TECHNISCHE
UNIVERSITÄT
WIEN

Diploma Thesis

Signature Graphs in the Context of Compositional Data

performed at the

Institute of Telecommunications,
Faculty of Electrical Engineering and Information Technology,
TU Wien

supervised by

Univ.-Prof. Dipl.-Ing. Dr. techn. Gerald Matz

by

Dimitrios Kalodikis, BSc

Vienna, May 2024

Contents

Abstract

Over the last years, graph signal processing has become a rich toolbox for treating data living on irregular domains. Graphs can be used to model pairwise relationships and thus provide high flexibility in modelling the structure of various problems. Many proposals were made to extend the traditional concept of graphs in order to capture a problem-inherent structure even more accurately. Naturally, each of these graph classes requires a suitably adapted mathematical framework.

A recent proposal of a novel graph class by Dittrich and Matz are so-called signature graphs. These graphs model relationships in data by a non-negative scalar and a vector of signs, capturing an overall distance/correlation in the data by the scalar part, as well as relationships of different features in the data, each described by one sign in the sign vector. This model was shown to provide advantage over ordinary weighted signed graphs in clustering tasks. Recently, it was discovered that the usefulness of signature graphs can be leveraged when used in conjunction with involutions that describe symmetries in the data at hand.

In this thesis, we develop a framework to use signature graphs with compositional data, that is datasets in which each datapoint describes a composition, e.g., chemical compounds of a sample. Since for such data only proportions matter, statistical treatment of compositional data shall be invariant to scaling. Therefore, traditional methods based on Euclidean geometry cannot be applied for a meaningful analysis. Aitchison laid the foundations of compositional data analysis by defining a new geometry (Aitchison geometry) which respects the principle of scale invariance among other advantageous properties.

After giving a more thorough introduction into the fundamentals of both, graph signal processing and compositional data, we introduce novel linear and affine involutions in compositional data and propose a new type of transform to the Euclidean vector space, which allows a convenient description of the involutions in question. These involutions are parametrised, which makes them flexible in adapting to the considered data, but also necessitates a knowledge of these parameters for meaningful application. Thus, we proceed by developing methods to estimate the parameters in two scenarios: First, we assume to know pairwise relations of a few datapoints a priori, i.e. the signature that is later characterised by the signature graph. Based on this assumption we develop an involution estimator, which is ultimately targeted for clustering applications. Then, we propose a method following the concepts of blind source separation, which relies on prior information about the existence and non-existence of statistical correlations between datapoints, targeted for interpolation tasks in a scenario were we have a good understanding of the problems topology.

Based on the identified involution, we propose a method for learning a signature graph from compositional data. Two equivalent formulations are stated and rated according to their computational costs. We then proceed to describe the clustering of the learned graph, going into the peculiarities of signature graphs and the concept of balancedness in signature graphs. A numerical study proves the advantage of using signature graphs over ordinary graphs as a basis for classification.

Finally, we deal with the problem of reconstructing graph signals on balanced signature graphs from incomplete observations, i.e., interpolation. We start out by elucidating how the edge weights of a signature graph can be learned from correlations in observed data. Furthermore, we discuss the issue of balancing an unbalanced signature graph, before we present methods for bandlimited and Laplacian reconstruction on signature graphs, and how they can be simplified if the graph is balanced. Numerical experiments confirm the usefulness of our proposed methods.

Kurzfassung

In den letzten Jahren ist das Feld der Signalverarbeitung auf Graphen aufgeblüht und stellt eine reiche Methodensammlung für Signale auf unregelmäßigen Domänen dar. Graphen können zur Modellierung paarweiser Relationen genutzt werden und bieten damit eine hohe Flexibilität in der Erfassung von Strukturen, die verschiedensten Problemstellungen zugrunde liegen. Es wurden viele Vorschläge zur Erweiterung des Konzepts traditioneller Graphen gemacht um die probleminhärenten Strukturen noch akkurater abbilden zu können. Selbstverständlich erfordert jede dieser Graphklassen ein eigenes, angepasstes, mathematisches Gerüst.

Ein neuerer Vorschlag von Dittrich und Matz für eine solche Graphklasse sind sogenannte Signaturgraphen. Diese erlauben die Beschreibung von Relationen zwischen Datenpunkten durch einen Skalar und einen Vektor aus Vorzeichen. Somit kann eine generelle Distanz/Korrelation zwischen Datenpunkten und eine binäre Beschreibung mehrerer Merkmale der Daten als ähnlich oder gegensätzlich ausgedrückt werden. Es wurde gezeigt, dass in Clusteringanwendungen diese Beschreibung gegenüber gewöhnlichen, signierten und gewichteten Graphen überlegen ist. Jüngst wurde auch gezeigt, dass Signaturgraphen besonders zweckmäßig eingesetzt werden können, wenn gewisse Symmetrien im Datensatz vorhanden sind, die mit Involutionen beschreibbar sind.

In dieser Arbeit entwickeln wir ein methodisches Gerüst zur Behandlung von Kompositionsdaten auf Signaturgraphen. Kompositionsdaten beschreiben Teile eines Ganzen, z.B. die Zusammensetzung einer chemischen Probe. Da bei solchen Daten nur die Proportionen zählen, sollte die statistische Behandlung von Kompositionsdaten invariant gegenüber Skalierung sein. Dies schließt Werkzeuge, die auf der Annahme einer euklidischen Geometrie fußen, für eine aussagekräftige Analyse aus. Aitchison hat das Feld der Kompositionsdaten begründet, indem er eine alternative Geometrie (die Aitchison-Geometrie) erdacht hat, die unter anderem das Prinzip der Skalierungsinvarianz berücksichtigt.

Nachem wir eine umfassendere Einführung in Graphsignalverarbeitung und Kompositionsdaten gegeben haben, widmen wir uns Involutionen in Kompositionsdaten. Wir geben die lineare und die affine Involutionsklasse an und führen eine neuartige Transformation in den euklidischen Vektorraum ein, die eine ergonomische mathematische Beschreibung besagter Involutionen erlaubt. Der Umstand, dass diese Involutionsklassen parametrisiert sind, macht sie zwar flexibel in der Anpassung an unsere Daten, erfordert aber gleichzeitig eine Kenntnis über die Parameter. Wir erarbeiten zwei Methoden zur Parameterschätzung: Zuerst nehmen wir an, wir kennen von einer kleinen Menge von Datenpunkten ihre Signaturrelation. Die abgeleitete Schätzung ist auf Clusteringanwendungen ausgelegt. Der andere Schätzer nimmt Anleihen aus dem Gebiet der blinden Quellenseparation. Wir leiten dann den Schätzwert aus der angenommenen Existenz bzw. Nichtexistenz paarweiser Korrelationen in den Daten her. Diese Methode ist zugeschnitten auf Szenarien, in denen wir eine gute Kenntnis über die dem Problem zugrundeliegende Topologie haben.

Auf Basis der identifizierten Involutionen schlagen wir eine Methode zum Lernen von Signaturgraphen aus Kompositionsdaten vor. Zwei äquivalente Formulierungen werden angegeben und bezüglich ihres Rechenaufwands verglichen. Sodann beschreiben wir das Clustering des gelernten Graphen, wobei wir auf die Eigenheiten von Signaturgraphen und das Konzept der Balance auf diesen eingehen. Eine numerische Studie zeigt, dass Signaturgraphen gewöhnlichen Graphen beim Clustering überlegen sein können.

Abschließend befassen wir uns mit dem Problem der Rekonstruktion von unvollständigen Graphsignalen auf balancierten Signaturgraphen, also Interpolation. Wir beschreiben, wie Kantengewichte aus Korrelationen in beobachteten Daten geschätzt werden können. Außerdem diskutieren wir das Prob-

lem des Balancierens eines unbalancierten Signaturgraphen, bevor wir Methoden zur bandlimitierten und laplaceschen Rekonstruktion auf Signaturgraphen präsentieren und wie sich diese im Falle von balancierten Graphen vereinfachen. Eine weitere numerische Studie veranschaulicht die Leistungsfähigkeit der entwickelten Methoden.

Acknowledgement

First and foremost, I would like to express that I am deeply indebted to Prof. Gerald Matz, not only for his excellent supervision and guidance along the way of writing this thesis, but also for his distinguished lectures which were the initial spark for my interest in signal processing.

I would also like to express my gratitude to my colleague Philipp for his companionship in general and especially for countless inspiring conversations regarding the matters of this thesis.

Furthermore, I am thankful for my significant other for providing emotional balance and courage in times of doubt and stress and for being overall supportive.

Lastly, I would like to thank my parents wholeheartedly for supporting my educational path – even though I did not chose to become a physician.

Notation

We will introduce the notational conventions of this document by example.

Example	Meaning
	Scalar
	Vector
	Vector entry
	Matrix
	Matrix row
	Matrix entry
	All-one vector
	Expectation operator
	Set
	Number set
–	Random variable
	Indicator function

1 Introduction and Outline

We are undoubtedly living in a world that views data more and more as a resource of high value, which has led sociologists to coin the term information age to describe the past decades [1]. Some experts call data even the most valuable resource, superseding oil [2]. But not only is its value estimated higher than ever, we are also confronted with an exponential increase in the quantity of data generated and replicated worldwide [3]. These symptoms can be attributed to the success story of signal processing, a field which developed numerous methods for the treatment of data, like analysis, alteration and compression.

Traditionally, data has been viewed as numerical objects living on a regular domain, i.e. a domain with order relations and a distance metric. Classical examples include audio, where the datasets are recordings of the sound pressure along time. Order relations in time are described by the terms *before* and *after* and distance is understood as *delay*. Many methods implicitly assume such a regular domain, for example linear filter theory. However, the assumption of a regular domain is too strict for various types of data. Think for instance about a dataset that describes the gross domestic product (GDP) per capita of every country. The domain is now the set of countries, however there is no a priori notion of order or distance between countries. Without any domain description, however, we are severely limited in what we can do with the data.

A concept that has been proven effective to describe irregular domains are graphs. Graphs describe pairwise relationships between objects, the so-called nodes. The pairwise relationship is expressed by so-called edges, which connect two nodes [4, ch. 1.1]. In our GDP-per-country example we could think of a graph, whose nodes represent a country each. If we assume that neighbouring countries exhibit a correlation in the GDP it is sensible to add edges to our graph that represent the fact that the connected nodes are neighbouring countries. The irregular domain described by the graph is still not providing any notion of order, but a measure of distance can be defined, e.g. by the minimum number of edges to pass between two nodes, which greatly improves our signal processing capabilities. For instance, we can now check whether our GDP dataset is smooth over the graph, i.e., if neighbouring countries exhibit a positive correlation in their GDP.

These ideas are the foundations of graph signal processing, a field of active research [5]. Many generalisations have been proposed to the concept of graphs to achieve even more flexibility in modelling the domain, e.g. hypergraphs, which are characterised by having edges that connect more than two nodes [6] and multiplex graphs, which provide multiple edge sets [7]. The most basic extensions, however, are weighted and signed graphs, which assign (possibly negative) weights to the individual edges, which allow to express additional information about the edge, e.g. how strongly correlated signals are across the nodes that are connected by the edge.

Choosing the graph flavour is highly problem specific and depends on the relations that shall be represented by the graph. Naturally, the methods for signal processing change according to the graph type chosen. In this work, we will focus on developing methods applicable to the novel model of signature graphs, which are characterised by edges that have a tuple associated to them, consisting of an edge

weight and a vector of signs [8]. Such signature graphs have been proven to be advantageous in the context of data featuring symmetries that can be described by involutions [9].

In particular, we will study applications of signature graphs in conjunction with compositional data, which is data that describes parts of a whole and occurs in various fields, e.g. chemical analysis, voting data and demographics. Due to the inherent constraint that all parts must add up to a constant and no part can be negative, the data is not living in a subspace of . This renders the statistical treatment of such data with standard tools impossible. In 1982, John Aitchison laid the foundations for dealing with such data in a sensible way by introducing a vector space structure tailored to compositional data [10].

We will start this work by giving a brief introduction into the topics of compositional data and graph signal processing (Chapter 2). Thereafter, we will delve into the issue of involutions in compositional data (Chapter 3). Not only will we state parametric classes of involutions, but also explore how to estimate their parameters based on observed data. A semi-supervised and a blind method will be presented.

Chapter 4 is concerned with the classification of compositional data using signature graphs. We will explore how a signature graph can be learned from data, given the identified involution. Two equivalent methods of calculation will be stated and subsequently compared with regard to their computational cost. In the next step we cluster the learned graph considering the specifics of signature graphs and conduct a numerical study of the proposed methods in comparison with ad hoc methods. The chapter is closed with a remark on how to adapt the methods when dealing with multiple realisations of the data.

Chapter 5 proposes a framework for interpolation of graph signals on balanced graphs. We will assume to know the topology of the graph but not the edge weights and discuss a strategy to learn them from observations. A method for balancing an unbalanced graph is discussed in preparation for the actual interpolation that will rely on the balancedness to lower its computational burden. We conclude with a numerical analysis of the proposed methods.

2 Basics

This chapter is concerned with compiling various mathematical concepts that will be fundamental for the rest of the work. It aims to provide sufficient information for understanding the remaining work without getting lost in unnecessary detail. In Section 2.1, the framework for working on compositional data is introduced, largely based on the works of Aitchison [10] and on [11]. Section 2.2 is concerned with introducing graph theory, i.e., the treatise of datasets characterised by pairwise relations.

2.1 Compositional Data

The field of compositional data is concerned with the description of and operation on data that represents shares of a whole. Examples include electoral data where the sum of percentages per party is known to be 100 or the quantitative chemical analysis of a mixture whose mass components add up to the overall mass of the sample.

