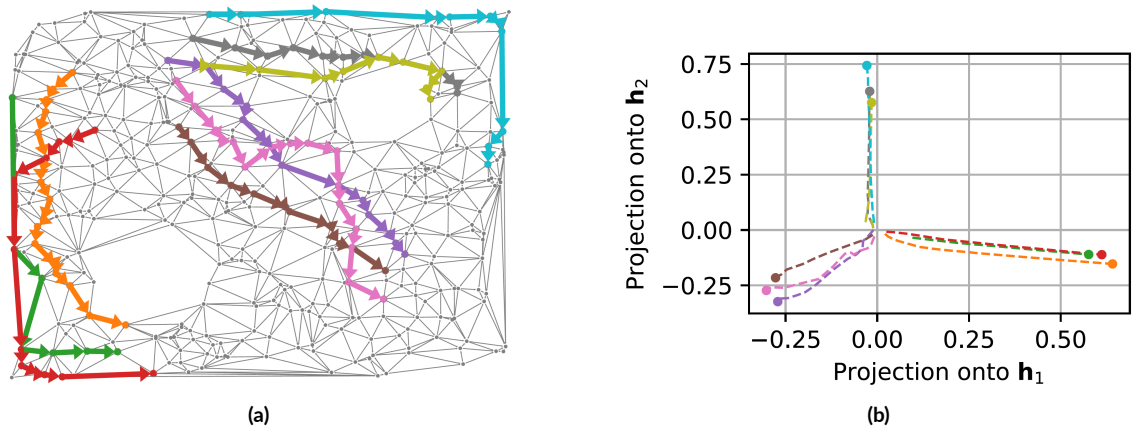


Figure 5.8: (a) The set of 9 trajectories on the simplicial complex, and **(b)** the projection of those trajectories onto the harmonic eigenvectors of $\mathcal{L}_1^s$

We can see how trajectories that take similar paths in relation to the topological structure (below the south-western hole, between the holes and above the north-eastern hole) are clustered together once projected onto the harmonic subspace.

In their work, Schaub et al. also give an interesting real-world application by employing the previous techniques to analyze ocean drifter trajectories around Madagascar, by considering the island as the hole around which the simplicial complex (the ocean) is built. More details can be found in [18].

5.5 Community detection

One of the main topics in Network Science is community detection, which aims to identify groups of people that share some kind of trait inside a network. Usually these networks are modelled as graphs, and one way in which community detection can be done on a graph is to apply some form of spectral clustering. The results, however, are usually not satisfying and other methods are employed to obtain a better detection.

Considering that many real-world networks deal with a flow along the edges, and that a community structure is surely better captured by a high-dimensional network compared to a graph, it makes sense to approach the problem of community detection through the use of simplicial complexes.

One such approach is given by Krishnagopal and Bianconi [19], where they propose an algorithm for simplicial community detection using the spectrum of the Hodge Laplacian.

5.5.1 Up- and down- Laplacian

In their work, Krishnagopal and Bianconi underline the geometrical significance of the Laplacian through the use of the n-up and n-down Laplacian (recall definition 5.6). In their work, though, the definition of n-down Laplacian differs a bit from the one we previously used, and a reader who would confront the two might be confused, so we will try to set things straight.

Krishnagopal and Bianconi refer to the orientation of two n-simplices to define the value of L_n^d, implying that two n-simplices have intrinsic orientations that can be confronted by just observing them. This is not entirely true, because if the two n-simplices are not lower adjacent there is no way to deduce the relation between the two orientations. While their decision probably stems from an unspecified assumption of lower adjacency, we felt a more correct approach to explicitly rely on the common lower simplex (or lack thereof) for the definition of the n-down Laplacian.

Remark 5.16. L_k^u is encoding information of the flow from k-simplices to other k-simplices that are $(k+1)$-connected, while L_k^d encodes information between k-simplices that are $(k-1)$-connected. From the matrix representation we have given, we can see how the off-diagonal elements are non-zero only among pairs that are either upper or lower-adjacent. Thus, there exists a base of eigenvectors of L_k^u in which the k-simplices in the support of each eigenvector belong to a single k-simplicial community. In a similar fashion, there exists a base of eigenvectors

of L_k^d in which the k-simplices in the support of each eigenvector belong to a single $(k-1)$-simplicial community. By looking at the spectral properties of L_k^u and L_{k+1}^d, it is thus possible to identify the k-simplicial communities of a simplicial complex. One last thing to note is that since there is an isomorphism between L_k^u and L_{k+1}^d, the k-up communities are isomorphic to the $(k+1)$-down communities.