It is natural to describe the data as vectors with the dimension , which denotes the number of components. We have the additional constraint that the sum over the components equals a fixed amount . This depends on the chosen unit and is rather arbitrary. For instance, a poll result might be represented in percent per party or in number of voters per party, resulting in or being the number of voters, but both representations essentially convey the same information. Operations on the data should therefore only consider proportions of the entries rather than absolute numbers. Furthermore, it is sensible to disallow negative components, which are meaningless in the context of compositional data. In fact, it is common practice to only allow strictly positive entries in order to allow easier mathematical treatment.

2.1.1 Vector Space Structure

The data is represented traditionally by row vectors and lives in the so-called simplex which is formally defined as follows:

Definition 2.1 (Simplex):

From this definition it is immediately clear that using Euclidean geometry in connection with the simplex does not yield a vector space, as, for instance, the inverse element with regard to addition will have negative entries and thus does not live on the simplex . However, the simplex space can be elevated to a vector space by definition of alternative bivariate operations under which is closed. A popular choice of these operations constitute the Aitchison geometry, named in honour of its inventor, which will be treated subsequently [10]. First and foremost we will have a look at the closure, which is an essential building block for more advanced operations.

Definition 2.2 (Closure):

The closure is an operation that normalises the 1-norm of a given vector with strictly positive entries to . It is best understood as a projection of the vector in the first orthant onto the simplex along a ray, which intersects with the origin.

Based on the closure we can now define the two operations that constitute the vector space structure. First we will look at the equivalent of ordinary addition.

Definition 2.3 (Perturbation):

We can also equivalently write for every entry of

The perturbation can be viewed as an analogon to the sum in Euclidean space: Two vectors on the simplex are combined to form a new vector on it such that constitutes an Abelian group: Commutativity and associativity hold, a neutral element exists and there exists an inverse element for any element in . The neutral element can be shown straightforwardly to be — — , i.e., the elements of are

The inverse element of is given by

or equivalently by

We confirm this statement by evaluating the perturbation of an arbitrary vector with its inverse, i.e., with elements

As expected, every entry of the resulting vector equals —, i.e., the entries of the neutral element w.r.t. the perturbation. This also motivates the introduction of another operator that perturbs the first argument with the inverse of the second,

The second operation that constitutes a vector space has also two operands, however, one is a scalar. It can be thought of as the equivalent of ordinary multiplication.

Definition 2.4 (Powering):

The powering operation resembles a scalar multiplication in the Euclidean domain, as its properties are directly transferable (the distributive laws holds in conjunction with the perturbation operation, yields the identity operation, composition of subsequent powerings is equivalent to multiplication of the scalars and one final powering operation). For convenience, we define the order of operations such that powerings are evaluated before perturbations.

Additionally, it is even possible to define an inner product [11, ch. 3.3], thus turning into an inner product space.

Definition 2.5 (Inner product):

Consequently, it is possible to introduce the induced norm as

2.1.2 Transforms

The simplex as the domain of compositional data is an ad hoc choice based on our general understanding of compositions. However, as seen in the previous section, the mathematical treatment is rather complex and unintuitive. Multiple transforms to the Euclidean space have been discussed in the literature, of which two will be presented in the following due to their relevance to this work.

We have seen that the perturbation, which is analogous to addition in the Euclidean vector space, is based on entry-wise products of its arguments and the powering, which resembles a scalar multiplication, relies on taking entry-wise powers. Knowing that the logarithm relates products to sums and powers to multiplications, it seems natural to transform the vectors from the simplex by element-wise logarithms into a new domain. Additionally, we know that the logarithm is bijective, and therefore this transform is invertible. Augmented with a pre-normalisation factor we obtain

Definition 2.1 (Centred logratio transform):

The centred logratio (clr) transform relates every vector entry to the geometric mean over all entries and then applies the natural logarithm to each ratio afterwards. We can reformulate the entries of as

Therefore we can understand the clr transform as generalised log-likelihood ratio, specifically as the average pairwise log-likelihood ratio.

The inverse transform is given as

or equivalently

This function is well-known in the context of machine learning as softmax function.

As intended, the clr transform relates the vector space structure of the simplex described in the previous section to the Euclidean vector space: A perturbation in the simplex domain is equivalent to an addition in the clr domain and powering on the simplex to multiplication. Furthermore, the inner product and consequently the induced norm are related to their respective counterpart [11, ch. 4.3],

Another interesting property can be revealed by summing over the vector in the clr domain:

The sum is always zero, which means that the admissible vectors live in a hyperplane that is orthogonal to the all-ones vector and contains the origin. This subspace has dimensionality which is not too surprising, since the simplex itself is also a manifold of dimension .

Equipped with this knowledge, it is possible to construct orthonormal bases in the clr domain, i.e., for the described subspace of , and then transform them to the simplex domain using the inverse clr transform. They retain their property of orthonormality across domains due to the equivalence of the inner product (cf. (17)). The orthonormal clr basis vectors can be grouped into a matrix where the row sums are zero and consequently.

This motivates the introduction of the isometric logratio (ilr) transform, which builds on the clr transform and an orthonormal basis [12].

Definition 2.2 (Isometric logratio transform):

As the basis can be chosen arbitrarily (within the mentioned limits), the ilr transform is not really one transform but a whole set of transforms. It is viewed best as a coefficient expansion of the compositional vector with respect to the chosen basis. Thus, it can be computed equivalently with operations of the Aitchison geometry,

The inverse ilr transform can thus also be written either by means of the inverse clr or as a linear combination in the simplex domain:

As the ilr transform is related to the clr transform only by an orthonormal basis expansion, its properties translate well to the ilr transform,

2.1.3 Endomorphisms

Based on the importance of matrix multiplication in Euclidean spaces, it is sensible to look for an analogous operation in the realm of compositional data. Particular interest lies in endomorphisms, i.e., mappings from to . There are two approaches to this, one operating directly in the simplex domain and one exhibiting the ilr representation, thus being dependent on the choice of the basis . The latter one is presented first.

Definition 2.1 (Endomorphism via ilr):

Another approach is to take inspiration from the conventional vector matrix product and translate it to the basic operations of the Aitchison geometry, i.e., replacing sums by multiplications and multiplications by powers:

Since there is no guarantee that , a subsequent application of the closure on is necessary.

Definition 2.2 (Direct endomorphism):

Working in the simplex domain however comes at a price: Different choices of lead to the same operation, e.g., the operation is invariant to the addition of constant rows to the rows of . This happens because this addition only introduces a scaling of which is cancelled out by the closure.

The two endomorphisms are closely related to one another. In fact, it is possible to find a suitable for every , so that the operations are equivalent.

This derivation makes use of the fact that multiplying the all-ones vector with yields the zero vector, as the column sums of are zero by construction.

Be aware of the fact that the inverse statement is not true: A direct endomorphism can not always be represented by an endomorphism via ilr, because it is not necessarily possible to decompose a generic matrix into the product . This is obvious, because named product has at most rank , as , but can even have a rank of , due to its larger dimension. The condition when the decomposition is possible can be made even more precise: Since has row sums of zero, can only have row and column sums of zero, as well. Thus has to fulfil this criterion in order to be decomposable and to express the associated operation in terms of the ilr.

2.2 Graph Theory

Graph theory is one of the main topics in discrete mathematics. It deals - as the name implies - with graphs, which are structures, that model pairwise relationships between a set of arbitrary objects [4, ch. 1.1].

Definition 2.1 (Graph): A graph formally consists of a set of vertices (also called nodes) and a set of edges ,

The edges express pairwise relationships between vertices. The definition stated above addresses specifically undirected simple graphs, i.e., graphs, whose edges have no assigned direction and connect two distinct nodes (no loops). Thus far, we can only make a binary statement about the relationship of two nodes – either they are connected by an edge or not. It is, however, often beneficial to express the relationship between nodes on a more fine grained spectrum. Traditionally, this is done by assigning a non-negative weight to each edge, which can be interpreted as strength of the link between the two vertices it connects, as a distance between them, or as a measure of similarity. The weight function is used to describe the edge weights.

In recent years, however, the concept of graphs with negative edge weights gained traction, so-called signed graphs, which can not only express strong similarities between vertices but also strong dissimilarities or opponents. Their advantage has been proven in countless scenarios [13]. The extension to signed graphs has led to the definition of the concept of balancedness by Harary [14]. It expresses whether every cycle (a list of edges, such that consecutive edges – as well as first and last one – share a common node) in the graph has an even number of negative edges. When, for instance, the graph describing friend and foe relations with positive and negative edges respectively, is balanced, we can view this as a generalisation of the proverb "the enemy of my enemy is my friend". As we will see later, balancedness has strong implications for classification and interpolation tasks.

An even more recent development considers graphs with vector valued edge weights, called multiplex graphs [15]. In particular, this work will deal with signature graphs, whose edge weights underlay an additional constraint: An edge vector is only admissible, if all of its entries share the same absolute value and may thus only differ in the sign [16]. Equivalently, these edge weights can be viewed as a tuple consisting of a positive real number and a binary vector from the alphabet . This allows to indicate the overall strength of a link with the real number while expressing the similarity/dissimilarity of the nodes regarding multiple features with little memory overhead. For our analytical description of the weight function, we will however persist on the former viewpoint,

For convenience, we will denote the scalar weight part by and the sign vector by

2.2.1 Algebraic Representation

Graphs can also be represented in terms of matrices [4, ch. 2.3]. A popular choice is the so-called adjacency matrix , with elements

In a graph of vertices there can be at most edges, but normally the number of edges is much smaller, which makes this matrix sparse.

As the adjacency matrix only represents the structure of the graph, but not the edge weights, it is sensible to replace the ones in the adjacency matrix with the corresponding weights. This leads to the weight matrix with elements

Since only undirected graphs are considered, the adjacency matrix and the weight matrix are symmetric.

Note that the diagonal elements of and are all zero, as no edges from a vertex to itself are allowed. The degree matrix on the other hand is a purely diagonal matrix whose values are chosen as the sum of all absolute edge weights incident on a vertex,

This allows the definition of the Laplacian matrix which plays an important role for many graph related tasks,

Transferring the concept of the weight matrix to signature graphs yields a three-dimensional tensor. It can be viewed as layers of weight matrices that all exhibit the same sparsity structure. The degree matrix on the other hand is equal for all layers by construction. It is thus possible to define different Laplacian matrices – one for each layer.

Another matrix description is given by the incidence matrix , whose columns each describe one edge . It is element-wise defined by

Interestingly we can calculate the Laplacian from by .

2.2.2 Learning

Learning or constructing a graph means building up a graph from a set of data points, i.e., considering them as vertices and installing edges between them, typically based on some defined distance function [17]. The most common schemes are the -NN construction, where every vertex is connected to its nearest neighbours, and the -NN construction, where every vertex is connected to all other vertices that lie no farther away from it than some radius . These methods rely on the calculation of all pairwise distances between the nodes and thus have a complexity of . More efficient algorithms were proposed, like [18], where only an approximation to the -NN graph is computed, but with a

significant lower complexity. It is sensible, to also assign weights in this process, for instance the calculated distance between nodes. Especially when learning signature graphs, however, it is necessary to distinguish between the function that is used in assessing the distance between nodes and the function for assigning edge weights as they operate on different image sets (vs.).

2.2.3 Classification

In the context of graphs, classification or clustering is the operation of partitioning the set of graph vertices into disjoint subsets to , such that the following two criteria are met [19]:

In other words, every vertex of the graph is assigned exactly one label from 1 to . Furthermore, this dissection should be optimal with respect to a predefined criterion. For example, one could demand to minimise the summed up weight of edges between vertices of different sets, yielding the so-called minimum cut [20]. In the realm of signed graphs it might be beneficial to construct a cost function, that penalises positive edges between different subsets and negative edges within subsets. Indeed, if the graph is balanced, a perfect solution for can be achieved in the sense that only negative edges are cut [21]. Often, it is necessary to also exclude trivial solutions like empty subsets, e.g., by rewarding balancedness of the subsets' cardinalities.

To simplify the minimum cut problem several relaxations were proposed, of which the so-called spectral clustering techniques are very popular [22]. The method we will use is based on the eigenvectors of the Laplacian associated to the smallest eigenvalues. For two clusters and a non-signed graph, it comes down to assigning each vertex to either of two groups based on the sign of the corresponding entry in the second eigenvector (sorted ascending by the corresponding eigenvalue). For signed graphs, the first eigenvector is used. For signature graphs it is reasonable to apply this method on every layer, i.e., based on every . This yields different binary labels for every vertex and thus a partition into clusters [16].

2.2.4 Graph Signals and the Graph Fourier Transform

Up to this point we have predominantly dealt with numerical quantities on the graph in the context of edges, i.e., edge weights (the exception being learning where we start off with nodes derived from datapoints). In the following we assign (possibly vector valued) quantities to the every vertex. This assignment constitutes a so-called graph signal.

For example every node could be representing one sensor in a sensor network and would be connected to the nodes that correspond to the spatially nearest sensors by edges. When we take a measurement across the sensor network, we would get one signal value per node, i.e., a graph signal. Such a graph signal can be represented by a matrix where every row corresponds to a node of the graph. In the case of scalar signals the matrix degenerates to a vector .

Driven by the convenience of the Fourier transform in signal processing over regular domains, we look for an analogon in graph signals [4, ch. 3]. Recall that the Fourier transform is a basis expansion with orthonormal basis vectors which are sorted in an ascending order w.r.t. the frequency they represent.

Thus, we are also looking for an orthonormal basis of the graph signal, however we have to specify how the concept of frequency shall translate to graph signals.

One way to understand frequency of a time series is to look at the similarity of adjacent signal values: If the signal values are close to each other, we call the signal low frequency and if big changes in signal value occur between neighbouring samples we speak of a high frequency. To asses this property in graph signals we can express the variation as the sum over the squared signal value differences across edges, weighted with the edges weight,

It can be shown that this is equivalent to the quadratic form over the Laplacian,

However, there is a flaw with this formulation, which does not comply with our understanding of frequency: The signals variation changes when scaling the signal. A normalisation based on the graph signals energy solves this problem, yielding the Rayleigh quotient

———

Knowing that the Rayleigh quotient computes the eigenvalues of if we plug in its eigenvectors and that said eigenvectors are orthogonal, motivates the use of the eigenvectors as basis vectors for the graph Fourier transform (GFT). The corresponding eigenvalues then constitute their frequencies. We will denote the matrix whose columns correspond to the normalised eigenvectors of sorted ascending in terms of the corresponding eigenvalues by . Note that for unsigned graphs, as is positive semidefinite all frequencies are non-negative and the first is exactly zero, corresponding to some sort of DC signal. In fact, it can be shown that the corresponding eigenvector is the all-ones vector . These properties match very well with our intuition about frequency and the Fourier transform, which is now formally defined as

with its inverse being

2.2.5 Interpolation

As we are blessed with a variety of convenient signal processing techniques in regular domains (time series, images etc.), we might look for analoga applicable to the graph domain. Interpolation is one of the concepts that we aim to transfer and describes the reconstruction of a graph signal from incomplete data, i.e., unknown entries in . As this would be impossible without additional constraints, it is common practice to enforce some sort of signal smoothness for the reconstructed signal. We call a graph signal smooth when the values of connected nodes are similar.

Interpolation is not one method but a whole class of methods that act on incomplete graph signals. These methods can be grouped into two distinct categories, local and global methods respectively: Local methods estimate a missing value solely by its neighbouring nodes, while global methods incorporate the whole signal into the estimation. The former clearly are computationally favourable but perform worse than the global methods in general due to the missing information [23]. Another

categorisation of methods stems from how they deal with the existing values: They may either be considered to be the ground truth or corrupted by noise. In the latter case, the interpolation method also performs some denoising resulting in a graph signal that in general does not retain its values at observed nodes.

Formally, we can express the observed signal as a matrix multiplication of the complete, yet unknown graph signal with a selector matrix that dismisses the not observed data,

with the selector matrix built-up from the standard basis vectors corresponding to the observed nodes ,

Here, is the number of vertices that are sampled.

Signal smoothness can be straightforwardly achieved by enforcing that the reconstructed graph signal lives in the subspace spanned only by the first eigenvectors of the Laplacian, which correspond to small signal variations, i.e., some measure of smoothness, as we have seen in the previous section. To perform this bandlimited interpolation we introduce the matrix which contains only the first columns of . The reconstructed signal is expressed by

If we select as many basis vectors as we have observed values, , a reconstruction is possible such that the reconstructed signal matches the incomplete signal at the observed nodes, i.e., no denoising takes place.

Note that the graph Fourier transform has only entries, whereas has entries. As we want to enforce equality of the observed signal and the estimated signal we can write,

which leads to the Fourier domain estimate

Thus the reconstructed signal is given by

When we have more samples than degrees of freedom. Thus, we resort to approximate equality in the least squares sense. The solution looks similar to (46), however, the inverse is replaced by the left pseudoinverse [24],

Another way to face the task of interpolation is by setting up an optimisation problem that finds a signal that is maximally smooth while simultaneously explaining the observed signal as good as possible [4, ch. 4.3.5]. This can be formally written as

with the yet to be determined penalty function , which basically represents a measure of non-smoothness and a factor which adjusts the relative importance of smoothness versus closeness to the observation.

A reasonable proposal for the penalty function is to use a weighted sum over the squared differences between adjacent nodes,

The idea behind this is to use the difference of the nodes when the edge is positive, but at the sum for negative edges. We can also express this function using the incidence matrix,

Using this penalty function, we arrive at the so called Laplacian reconstruction,

which is a quadratic optimisation problem and can be solved analytically by

—

3 Involutions in Compositional Data

An involution is an endomorphism, that yields the identity map upon composition with itself. A simple example in the real domain is the sign flip, which yields the original number, when applied twice. As shown in [9], signature graphs provide a powerful framework in connection with involutions. In this chapter we will, therefore, discuss classes of involutions on compositional data, namely the linear, the affine, and the nonlinear involution. Then, we will concentrate on affine involutions, which are parametrised, and study two methods on how these parameters can be estimated.

3.1 Motivation from

To find involutions for compositional data we will draw some inspiration from known involutions in . The simplest involution in is the matrix product with an involutory matrix.

Definition 3.1 (Involutory matrix): An involutory matrix is a square matrix that is inverse to itself [25, ch. 3.1],

As the matrix is self inverse, twofold multiplication of it with a vector is equivalent to the unity map,

It can be shown that the eigenvalues of such an involutory matrix are either or . Hence, the matrix is diagonalisable into a so-called signature matrix , which is a diagonal matrix with only on the main diagonal,

The columns of are eigenvectors of .

3.2 The Skewed Logratio Transform

The most straightforward way to find a class of involutions for compositional data is by using involutory matrices in the ilr domain (cf. Definition 2.1). The interplay between the chosen basis and the involutory matrix will be examined in the following:

It is helpful to understand that is the right pseudoinverse of since . Now we introduce with its right pseudoinverse as

The involution can the be written more compactly as

inherits some properties of by construction: It is a by matrix, whose row sums equal zero and it has full row rank. There is however no guarantee for orthogonality of , let alone orthonormality. Thus, the rows of represent a skewed basis in general. Likewise, has column sums of zero by construction and has dimension .

The structural resemblance of (58) with (56), which was based on the ilr endomorphism, justifies the relaxation of the ilr transform to its superset, which we will call skewed logratio (slr) transform.

Definition 3.1 (Skewed logratio transform):

Its inverse can be calculated in analogy to the inverse ilr transform as a linear combination of the basis vectors in either the simplex domain or the clr domain,

3.3 Classes of Involutions in Compositional Data

3.3.1 Linear Involutions

As a result from Section 3.2, every linear involution in Aitchison geometry can be simply described as element-wise sign flips in the slr domain, when an appropriate oblique basis is chosen:

As seen above, the slr transform is a linear transform of the clr transform, but unlike the ilr transform it is - due to its oblique basis - not an isometric transform, i.e., distances are not preserved. Still, it is possible to relate the inner product and norm in the slr domain to operations that are directly applied in the simplex:

Thus, we can define the norm as

Apparently, the Gramian of has to be taken into account. The derivation makes use of the fact that it is possible to split up every into , but the final result does not depend on this decomposition.

From a computational perspective it would be interesting to get rid of the asymmetry present in (62). Luckily, is symmetric and positive semidefinite by construction as a product of two matrices that are transposed versions of each other. Therefore, its square root is defined, real and can be computed from the eigendecomposition as

Obviously, is symmetric and positive semidefinite by construction, too. itself has column and row sums of zero only. This implies that fulfils this property, as well. We can prove that by contradiction: Let us assume that contains rows, whose entries do not sum up to zero, i.e., . Exploiting the symmetry of we can then look at the norm of and see that

But this contradicts our assumption of . Analogously, we can argue about the column sums of . This allows us to formally express as the product

Now, we can reformulate the inner product as follows:

Exploiting the identities from (62) we can rewrite the inner product as

The norm is then given by

We also want to describe the linear involution without switching domains, i.e., only with operations that work on the simplex representation of the data, which leads to

To show that (70) is actually an involution, we will calculate , but to ease the calculation, we will substitute by

This is no restriction to generality as every vector on can be represented like that (note the conceptual similarity to the clr, however we have omitted the closure). We can then apply the involution,

Now, let us apply the involution another time,

Let us take a closer look at the expression : It constitutes the th column of the matrix product

We have used the fact that constitutes a projection matrix that projects into the row space of . Since we know that has full rank and its kern is the span of , we can reformulate the projection as , where is the projection matrix onto ,

Now we can evaluate where denotes the th standard basis vector,

The second factor in this equation does not depend on , appears in every entry of the resulting vector and can therefore be omitted, as the subsequent closure normalises the result anyway. We can thus simplify to

It remains to analyse the product of the powers of , yielding

since we know that the column sums of are zero. Thus, we arrive at the desired result,

3.3.2 Alternative Derivation

We can also look for an involution in the clr domain, i.e., a class of involutory matrices that maps vectors that are orthogonal to onto a vector, that is orthogonal to , as well. Mathematically we can express this as follows:

The involution can be viewed as a three-step operation consisting of a basis expansion, potential inversion of coefficients and linear combination of the basis vectors according to the transformed coefficients. As we choose the diagonal entries of individually, it is necessary that the subspace , that shall remain intact, is represented only by one single basis vector. Otherwise, we would have to invert all coefficients that have part in representing together.

Without loss of generality we thus demand that the first rows of are orthogonal to , i.e., have a row sum of zero. This automatically enforces that the last column of lives in . The last coefficient is thus always zero, when starting out with a orthogonal to . Hence, we can neglect the last column of , the last row of and reduce the dimension of to , each represented with a tick:

This expression happens to have the same structure as (61) and is therefore equivalent. Looking back at (56) reveals that and as well as and are related by the ilr basis ,

3.3.3 Affine Involutions

The linear involution presented above can be understood as mirroring points on the simplex in a set of coordinates around the neutral element — — . It can be slightly augmented to become an affine involution, which can be understood once again as mirroring operation, but with respect to another centre point , i.e., the origin of the coordinate axes is shifted to :

One can easily see that if is an involution, has to be also involutory,

3.3.4 Nonlinear Involutions

Motivated by the structure of the previously introduced involutions, we can observe that complex involutions can be constructed by using bijective transformations and a signature matrix. Let us consider the possibly nonlinear, but bijective mapping with its inverse . Let us furthermore assume that the set is closed with respect to matrix multiplications with . Thus, an involution can be constructed as

It is easy to show that is indeed an involution by composition with itself,

The mapping can be interpreted as a deskewing operation, that enforces axial symmetries around the origin.

Note that the linear and the affine involution are special cases, where the transformation is a linear or an affine function respectively – hence their name.

Finding a plausible involution underlying the observed data presupposes prior knowledge about the data, e.g., some basic understanding of underlying symmetries or a statistical model of it. In the following, we will concentrate on the class of affine involutions, for they are analytically treatable and serve as an example for presenting a methodology that may be translated to nonlinear cases.

3.4 Identification from Labelled Data

Identifying an involution is mandatory before any graph can be learned. In case of the affine involution this comes down to finding a reflection point and a suitable basis . In general, these have to be learned from labelled data. In the context of signature graphs, the possible labels are the binary vectors , with the index specifying the label. Thus, the number of clusters is upper bounded due to the dimension of the data.

3.4.1 Data Model

In order to derive a method for estimating the parameters of the affine involution, it is helpful to postulate a statistical data model:

We assume that the support of is roughly contained in one orthant.

Looking at this model in the clr domain reveals an affine structure,

Without loss of generality, we assume . This is justified by the fact, that we want to estimate , which can incorporate arbitrary scaling. Thus, the model can be split up further as follows, with denoting the random fluctuation of around :

In preparation of the estimation task, let be a matrix whose rows contain labelled data points in the clr domain, the matrix with the corresponding labels and the realisations of the random fluctuation:

3.4.2 Least-Squares Estimation

A naive problem formulation for the application of least squares is given by

All parameters that we want to estimate are grouped in one matrix that enters the equation linearly. The noise is skewed, but this is usually not a problem, as we can apply weighted least squares, a method specifically targeting such cases. However, there is a flaw: We can not make any assumption about the

noise covariance, in order to set the weights of a weighted least square estimation, because it depends on , which we want to estimate.

In order to apply a meaningful least squares estimation to the problem, the model has to be reformulated, so that the noise part appears isolated, i.e., without postmultiplication by . For that reason we postmultiply by its right pseudoinverse in order to get

where

Instead of estimating and directly, we will estimate and , which can be transformed to the former quantities straightforwardly. Minimising the norm of yields the well-known linear least squares estimator:

However, this reveals that the problem is ill-posed, since the inverse is not defined. Due to the fact, that the rows of are valid vectors in the clr domain, the respective row sum is always zero, which makes column rank deficient by construction. In other words, the solution to the minimisation task is not unique, since adding arbitrary constant columns to the columns of a given solution does not impair its optimality.

Hence, we can place additional constraints on the column sums of , which is quite fortunate. Up until now we have ignored the fact that has to be a valid slr basis, i.e., that its row sums must be zero. This translates to its right pseudoinverse having only column sums of zero by construction. The opposite statement is true as well, making the constraint necessary and sufficient for a valid solution.

Now, we need to find a way to incorporate the constraint into the problem formulation. We can formulate it in analogy to our model as

We can then add this as a row to our observation matrix and label matrix :

Performing the estimation task is now guaranteed to yield a valid solution, i.e. a , whose column sums are zero. Note that they are not approximately zero, but exactly, since we have constrained a degree of freedom of the previous ambiguous solution. The estimator in the new matrices is given by

For a numerical evaluation of the presented methods performance, please see Section 4.4.3.

3.4.3 Further Remarks

The basis that is to be estimated by the preceding equations shall have linearly independent rows and therefore has linearly independent columns. Based on the estimation equation this is only possible if has full column rank (being rank deficient is unlikely as it incorporates random quantities).

However, this criterion is not strict enough to acquire a meaningful solution. We can show this by assuming the noiseless case, i.e., . The model for then degenerates as follows:

To uniquely identify and we must ensure that has full column rank. This criterion has two (non-sufficient) implications: The labelled data must at least include different labels and every column of has to comprise positive and negative ones. Especially for large this conjuncture is a huge improvement over previous identification proposals, that necessitate labelled data from all clusters [9].

When additional information about the statistical properties of is available, it might be beneficial to resort to weighted least squares in order to anticipate the possibility of having different coefficients of variation (CV) per entry, defined as

As we have forced , we have implicitly assumed that all are equal, when using ordinary least squares. If we use weighted least squares, however, we can consider different variances. Equivalently, we can change the assumption of to

Thus the implicitly assumed variances are all one, as shown by

and we can proceed by application of ordinary least squares. The label matrix has to be adapted by postmultiplication with , to reflect our assumption.