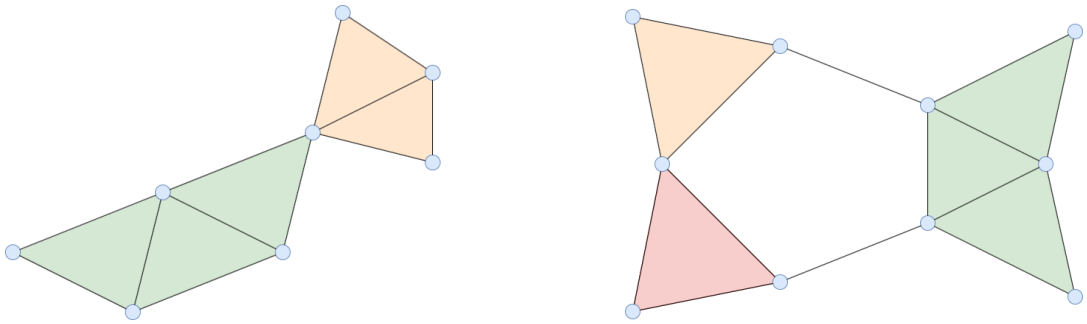


Figure 5.9: Two examples of communities detected from the spectrum of the 2-down Laplacian L_2^d. In **(a)** we have 2 detected communities of 2-simplices, while in **(b)** we have 3 communities of 2-simplices, with a hole in the middle.

In Figure 5.10 we extend the method to a higher dimension. Instead of limiting ourselves to communities of 2-simplices, we also try to detect 3-simplices through the spectrum of L_3^d (we stopped at dimension 3 for better visualization).

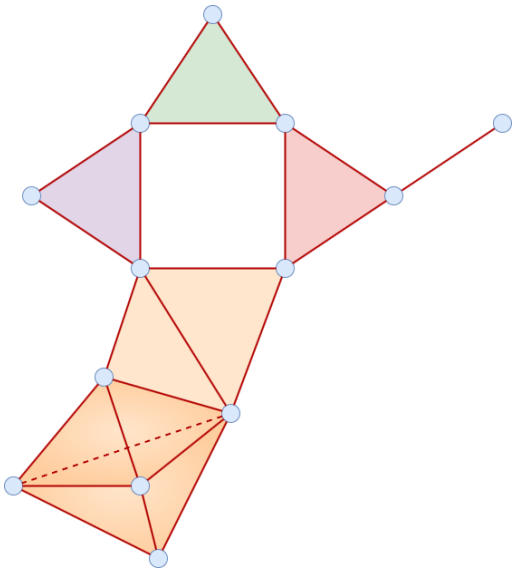


Figure 5.10: Colour-coded communities of different dimensions

Since the 1-skeleton is fully connected, there is only one 1-down/0-up community (edges connected by nodes/nodes connected by edges) and it is coloured in red. The 2-down communities (triangles connected by edges) have been divided in 4 categories, each of a different colour. Finally, the two tetrahedrons to the south of the figure form the only 3-down community. Note that the faces of the tetrahedron are all part of the same 2-down community coloured in orange, we just used a slightly different tint to emphasize the tetrahedrons.

5.5.2 The Algorithm

While the actual Python code is available through their paper [19], the pseudocode is the following:

Algorithm 5.1 Simplicial Community Detection

Data: n-dimensional simplicial complex
Result: k-simplicial communities
commList = []
for *k from* 1 *to* K **do**
  commList.append([])
  compute sparse boundary matrices B_k
  compute L_k^u
  find nonzero eigenvectors $\{v^n\} = [v_1^u, ..., v_{n_k}^u]$ of L_k^u
  Initialize $w_i^{(0)} = \sup(v_i^u)$
  if $w_i^{(r)} \cap w_j^{(r)} \neq \varnothing$ *and* $\lambda_i = \lambda_j$ **then**
    Take the union of overlapping supports
    $w_i^{(r+1)} = w_j^{(r+1)} = w_i^{(r)} \cup w_j^{(r)}$
  end
  commList[k].append(w)
  Visualize communities
end

5.5.3 Communities in Language

Another interesting example provided is given by a simplicial complex obtained from the word association network encoding relationships between characters from Les Misérables (the Victor Hugo's novel). It is built considering 77 nodes, each representing a character, and 254 edges connecting two characters whenever they appear in the same chapter. Each edge is given a weight, representing the number of times they both appear in the same sentence. The network is found to contain simplices of dimension up to 6, which is quite hard to properly visualize. Nonetheless, we write here as example some of the 5-up communities found through the algorithm:

- Blacheville, Dahlia, Fameuil, Fantine, Favourite, Listolier, Tholomyes, Zephine
- Babet, Brujon, Claquesous, Eponine, Gueulemer, Montparnasse, Thernadier
- Bamatabois, Brevet, Champmathieu, Chenildieu, Cochepaille, Judge, Vanjean

A complete list of the various communities found (from 1-up to 6-up) and a tentative visual representation is given in [19]. As seen here and in the paper, the general structure of the communities agrees with those in the novel.

6 Conclusions

Throughout the thesis we have seen how the graph structure can be used to model many problems, and how the spectral properties of the graph Laplacian can help solve them. We gave examples on denoising, clustering and dimensionality reduction, showing how widespread the applications are.

However, we also understood that the intricacies and complexities of the real world often require more than what the relatively simple structure of the graph can offer. It was from this perspective that we introduced the simplicial complex, a higher-order generalization of the concept of graph.