Apart from that, if the labelled data is corrupted by additive noise, it might be necessary to refrain from ordinary least squares and rather use methods of total least squares [26].

3.5 Blind Identification

In scenarios where we do not have any labelled data we have to resort to other methods in order to identify the involution. We will see that an estimation of the involution is still possible under rather vague assumptions about the correlation inherent in the data. Specifically, we will presuppose an unsigned graph derived from our knowledge about the targeted problem and assume that connected nodes exhibit a correlation. Imagine, for instance, a sensor network. The corresponding graph may be derived by -NN construction based on the spatial distances of the sensors. In such a scenario it is sensible to assume that sensors close to each other yield correlated readouts, given that the measured quantities exhibit a spatial correlation and the correlation distance is smaller than the sensor distances.

3.5.1 Model assumptions

In order to formalise the ideas above, we start out by establishing an unsigned graph with . A vector valued signal on that graph can then be expressed by the matrix

The th row represents the signal value at the th node. We furthermore assume that the signal values at every node are derived from uncorrelated sources in the slr domain,

Furthermore we expect a correlation between signal values in the form

In other words, we are looking for an slr representation of our observed simplex-valued graph signal such that is uncorrelated across layers and adjacent signal values are correlated only on a per-layer basis. This correlation of adjacent nodes can be either positive or negative, depending on the diagonal values of . One can think of these correlations as colouring of the underlying random process.

We are interested in estimating , but for the sake of simplicity we will assume that we have transformed our observed data to the ilr domain by some arbitrary basis and only estimate the basis instead of directly. This approach is justified as we can represent every valid skewed basis by this product as long as is a valid ilr basis (cf. Section 3.3.1). Thus, the graph signal in the ilr domain becomes

with

For the sake of notational simplification we will substitute by in the following:

– —

3.5.2 Problem formulation

We are now looking for a basis that explains the observed data best under the assumption of uncorrelated entries in ¯ . Or, to put it differently, we want to recover the uncorrelated sources that are mixed by multiplication with in (106), which is equivalent to finding and multiplying our observations with its inverse. This problem has gained much attention in the recent years under the name of blind signal separation, as it arises for instance in MIMO communication systems [27, ch. 1].

A first step to tackle it is to look at the autocorrelations of the signal at node ,

– –

— —

Instead of yielding directly we can only estimate its Gramian by empirical estimation of the autocorrelation matrix over observations indexed by ,

—

For better accuracy and as is independent of it is also sensible to average the estimates over all nodes,

—

Decomposing the symmetric, positive definite matrix into , however, is ambiguous, which stems from the fact that any with being unitary is also a valid decomposition, as

A popular choice for such a decomposition is the Cholesky decomposition [28, p. 143], which yields an upper triangular matrix such that . If we have estimated by this method we have reduced the problem to finding a unitary matrix such that . In the literature, there exist two approaches to solving this problem: Either we assume that the sources are non-Gaussian and use higher order statistics or we exploit the fact that the sources are non-white, yet independent processes [27, ch. 1.3]. We will choose the latter route, since our model is inherently non-white as argued above.

Under the assumption that our estimate is close to the true correlation matrix we can whiten the observed data by postmultiplication with the inverse of ,

– –

–

—

—

Therefore, ¯ can be understood as linear combination of the orthonormal rows of , with the coefficients being derived from individual sources. Due to the orthonormality of the rows we have already decorrelated/whitened the sources by this step. There is nothing more to be gained by looking at the autocorrelation.

Let us now have a look at the cross correlations between signals of adjacent nodes:

– –

— —

for . The signature matrix can assume different values as every of the diagonal entries can either be or , but as and yield the same up to a sign flip and being also only a scalar factor we expect at most truly distinct matrices .

Left and right multiplication of (112) with and respectively yields

Although we do not know the exact value of the right-hand side of (113), we know that it has to be diagonal. Therefore our problem of estimating , given the cross-correlation is equivalent to searching for a that diagonalises every .

Note, however, that this criterion leaves still ambiguous with regard to a permutation matrix and a sign matrix as both do not change the structure of the right-hand side of (113) when premultiplied to a valid . This fact does not bother us, because a permutation and possible sign flip of the entries in still leaves the sources separated. Furthermore, we need to encounter cross-correlations that correspond to pairwise linear independent matrices for a meaningful estimation [27, ch. 7.3].

3.5.3 Joint Diagonalisation

As we only have access to estimated cross-correlation matrices , we can not expect to find an orthonormal matrix that exactly diagonalises all jointly. A common expedient is to resort to approximate joint diagonalisation [27, ch. 7.4]. To formulate the underlying optimisation problem, let us first introduce the operator. It operates on a matrix and yields the sum over all squared off-diagonal elements,

When is zero, is perfectly diagonal. By this argument, we are interested in minimising jointly for all . As yields non-negative numbers, an obvious way to express the joint minimisation is to minimise over the sum,

A proven method to perform this optimisation in an iterative manner is based on the Givens rotation matrix [28, p. 215] , which can be defined element-wise as

The Givens rotation matrix describes a rotation by in one plane determined by and . The matrix is orthonormal and we can express every orthonormal matrix as a composition of subsequent Givens rotations in different planes. The idea is now to iteratively pre- and postmultiply the matrices we want to diagonalise by and until we reach a satisfactory diagonalisation of all cross-correlation matrices. Due to the structure of the Givens rotation, every rotation only affects two off-diagonal elements and is parametrised by a single scalar quantity . Therefore, it is easy to calculate the ideal in every iteration step.

Let us denote the matrices we get in each iteration step by . As starting value we set as these are the matrices we want to diagonalise jointly. In every iteration step we select a pair by sweeping over all possible combinations with . We can now calculate the product

The elements of are given by

Only two off-diagonal elements are affected by the rotation in the th iteration step. They evaluate to

As no other diagonal elements are affected by the th Givens rotation, the minimisation in the th iteration step reads

For solving this optimisation task we introduce a substitution to decrease notational complexity:

Inserting these substitutions and some trigonometric relations, we can rephrase the optimisation problem as

To optimise for we calculate the derivative and set it to zero:

We have divided by to get the last identity in (124). This is reasonable because we expect the solution to converge to over the iterations and thus divide by a non-zero number. Now we are confronted with a quadratic formula in which has the solutions

As one solution presents a maximum we have to select the correct one carefully. Instead of evaluating a second derivative of the objective function or inserting the solutions back into the objective function, we will resort to a more elegant though less rigorous argument: Towards the converged solution we expect only small angular values of and also small values for off-diagonal elements, i.e., and become small as they are quadratic functions in the off-diagonal elements. Furthermore, is linear in the off-diagonal elements and thus is expected to decrease as well, tough not as strongly. The possible solutions thus are approximately given by

Clearly choosing a minus sign in front of the square root gives a large result for in terms of magnitude and thus an angle near for . We expect a small angle, though. Hence we choose the solution with the added square root.

Realising that

leads us to our final result,

Having found a recipe for calculating the optimal in every iteration step, we now can state the whole iteration: The initialisation reads

The th iteration step can be performed by a Givens rotation with angle

The values of and should be swept over all admissible pairs . The algorithm converges towards no additional rotations, i.e., . A reasonable stopping criterion is to check if every of the last sweep over and has fallen below a predefined threshold.

The matrix of the last iteration is then the cumulative composition of all rotations that were performed to approximately diagonalise all empirical correlation matrices. Our estimate of the basis is thus given by

And for the slr basis matrix we get

For a numerical evaluation of the method see Section 5.4.

4 Classification

In this chapter we will look into the problem of clustering compositional data using signature graphs. The first step of learning the graph from the data will be formulated in the simplex, as well as in the slr domain. Next we will evalute, which method is of computational advantage by counting operations. Having learned the graph, we deal with the particularities of clustering balanced graphs in Section 4.3. The chapter is closed by a numerical study to show the advantage of the proposed framework over ad hoc methods.

4.1 Learning the Graph

As highlighted in Section 2.2.2 common graph learning algorithms need to assess some sort of distance between data points. Therefore, it is our task to formulate a bivariate function, that yields a (typically non-negative) value from an ordered set, which is then used by the learning algorithm to decide whether to place an edge between any two vertices or not. However, in the case of signed and signature graphs this distance cannot be used as a means to determine the weights of the newly placed edges. Thus, we need to design a separate bivariate function, that yields values from .

Let us first reiterate the family of involutions (cf. (83)). A reasonable foundation for both, the distance function and the edge weight function , is to find the involution , that minimises the distance between and :

Note that, although we recycle the symbol , it shall not be interpreted as a matrix that denotes a specific label, but rather as a description of the relationship between the labels of two clusters. Based on we can then introduce the edge weight and distance functions:

The selected learning scheme, makes use of in order to asses whether to place an edge between any two nodes. In the case of -NN learning, for instance, every node is connected to its nearest neighbours based on this distance metric. If the learning scheme decides to place an edge is used to calculate the edge weight. As we want to construct a signature graph, every edge weight must be composed of a scalar weight part and a sign vector. Using the inverse distance as the scalar part is reasonable, when viewing the edge weights as a measure of (dis)similarity. The sign vector on the other hand describes which involution has explained the relation between and best. It is given by the diagonal elements of .

4.1.1 Reformulation in the Slr Domain

As we chose the -norm for assessing distances, it is possible to reformulate the equations from above easily in the slr domain with basis . We therefore introduce the slr counterparts for the offset vector , as well as the two vectors we want to compare,

Applying these relations to (133), we get

As we have shown in (63), a -norm in the simplex domain can be related directly to a regular 2-norm in the slr domain, which entails

4.2 Computational Considerations

In this section, we are going to investigate, which of the presented approaches shall be taken as a basis for graph learning from the standpoint of computational efficiency. The main question we want to answer is whether the distance computations shall be performed in the simplex domain or in the slr domain. The -NN or -NN learning [17] are expected to have quadratic complexity in the node count, as the distance between any pair of nodes has to be calculated. This emphasises the importance of fast distance calculations. Since the prior identification of the involution typically takes only a fraction of the data into account, we can and will neglect its impact on computation time.

4.2.1 Simplifications

It can be seen that - due to the affine structure of the involution - the offsets and are subtracted from the datapoints. Therefore, it is beneficial to compute this difference once for every datapoint up front instead of doing it for every possible distance. We mark the quantities after removal of the offset by a tick, both in the simplex and in the slr domain,

Applying this substitution to (133) and (136), we get simplified expressions for the norm in both domains,

where we used

To minimise the respective norms, one could calculate the norm for every possible , but such an exhaustive search is not necessary in the slr formulation. The additive structure of the norm allows splitting the minimisation into independent minimisations – one for every vector entry:

We have used the fact that the quadratic function is monotonic increasing for non-negative arguments. Thus, the minimisation can also be performed over the squared norm and since the resulting sum has only non-negative summands that are decoupled, every summand can be minimised individually. Because can only be chosen as or , the summands are minimised if and have the same sign. Thus, the distance-minimising signs are given by

where

Note that this sign function is modified to accommodate for the case of a zero as argument, in which case both choices of yield the same summand, making its choice arbitrary.

Unfortunately, the -norm does not exhibit such an exploitable structure in the simplex domain, leaving only the possibility of exhaustive search among different choices for .

4.2.2 Operation Counts

We can now calculate how many elementary mathematical operations are necessary for learning the graph, i.e., to acquire the distance between any two vertices. The additional cost of the graph construction algorithm, for instance -NN (cf. Section 2.2.2), is not of interest in this comparison as it does not depend on how the distances were acquired.

The elementary operations are addition (or equivalently subtraction), multiplication, division, power, logarithm and boolean exclusive or (XOR). We assume the compositional data to be from and the number of data points/vertices to be .

The calculation can be split into two stages: First, every data point is preconditioned, e.g., by subtracting an offset. This is obviously done once per vertex and thus times. Afterwards, the actual distances are calculated, which will be performed ———, assuming the distance metric is commutative in its operands.

The preconditioning in the simplex domain consists of a subtraction in the compositional sense, i.e., a perturbation. Looking at Definition 2.3 reveals that multiplications have to be performed inside the closure and the closure itself amounts to additions and divisions according to Definition 2.2. This totals to multiplications, divisions and additions.

Calculation of the squared -norm equals the calculation of the inner product defined in Definition 2.5. At first glance it looks like logarithms have to be calculated, but once again we can exploit the structure of the problem to reduce the computational effort:

Thus, we would only have to take logarithms, perform subtraction, multiplications and additions per inner product, but as we were able to express the -norm as a symmetric inner product (cf. (69)), we can replace by in the Aitchison inner product for an even stronger simplification,

This leaves us with only logarithms, additions, 1 subtraction and products.

According to Definition 2.2, applying the direct endomorphism to amounts to powers and products inside the closure as well as additions and divisions for the closure itself. We will assume that all as well as are precomputed and thus do not include them into our record. For calculating the squared -norm we need to compute two of these endomorphisms, a perturbation, and an inner product. The total amounts per operation can be seen in Table 1. Bear in mind that the total amounts still have to be multiplied by , as the norm must be evaluated with all possible for minimisation.

Working in the slr domain poses a higher upfront cost, because every data vector needs to be converted to it first. As we can see from Definition 3.1 we have to compute the clr transform first. Once again, we can optimise the canonical Definition 2.1 for simpler computation. Recall that the clr transform made use of

The clr transform can then be rewritten as

This reformulation eradicated all multiplications and the power. We are left with logarithms, additions, one multiplications and subtractions per vertex. Additionally, is postmultiplied to the clrs, which amounts to another multiplications and additions. Finally, is subtracted from every vertex, which amounts to another subtractions. Table 2 compiles these results.

Operation	**Amount**	**Add/Sub**	**Mult**	**Div**	**Log**	**Pow**
Inner product	1					
Endomorphism	2					
Perturbation	1					
Total						

Table 1: Operation counts in the simplex.

Operation	**Amount**	**Add/Sub**	**Mult**	**Div**	**Log**	**Pow**
Clr	1					
Matrix product ( )	1					
Difference ( )	1					
Total						

Table 2: Operation counts for switching to the slr domain.

The computational costs for determining the optimum sign flip vector per node pair are exceptionally low. It suffices to apply the bitwise XOR to the sign bits of the slr vector entries. Calculating the squared norm adds the complexity of subtractions, multiplications and additions.

The final results are collected in Table 3 and show that working in the slr domain is truly advantageous due to mainly two reasons: The minimisation can be simplified drastically and complex operations like the logarithm and the power do not appear in the learning stage, which has quadratical complexity on its own, but only in the preparational step, that in itself has linear complexity.

Operation	**Amount**	**Add/Sub**	**Mult**	**Div**	**Log**	**Pow**	**XOR**
Upfront simplex							
Learning simplex							
Upfront slr							
Learning slr							

Table 3: Comparison of operation counts for simplex- and slr-based learning.

4.3 Clustering

The idea behind clustering signature graphs is to cluster every layer individually into two vertex sets. This can be thought of as annotating every node with binary labels, one for every layer. This gives possible combinations for labels and thus results effectively in a clustering with up to clusters.

As clustering in the case of a balanced graph is well understood [21], we will proceed to show that a graph constructed according to Section 4.1 is in fact balanced.

4.3.1 Inherent Balancedness

A signed graph is called balanced if the number of negative edges in every cycle is even (cf. Section 2.2.3). Balancedness is an interesting property in the context of clustering, as it is equivalent to stating that the vertex set can be divided into two disjoint sets such that every edge between sets is negative while every edge connecting two nodes in the same set is positive.

The property of balancedness was also translated to signature graphs. There it means that a graph is balanced if and only if every layer (which is a simple, signed graph) is balanced. Thus, we can prove the overall balancedness of the learned graph if we can show that every layer is balanced by construction.

The argument is understood best in the slr domain after subtracting : The sample space is , which means that all samples can be ascribed to one of orthants (if we neglect the case of data vectors with at least one zero entry, which often has zero probability anyway). As seen in the previous section, we can perform the minimisation, which determines the signature vector, element-wise,

This means the following: If the datapoints and live on different sides of the hyperplane that is defined by having the th vector entry equal 0, then the edge sign on the th layer is negative, or else positive. Therefore, we have a dichotomy of vertices, having either negative or positive values as th vector entry: Every edge connecting two vertices on the same side of the hyperplane is positive and edges that cross the hyperplane are negative. This makes the th layer balanced, i.e., every layer, and thus the whole graph.

These insights prove that classification based on the above graph construction yields no better results than classifying based on the orthant, in which any datapoint lies.

4.4 Numerical Experiments

The objective of this section is to validate the previously discussed strategies to deal with the classification of composite data based on numerical studies. A comparison with ad-hoc methods that neglect the structure inherent to compositional data will be performed.

4.4.1 Data Synthesis

To synthesise data we use the data model introduced in (87), i.e., we sample some distribution, multiply by the sign matrix, add an offset and apply the inverse slr transform. We chose a distribution derived from an independent, identically distributed (i.i.d.), normal random vector as follows:

—

This can be understood as a rescaling of the normal distribution, i.e., an augmentation of its variance, based on the direction of the sampled value. Thus, the sigma neighbourhoods do not form spheres centred around in the 2-norm sense but in the 6-norm sense. We define the signal-to-noise ratio (SNR) based on the variance of as .

The basis and the offset are also randomly drawn for every data set. The entries of are sampled from a uniform distribution in the interval . To randomly generate the basis vectors, we first sample a zero-mean i.i.d. standard normal distribution and then project these vectors to the hyperplane normal to in order to acquire valid basis vectors. To prevent numerical instabilities we will not allow that any pair of basis vectors exhibits an absolute cosine similarity of more than and no basis vector with a norm less than . These properties are enforced by rejecting sampled basis vectors that do not comply.

4.4.2 Exemplary Plots

The case is most suitable for visually representing the previously discussed data generation: We start out in the slr domain, i.e., after sampling 100 values from per cluster, multiplying them with the corresponding sign matrix from their right side and translating all resulting points by a fixed vector . The resulting plot is shown in Figure 1.

It is apparent that the four clusters exhibit a good separation in the slr domain. The implicit basis (indicated by the yellow vectors) is orthogonal, describing two axes of symmetry – however these axes do not intersect in the origin due to the deterministic offset . The 1σ, 2σ and 3σ neighbourhoods are highlighted in light grey.

For the sake of completeness, we also provide a plot in an arbitrary ilr domain, given by Figure 2. The basis was chosen to be

— —
– –
— — —
– – –

In this representation, the two vectors that represented the translated standard basis vectors in the slr domain, are no longer orthogonal. Accordingly, the point clouds are skewed.

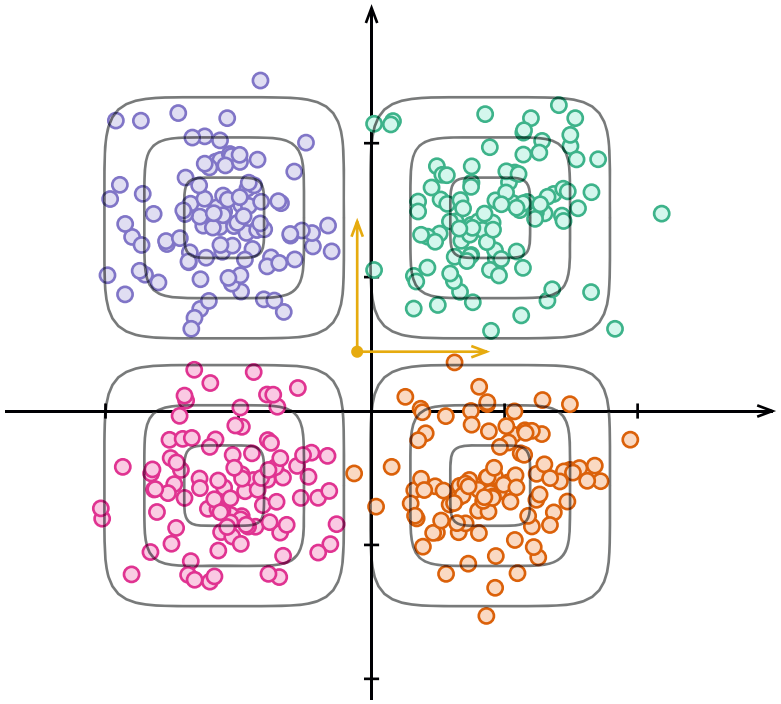


Figure 1: Data in the slr domain.

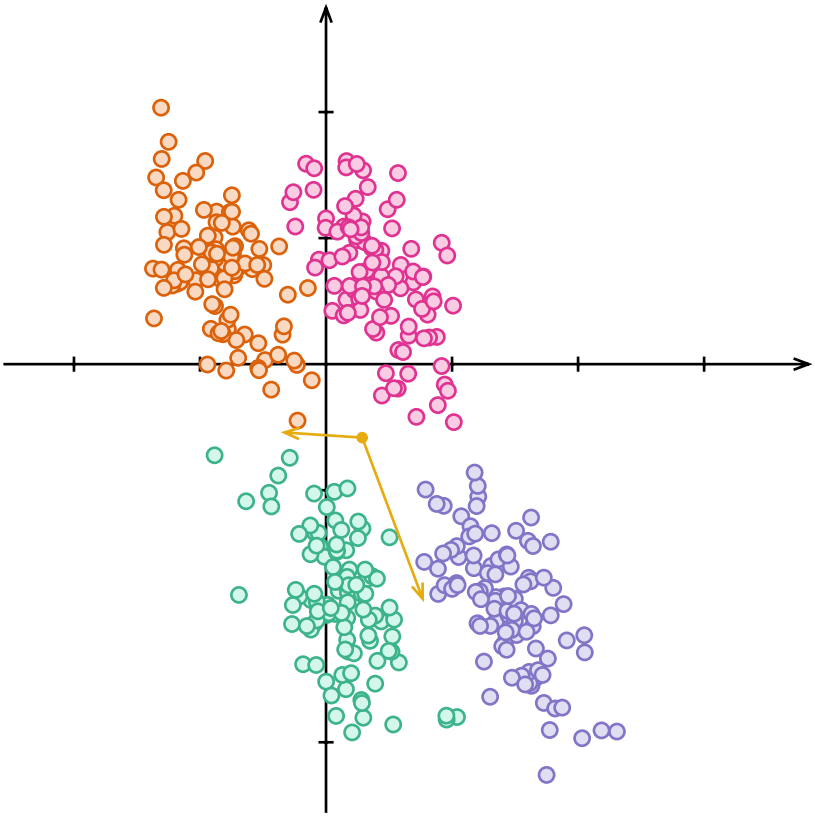


Figure 2: Data in the ilr domain.

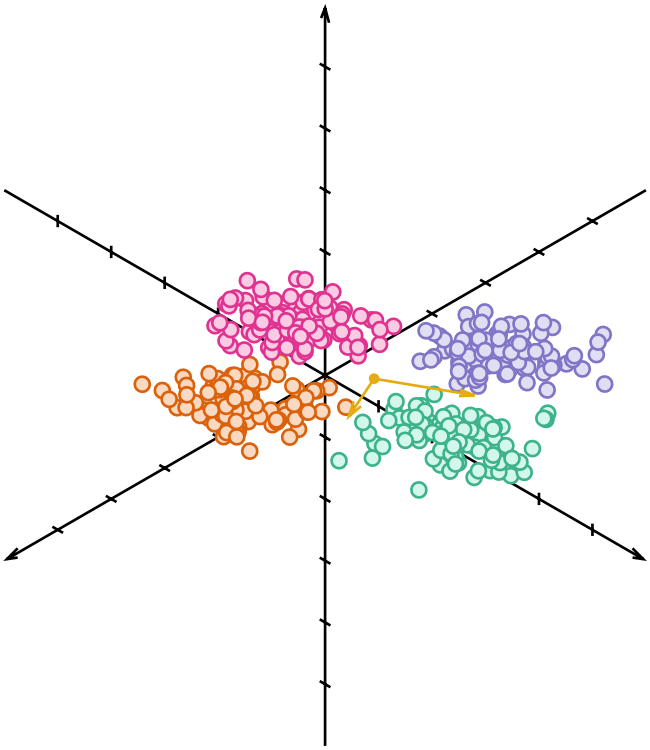


Figure 3: Data in the clr domain.

Looking at the same data in the clr domain also reveals a notable skew, as the chosen basis (denoted in yellow) is not orthonormal. All points lay in the plane orthogonal to . In Figure 3 we chose an orthographic projection that has this plane as projection plane. As all transforms up to this point were linear, the clusters are well separated, still.

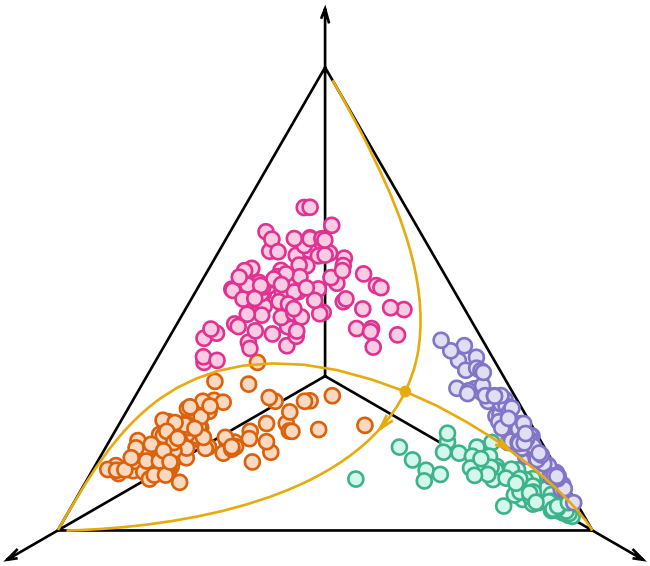


Figure 4: Data in the simplex domain.

Finally, Figure 4 shows the data in the simplex domain. Due to the nonlinear relationship between the clr domain and the simplex, scaling basis vectors in the clr – or powering vectors to speak in terms of the simplex – leads to the yellow curves, instead of straight lines. Especially towards the border of the

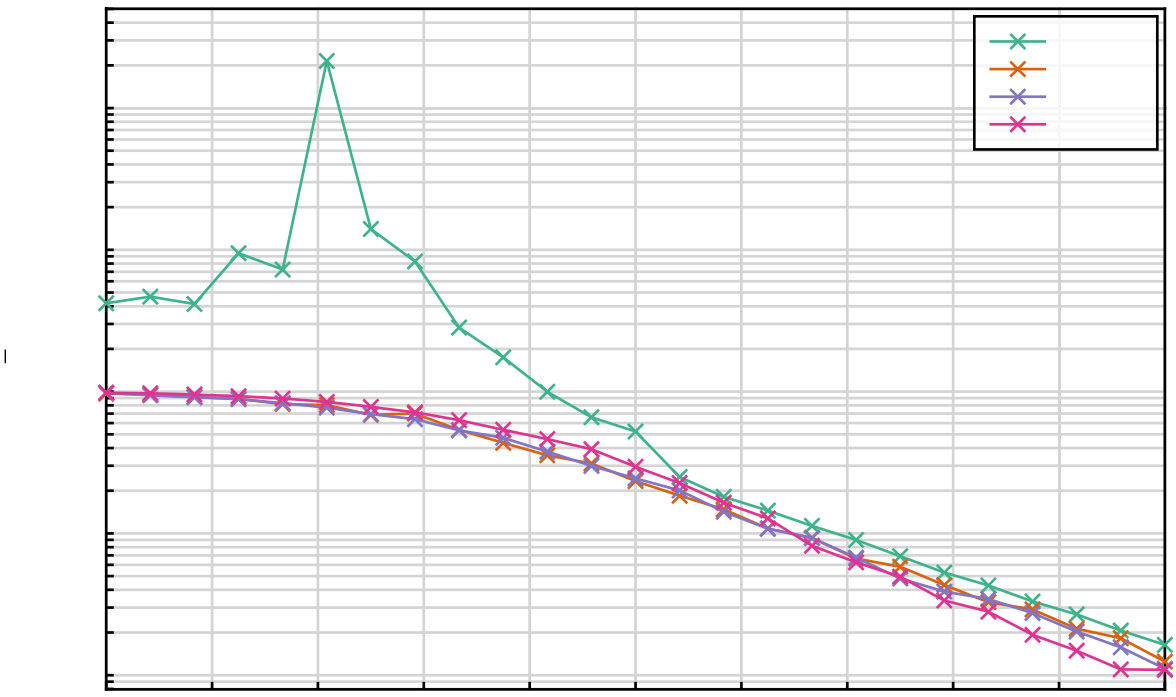
simplex, the density of the datapoints increases w.r.t. their pre-transformed counterparts, with fatal consequences: If the blue and green clusters were not marked differently, they could be misunderstood for being one joint cluster, if we would use the euclidean norm as distance measure.