We laid the necessary theoretical groundwork of algebraic topology to properly formalize the definitions, to then present the higher-order counterpart to the graph Laplacian: the Hodge Laplacian. In a parallel to what was done for the graph Laplacian, we presented examples of problems from different branches of data analysis. In all these applications we showed how the usual graph methods were inefficient and how employing a high-order approach was necessary to achieve satisfying results.

When dealing with a denoising problem, we showed how noise on the edges requires the Hodge Laplacian to be solved through what we called an "edge-space approach".

When dealing with a clustering problem, we once again realized how flow on the edges has to be seen through the lenses of a higher-order structure. This also helped us identify topological and geometrical properties of the space we were working in, such as "holes" or "cavities", that have clear real-world applications.

Finally, we looked at community detection. It is known in the literature how spectral approaches based on graphs are quite weak, so we looked at exploiting the Hodge Laplacian. By separating the operator in the up- and down-Laplacian, we focused on the geometrical significance of the Hodge Laplacian and managed to apply an algorithm to detect simplicial communities. This achieved satisfying results even when applied to a rather involved framework such as the book Les Misérables by Victor Hugo.

What emerges from all of this is the versatility and power of the Hodge operator. Only in the recent years it has been the subject of research in data analysis, and the results are promising. A complex world might not necessarily require a complex model, but we need a way to capture the intricacies of interactions beyond simple dyadic relations, and the higher-order structures we have seen can manage that. While it is certainly more difficult to properly model and build a simplicial complex compared to a graph, we feel like the effort is well worth it when faced with the results.

References

[1] C. Bick, E. Gross, H. A. Harrington, and M. T. Schaub, "What are higher-order networks?" *SIAM Review*, vol. 65, no. 3, pp. 686–731, 2023.

[2] J. A. Bondy and U. S. R. Murty, *Graph Theory*. Springer, 2008.

[3] R. Diestel, *Graph Theory*. Springer, 2017.

[4] G. Kirchhoff, "Ueber die auflösung der gleichungen, auf welche man bei der untersuchung der linearen vertheilung galvanischer ströme geführt wird," *Annalen der Physik und Chemie*, vol. 72, no. 12, pp. 497–508, 1847.

[5] E. M. Luks, "Isomorphism of graphs of bounded valence can be tested in polynomial time," *Journal of Computer and System Sciences*, vol. 25, no. 1, pp. 42–65, 1982.

[6] C. Kuratowski, "Théorème sur trois continus," *Monatshefte für Mathematik und Physik*, vol. 36, no. 1, pp. 77–80, 1929.

[7] W. Erb, "Lecture notes in matematical models and numerical methods for big data," 2024.

[8] J. Shi and J. Malik, "Normalized cuts and image segmentation," *IEEE Transactions on pattern analysis and machine intelligence*, vol. 22, no. 8, pp. 888–905, 2000.

[9] J. MacQueen, "Some methods for classification and analysis of multivariate observations," *Proceedings of the 5th Berkeley Symposium on Mathematical Statistics and Probability*, vol. 1, pp. 281–297, 1967.

[10] A. Hatcher, *Algebraic Topology*. Cambridge University Press, 2001.

[11] T. E. Goldberg, "Combinatorial laplacians of simplicial complexes," 2002.

[12] M. T. Schaub, Y. Zhu, J.-B. Seby, T. M. Roddenberry, and S. Segarra, "Signal processing on higher-order networks: Livin' on the edge... and beyond," *Signal Processing*, vol.

187, pp. 108–149, 2021. [Online]. Available: https://www.sciencedirect.com/science/article/pii/S0165168421001870

[13] M. T. Schaub and S. Segarra, "Flow smoothing and denoising: Graph signal processing in the edge-space," *CoRR*, vol. abs/1808.02111, 2018. [Online]. Available: http://arxiv.org/abs/1808.02111

[14] M. T. Schaub, A. R. Benson, P. Horn, G. Lippner, and A. Jadbabaie, "Random walks on simplicial complexes and the normalized hodge 1-laplacian," *SIAM Review*, vol. 62, no. 2, p. 353–391, Jan. 2020. [Online]. Available: http://dx.doi.org/10.1137/18M1201019

[15] L.-H. Lim, "Hodge laplacians on graphs," 2019. [Online]. Available: https://arxiv.org/abs/1507.05379

[16] M. de Berg, O. Cheong, M. van Kreveld, and M. Overmars, *Computational Geometry*. Springer, 2008.

[17] J. R. Munkres, *Elements of Algebraic Topology*. Addison-Wesley Publishing Company, 1984.

[18] A. Ghosh, B. Rozemberczki, S. Ramamoorthy, and R. Sarkar, "Topological signatures for fast mobility analysis," 11 2018.

[19] S. Krishnagopal and G. Bianconi, "Spectral detection of simplicial communities via hodge laplacians," *Physical Review E*, vol. 104, no. 6, Dec. 2021. [Online]. Available: http://dx.doi.org/10.1103/PhysRevE.104.064303