4.4.3 Performance of the Involution Estimation

We will look at different variations w.r.t. the number of pre-labelled nodes and analyse the performance of the estimation described in Section 3.4 over different SNR values. As a performance metric we chose the normalised Frobenius norm and the 2-norm for the errors in and respectively,

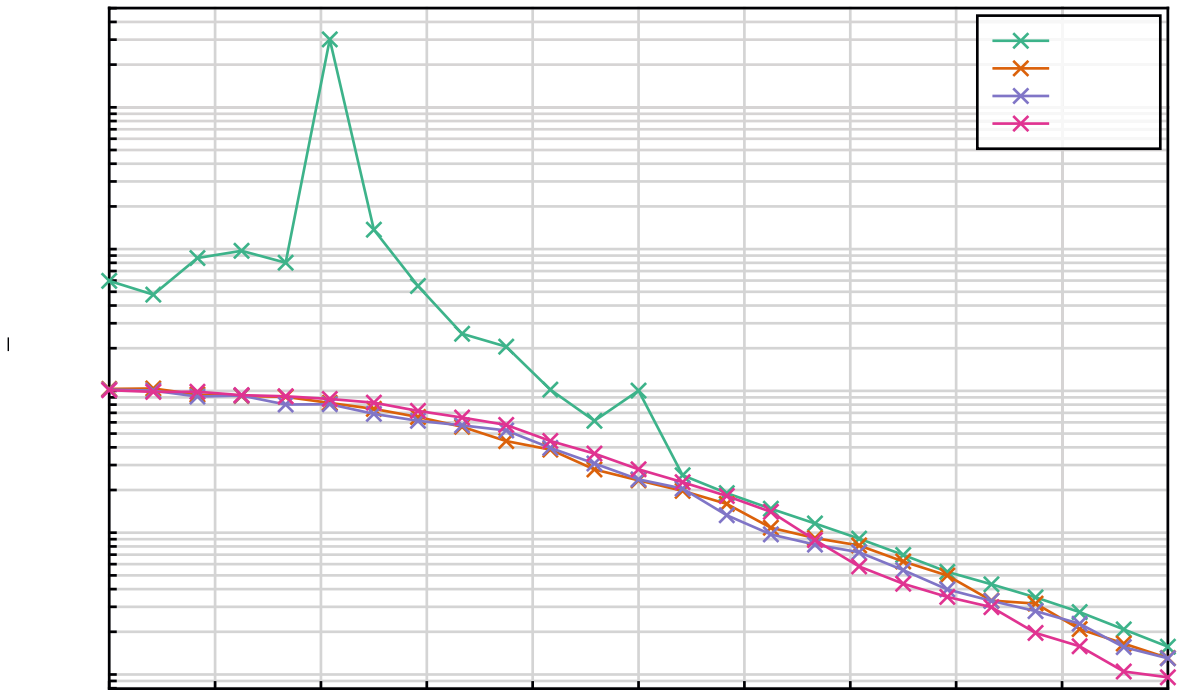
As the offset and the basis are chosen randomly, the resulting estimation errors are random variables, too. We will characterise them by their (empirical) mean in the following plots. The dimension was set to and the number of labelled data points is marked with . Only 4 distinct labels were used, each assigned to − datapoints.

Figure 5 and Figure 6 show the results. It is seen that the estimation performance improves with increasing SNR. We observe severe improvement in the low SNR regime when going from to , whereas the number of labelled vectors plays hardly any role for the performance at high SNR values. The errors roughly decrease by an order of magnitude per 20 dB for high SNRs.



SNR

Figure 5: Mean estimation errors in .



SNR

Figure 6: Mean estimation errors in .

4.4.4 Comparison of Clustering Methods

We will look at different approaches to clustering data and their performance in terms of the misclassification rate. The data model of the previous section is used here as well to generate datapoints. We once again work with a dimension of and adopted the same definition of the SNR as before. The test data was generated for all clusters, with 100 points each, totalling to 800 datapoints per test set. For the identification, we randomly drew datapoints from the set, with the only condition of having full rank.

Four methods will be compared, all relying on -NN learning a graph with and subsequent spectral clustering. The first and most naive method is based on constructing an unsigned graph in the simplex domain, i.e., with the Euclidean norm as distance measure. The edge weights are the dampened inverses of the distances d, calculated by

The second method is a slight modification in the sense that the distance is measured in the clr domain, or, in other words, the Aitchison distance is used to assess the nearest neighbours and corresponding edge weights (once again dampened). In the third case, an unsigned graph was built using the datapoints transferred to the slr domain. The preferred method, however, is to construct a signature graph over the datapoints using the distance in the slr domain. In order to transform the datapoints to the slr domain, we used four different bases and offsets , respectively, either based on perfect knowledge or on estimations with 1% ( ), 2% ( ) and 3% ( ) known labels.

For each SNR, 1000 runs were performed with subsequent averaging over the misclassification rate. As the cluster labels might be permuted w.r.t. the ground truth, the assessment of the misclassification rate was based on the Hungarian algorithm [29].

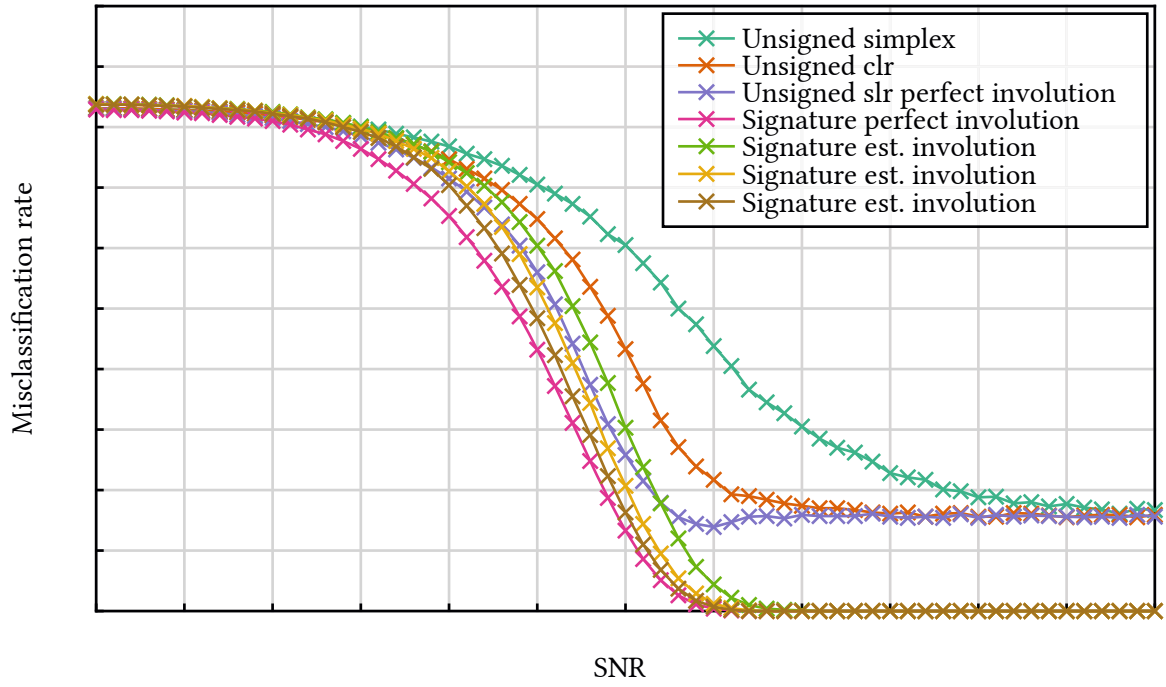


Figure 7: Misclassification rate of different methods based on spectral clustering over SNR.

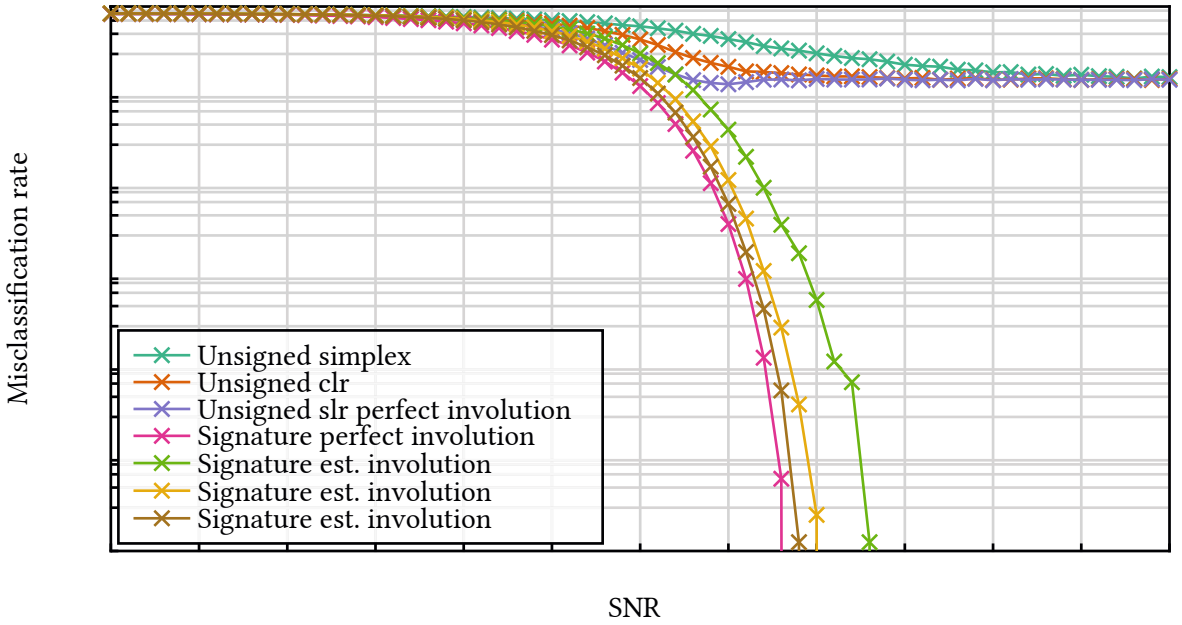


Figure 8: Misclassification rate of different methods based on spectral clustering over SNR, logarithmic representation.

Figure 7 and Figure 8 compare the misclassification rates for the methods described above: Zero misclassification is impossible to achieve using unsigned graphs, even in the high SNR regime, as we always hit a bottom at around 18%. However, the performance in the medium SNR regime gets better, moving from simplex to clr and finally to slr (with perfect knowledge about the involution).

Signature graphs, on the other hand, achieve flawless classification in the high SNR regime. Depending on the number of preallocated labels for estimating the involution, the method performs differently in the transient area, converging to the lower bound, which uses perfect knowledge about the involution.

4.5 Multiple Samples

Up to now we have assumed to have only one datapoint per node. However, we might face the situation that multiple datapoints are assigned to one node, for instance multiple measurements of one sensor at different points in time, that can be modelled as sampled random variables. Asserting the distance between them, in order to learn the graph, must therefore incorporate an expectation. We straightforwardly extend our previous notion of distance directly in the simplified slr domain:

When working with the sampled data, i.e., multiple realisations of and , we have to replace the expectation and covariance operations in (152) by their empirical counterparts. The th entry of the sign vector is now dependent on the sign of the covariance between the th components in the random vectors instead of relying on the sign of their product. In the case of a single sample, however, the empirical covariance falls back to a simple product, thus being compatible to our previous approach.

It is noteworthy that a graph learned based on this extended framework is not guaranteed to be balanced any longer. This stems from the fact, that three scalar random variables can have pairwise correlations that are negative and thus potentially form a cycle with three negative edges as the following example shows.

Let be a zero-mean random vector with two entries and identity covariance matrix,

Consider now the transformed random vector with

The correlation of this vector then equals

As all off-diagonal elements are negative, every pairwise correlation of the entries in $\bar{}$ is negative. Looking at three generic scalars, on the other hand, it is impossible that every pair of them has different signs, governing balancedness if they form a cycle.

5 Interpolation

In this chapter we will look at common interpolation strategies in the context of signature graphs. We will start out with an unsigned, unweighted graph, which will then be altered to become a signature graph, based on observed data. In the second part of this chapter we will discuss how to interpolate partly observed signals on this graph and discover computational advantages that can be exploited in the case of balanced graphs. The chapter is closed with a numerical study that indicates the functionality of the provided methods.

5.1 Learning Edge Weights

Let us face the following scenario: We observe a compositional-valued graph signal, for instance from a sensor network that measures the compounds of a gas at different locations. Based on our knowledge about the spatial distribution of the sensors we build a graph that expresses our expectation of pairwise correlations by unweighted edges. For instance, we might build the graph using -NN construction as we expect that sensors in close proximity to each other exhibit some sort of correlation. We neither specify how they are correlated nor how strong of a correlation we expect. As we have seen in Section 3.5, we can proceed to identify the basis, which separates our signal into mutually uncorrelated components and constitutes the involution.

Once we have performed this identification based on observed data, we aim to alter the graph in two ways: As the graph structure was just derived from an educated guess about the signals behaviour, it might be beneficial to now also incorporate the empirical knowledge from our observed data in it. This can be done by removing edges between nodes for which we have estimated a low correlation. Furthermore, we want to add signature weights to the edges that we keep.

Let us denote the slr basis identified according to Section 3.5 by and the observed graph signals used for this identification task by

The data transformed to the slr domain is denoted by

Our initial graph shall be called . We now define the weight function,

based on the empirical correlations,

Note that we can also directly pull these empirical correlations from intermediate results of the identification task, i.e., without explicitly calculating . They are given by the diagonal elements of the matrix in the last iteration of the joint diagonalisation (see (130)),

In the next step we remove edges, which have a low weight attached. For the subsequent interpolation low-weight edges do not play a significant role and could therefore be kept, however, it might be beneficial from a computational perspective to make the Laplacian more sparse without losing essential information. We can describe this step by introducing a modified weight function,

The threshold has to be chosen carefully. One way to asses a suitable value is to plot a histogram of the weight magnitudes and look for a phase transition in it.

5.2 Balanced Signature Graphs

As we will see in Section 5.3, balanced signature graphs provide computational advantages over unbalanced signature graphs when we apply common interpolation strategies. Although balancedness seems like a rather strict constraint, it is actually often encountered in real world data, e.g., in social networks [30]. If we are provided with an unbalanced graph (for instance by construction as described in the previous section) we might raise the question if it is at least approximately balanced and, if so, how we can balance it.

5.2.1 Assessing Unbalancedness Quantitatively

Under the assumption that our layer decorrelation efforts in the involutions identification were successful, it is reasonable to look at the graph layers independently, each constituting an ordinary signed graph. Recall that we call a signature graph balanced if and only if all layers are individually balanced. Thus we can asses the potential unbalancedness of our signature graph by looking at the unbalancedness of every layer.

Two measures were proposed in the literature for characterising unbalancedness of a signed graph: The straightforward measure – at least by definition – is the frustration index . It is the minimum number of edge changes we have to perform to convert our unbalanced graph into a balanced one. Although being an easy to grasp definition, calculation of turns out to be NP-hard. This is due to the fact that it is related to the Max-Cut problem, which is known to be NP-hard in itself [31].

Another measure, which is computationally feasible, but not as simple to interpret is the so-called algebraic conflict. It is the smallest eigenvalue of the Laplacian . The reasoning behind this is that a balanced graph has exactly zero as smallest eigenvalue [32] and a near-zero value, therefore, indicates approximate balancedness. Indeed, according to [33, ch. 4], adding a normalisation relates the algebraic conflict approximatively to the frustration index by

—

If this approximative frustration index is significantly lower than the number of edges on all layers of the graph we can assume approximate balancedness.

5.2.2 Balancing the Graph

The problem of balancing a signed graph is closely related to computing the frustration index: We look for the smallest set of edges we need to alter to arrive at a signed graph. In fact, we can reduce the problem of calculating the frustration index to the graph balancing problem, as the frustration index is a by-product of the balancing problem, simply as the sum of all changed edges. Therefore, it is NP-hard as well.

Commonly, edge signs are flipped in order to arrive at a balanced graph, however this seems wrong in the context of interpolation, as our graph then contains edges that do not conform with our prior observations and thus do not provide a good model on which to base the interpolation. It is much more favourable to discard edges rather than altering them. Ultimately, the underlying problem does not change as edge removal and sign flip both resolve the same conflict in the signed graph.

As exact evaluation of the problem is already infeasible for signed graphs, we can not expect to find a feasible solution in the realm of signature graphs. Note that the balancing problem on signature graphs is not decomposable into layer-wise signed graph balancing problems. This is due to the fact that changing the same edge at two different layers should not be penalised as two edge changes but rather as one. This couples the problems inextricably.

Multiple balancing algorithms on unweighted signed graphs were proposed by Diao [31]. We will adapt his local search algorithm to weighted signature graphs, but let us first look into the original formulation by Diao et al. as a starting point: The algorithm begins by setting up two vertex sets, and . The goal is now to move vertices between sets such that the number of negative edges between sets is as high as possible, while the edges between vertices in the same set are preferably positive. The algorithm does this by iterating repeatedly over the vertices and checks whether it might be favourable to move the vertex from one set to the other. If so, the vertex is moved. Clearly, this might only converge to a local solution of the underlying optimisation problem. At the end we either flip the edge signs that do not agree with the vertex partition or discard them as proposed previously.

The algorithm can be equivalently reformulated in terms of vertex labels, i.e., we store the vertices affiliation with one of the sets as a label local to the node. We assign the label to the th node to mark a vertex in and for one in . This allows us to formulate the local optimisation in terms of labels,

This can be understood as a voting procedure in which all nodes adjacent to node vote for a their label if and for the opposite label if . The extension to weighted graphs is now achieved straight forwardly, by admitting real values as weights. The voting analogy remains intact, but edges with larger weight have more impact.

Moving to signature graphs with layers, we have to maintain sets or equivalently labels of dimension . The local optimisation now reads

with being the indicator function and denoting the Hadamard product. and shall be understood according to the convention introduced in Section 2.2. Unfortunately, this minimisation does not decompose into layer-wise minimisations, which is no surprise as we have argued above already. For small , we can approach it in a brute-force manner, but for larger the minimisation becomes infeasible to the authors best knowledge.

The rest of the algorithm is adapted accordingly: Edges that do not comply with the converged solution are removed, yielding the new edge set of the balanced graph,

5.3 Interpolation on Balanced Signature Graphs

The two interpolation methods described in Section 2.2.5 will next be examined in the context of signature graphs. The general idea is that we perform the interpolation layer by layer as the basis identification has already untangled the inter-layer correlations.

5.3.1 Bandlimited Reconstruction

The bandlimited reconstruction assumes that only a limited number of graph Fourier basis vectors are excited. Lets recall (47),

Remember that is the matrix that formally dismisses all unobserved values of the graph signal and is containing only the first graph Fourier basis vectors. The interpolation comes down to calculating a matrix vector product. is the same for every layer, as we either observe the full signal at a node, or none, but generally differs across layers. In the special case of balanced graphs, however, it can be show that , and therefore , is indeed equal across the layers up to a row-wise sign flip [8]. Therefore we can always express the graph Fourier basis of a balanced graph by a product of the graph Fourier basis belonging to the unsigned counterpart and a diagonal sign flip matrix ,

Inserting this relation into (166), yields

As and both are diagonal and square by construction, they commute. Furthermore, we note that and introduce such that

is the sign flip matrix reduced to observed nodes and can be formally calculated from and ,

We can now write the reconstruction as

(171) shows that the interpolation of a signed graph can be performed by flipping the the signs of the signal values according to , interpolating over the unsigned graph and finally flipping the signs of the interpolated signal again.

The computational gains are obvious: We only need to calculate the Laplacians eigenvectors once for the unsigned graph, in contrast to layer-wise. Furthermore, the pseudo-inverse has to be calculated only once. Additionally, we need the matrix , but its calculation is not very expensive: One way would be to acquire the diagonal elements of from the sign pattern of the first eigenvector. Another option is to use the labels that were assigned by the balancing algorithm presented in the previous section.

5.3.2 Laplacian Reconstruction

In the same spirit we want to reduce the problem of Laplacian reconstruction on balanced signature graphs. Let us recall for that purpose the reconstructor from (52):

—

Naively, we have to calculate the term called interpolator for every layer individually. Once again the idea is to exploit the fact that the eigenvectors only switch signs when switching to the unsigned version of the graph, while the eigenvalues do not change at all. As we can decompose the Laplacian into

we can also use the sign switching matrix defined in the previous section to state the relationship between the balanced signed and the unsigned graph (now takes the role of),

Thus, we can write the interpolation as

—

Once again arguing that the expression is diagonal and that self-inverse as well as diagonal, we can pull out,

We encounter the same structure as in the bandlimited case with all the mentioned benefits: The interpolation on the signed graph comes down to an interpolation on the unsigned graph with anterior and posterior sign switches in the respective graph signals.

5.4 Numerical Experiments

In this section, we evaluate the performance of the blind involution identification independently, as well as the bandlimited interpolation method presented in Section 5.3. The creation of the synthetic graph signals used is described next.

5.4.1 Data Synthesis

As we want to apply a method from Section 5.3, we will work on balanced graphs only. The starting point of the signals synthesis is an unsigned unweighted graph. In our case, we used a graph representing the municipalities of Austria (in prospect of future work on demographic/voting data). Each vertex represents one municipality. The edges were constructed by -NN construction with and using the spatial distance between municipality centres as metric. The graph has 2118 vertices, 12216 edges and is connected.

Since we want to interpolate on the graph, signals on it must exhibit some sort of smoothness. To generate them, we first calculate the first eigenvectors of the graph Laplacian, , which will serve as the signal's basis, that ensures low-pass behaviour. The first stage of the signal generation is to draw a random matrix with entries distributed standard normally and independent. We then calculate the matrix product,

By this construction, the values across columns of are uncorrelated, but the columns in itself feature a correlation induced by the graph structure. is now a graph signal, with each row belonging to one node of the graph.

Connected nodes tend to exhibit positive correlations, but the signals we are interested in shall also show some strong negative correlations between adjacent nodes, that we can characterise with signature graphs conveniently. The idea is now to flip the sign of some signal values. If only one of two adjacent nodes gets flipped, the positive correlation along the connecting edge becomes negative. Furthermore, this enforces balancedness by construction. We will describe the sign flip operation as a Hadamard product,

The matrix is constructed randomly, by starting out with an all ones matrix and flipping every entry with a probability of , but it remains constant across all observations, as it is part of the graphs description. One may think of the sign flips as an introduction of deterministic high frequency components into the signal.

Still is uncorrelated across layers/columns, i.e., our interpolation problem might still be solved by looking at individual layers on the graph and perform interpolation tasks on each layer individually. The logical next step is to postmultiply by a randomly chosen but constant basis
to get

or in compositional terms, to transform the data to the simplex with basis ,

Recall from Section 3.5.1 that and are related by an ilr basis ,

Having established a framework to generate the graph signals or with the desired properties of inter-layer correlations based on and intra-layer correlations based on the graphs structure we may now direct our attention to a performance evaluation of the blind basis identification.

5.4.2 Identification Performance

To evaluate the performance of the basis identification, we want to calculate the error between the estimated basis and the ground truth that we have used for synthesis. However, a mere subtraction of the two matrices with subsequent calculation of the Frobenius norm is not viable, as the estimated basis is ambiguous with respect to a permutation, scaling and column-wise sign flips (cf. Section 3.5.2).

To deal with the scaling aspect, we start out by normalising and ,

The error can then be defined as

with the unknown permutation and sign-flip matrix . To guess the correct we make use of the following ad hoc method. We postmultiply the content of the norm, which we expect to be close to , by the inverse of ,

As , we can reformulate this identity as

The right-hand side therefore gives an approximation to the transposed permutation matrix. To convert to a valid permutation matrix we look for the position of the entry in with the largest magnitude and decide that our permutation matrix is at this position, depending on the sign of that entry in . Then we proceed to do the same for the largest value in excluding the row and column of the previously selected position. We keep iterating this step, always excluding all previously selected rows and columns from the search set until we exclude the whole matrix. Then we are left with a valid signed permutation matrix . We do not give a guarantee that this solution is optimal in the sense

–

when encountering large estimation errors.

For the sign flip probability we have chosen . This results in a probability of 32% that any given edge is negative, as the underlying probability that the nodes connected by an edge have different signs can be calculated by

For every number of observations we have run the estimation over 1000 different randomly generated signals using the first 10% of eigenvectors. We then have calculated the sample mean and median of the errors ¯ . They are depicted in Figure 9 on a log-log scale.

As expected, we observe a decrease in the error with increasing observation data. In the case of the median error the decrease is linear in the log-log representation. The mean error also initially decreases linearly, but hits a plateau at 10 observations and 2% relative error energy. This behaviour suggests that we encounter an error distribution with large outliers that shift the mean up but do not affect the median performance.

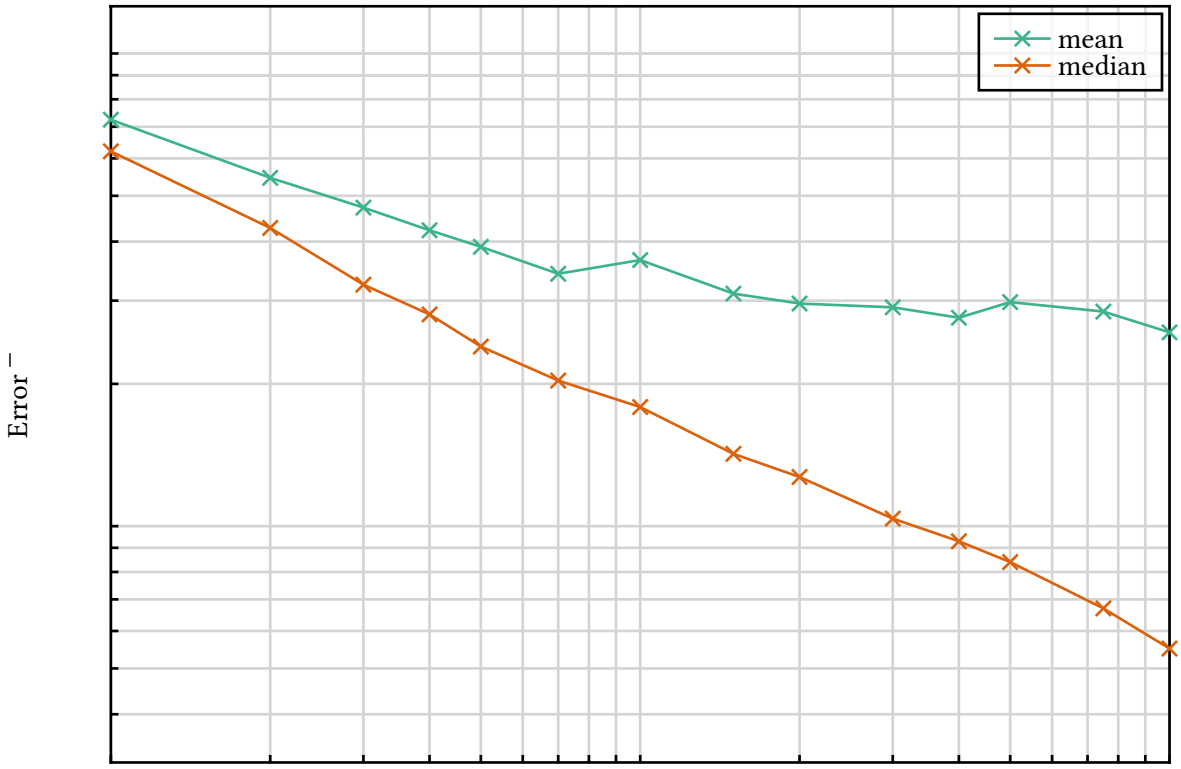


Figure 9: Mean and median estimation errors in .

5.4.3 Interpolation Performance

Let us now turn our attention to the actual interpolation. We keep using the signal synthesis described in Section 5.4.1, but decrease the graph's size to reduce the computational burden. This is done by selecting the subgraph that describes Upper Austria exclusively, leaving us with 438 vertices and 2360 edges.

As we know that the synthesised signal is bandlimited, we will focus on bandlimited reconstruction. We are using 10% of the eigenvectors for both, synthesis and reconstruction. Furthermore, we assume perfect knowledge about the graph instead of learning it from the observed data, in order to assess the interpolation performance individually. The involution, on the other hand, was estimated from full observations.

The actual reconstruction is performed on signals of which we have discarded half of the signal values by random selection. The reconstruction error is calculated as error norm between the actual signal and its reconstruction ,

A normalisation to the signal energy is imperative for a fair comparison,

The results are displayed in Figure 10 by means of three sample statistics – mean, median and minimum.

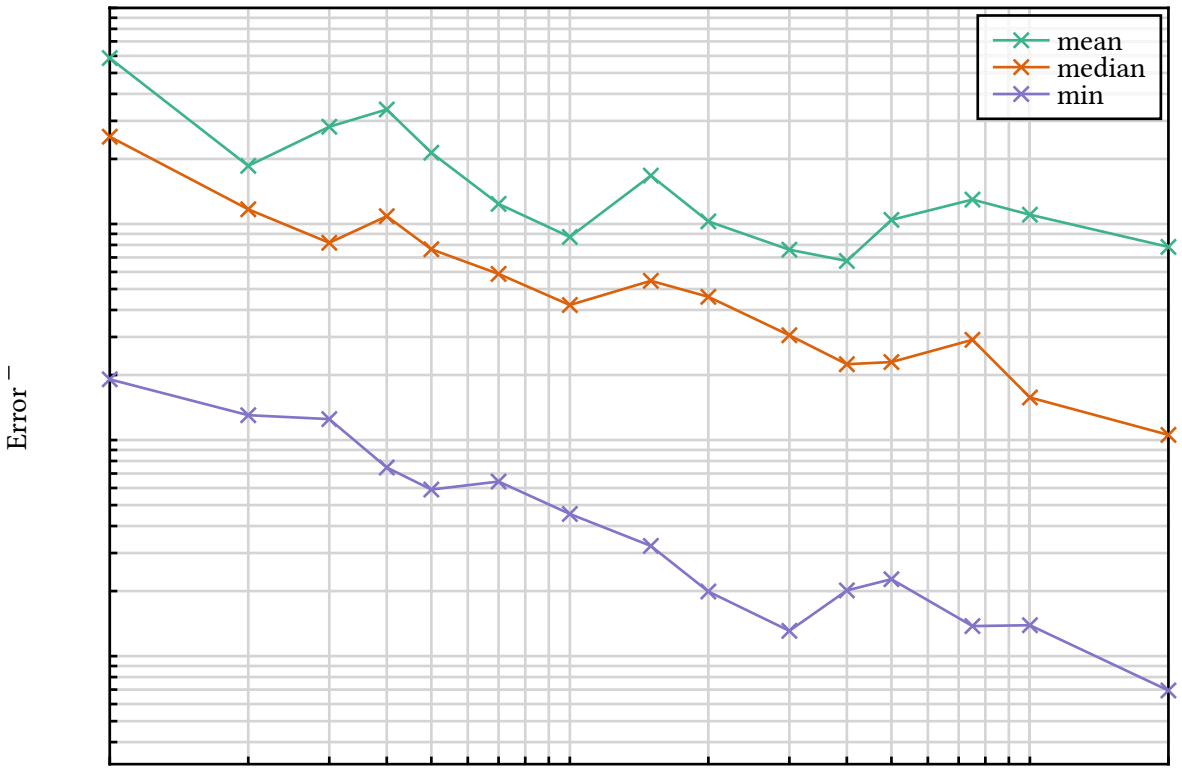


Figure 10: Mean and median and minimum reconstruction errors.

It is once again obvious that there is a large spread between median and mean performance. Therefore, we want to examine the underlying reasons for the occurrence of large interpolation errors. To do so, we have plotted the interpolation error $^{-}$ over the involution estimation error $^{-}$ in Figure 11. And indeed we find a fairly strong correlation between these errors. Thus, we conclude that the interpolation is quite sensitive to errors in the basis of the involution and a good estimate of the involution is of utmost importance for a successful reconstruction.

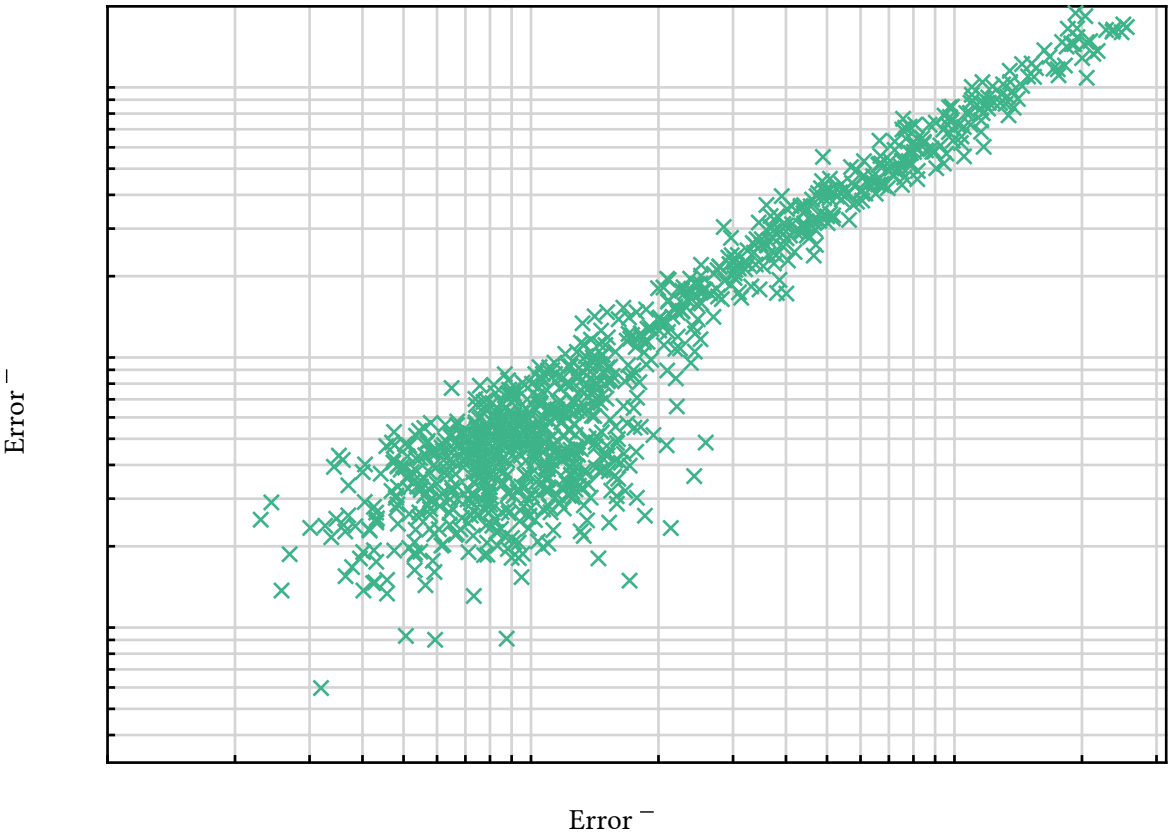


Figure 11: Bandlimited reconstruction error over involution estimation error.

6 Conclusion and Outlook

This chapter's aim is to recall our findings in the three areas, involutions in compositional data, classification, and interpolation. We also want to point towards open questions that could not be answered in the context of this work and might provide starting points for future research about signature graphs.

6.1 Involutions

We have identified the class of linear and affine involutions in compositional data. Furthermore, we have introduced the slr transform as a tool for interpreting these involutions as well as a preparatory stage for subsequent graph learning. A data model adapted to the necessities of compositional data was stated and used as a foundation for the identification of the involution's basis by means of labelled data. We then took another perspective on the identification task were we assumed to know which datapoints exhibit a significant pairwise correlation, e.g., from knowing the topological structure of an underlying graph. Methods from blind source estimation were transferred towards application over the graph domain as a means to identify a linear involution. Both methods were proven to work, however the blind estimation exhibits a large gap between mean and median performance. It is yet to be investigated what causes the seldom but large increase in error and how it can be eventually prevented – especially since we have shown that minimising the error is crucial for good interpolation results.

6.2 Classification

With regard to classification, we have shown a method to learn a signature graph from compositional data given the involution. Learning in the slr domain, as opposed to the simplex domain, was shown to be computationally advantageous as the combinatorial minimisation occurring in the distance calculation can be solved analytically in the slr domain. The proposed learning method was proven to produce balanced graphs, making perfect layer-wise clustering possible. In a numerical study we have shown the advantage of signature graphs for clustering synthetic data over simple unsigned graphs.

Using ordinary graphs for clustering relies on the concept of sparse embedding: The -dimensional relation between vector valued datapoints is reduced to a one-dimensional distance. With -layered signature graphs on the other hand, we express the relationship between distinct features of the datapoints. In this work, we have chosen the extreme case of , i.e., we did not perform a reduction in dimension at all, which somehow contradicts the original idea of a sparse embedding for graph clustering. Also, the requirement on the symmetries in the data may be to strict for many real world datasets.

Therefore, we think that the middle ground of using signature graphs with is a promising approach which shall be looked into in the future and that for two reasons: It better catches the spirit of sparse embedding and it is less demanding on the symmetries in the data. In fact, the symmetries must only be present in the sparse embedding of the data. How to design such an embedding is yet to be explored in depth.

6.3 Interpolation

Our main contribution to the untouched field of interpolation on signature graphs was the transfer of common reconstruction methods on ordinary graphs – namely Laplacian and bandlimited reconstruction – to signature graphs. Significant simplifications in the calculation could be achieved under the condition of balancedness. This motivated the statement of an approximate criterion whether a signature graph is nearly balanced. We also transferred an algorithm to balance signed graphs for its use in the realm of signature graphs. A numerical study of the bandlimited reconstruction was conducted and has uncovered a strong dependency between the quality of the involution's estimate and the performance of the bandlimited interpolation. How to decrease the interpolations sensitivity to errors in the involution's description is yet to be investigated.

Bibliography

[1] M. Castells, "Front Matter", in *The Rise of the Network Society*, John Wiley & Sons, Ltd, 2009, p. i-lvii. doi: https://doi.org/10.1002/9781444319514.fmatter.

[2] "The world's most valuable resource; Regulating the data economy", *The Economist*, vol. 423, May 06, 2017. Accessed: May 02, 2024. [Online]. Available: https://www.proquest.com/magazines/worlds-most-valuable-resource-regulating-data/docview/1895941741/se-2

[3] "Volumen der jährlich generierten/replizierten digitalen Datenmenge weltweit von 2010 bis 2022 und Prognose bis 2027 (in Zettabyte)". Statista, May 18, 2023. Accessed: May 02, 2024. [Online]. Available: https://de.statista.com/statistik/daten/studie/267974/umfrage/prognose-zum-weltweit-generierten-datenvolumen/

[4] A. Ortega, *Introduction to Graph Signal Processing*. New York (NY): Cambridge University Press, 2022. doi: 10.1017/9781108552349.

[5] A. Ortega, P. Frossard, J. Kovačević, J. M. F. Moura, and P. Vandergheynst, "Graph Signal Processing: Overview, Challenges, and Applications", *Proc. IEEE*, vol. 106, no. 5, pp. 808–828, May 2018, doi: 10.1109/JPROC.2018.2820126.

[6] A. Brandstädt, V. B. Le, and J. P. Spinrad, "1. Basic Concepts", in *Graph Classes: A Survey*, Philadelphia (PA): Society for Industrial and Applied Mathematics, 1999, pp. 1–19. doi: 10.1137/1.9780898719796.ch1.

[7] M. Kivelä, A. Arenas, M. Barthelemy, J. P. Gleeson, Y. Moreno, and M. A. Porter, "Multilayer networks", *Journal of Complex Networks*, vol. 2, no. 3, pp. 203–271, 2014, doi: 10.1093/comnet/cnu016.

[8] G. Matz and T. Dittrich, "Signature Graphs — Fundamentals, Learning, and Clustering", in *Proc. Asilomar Conf. Signals, Systems, and Computers*, Pacific Grove (CA), Nov. 2022, pp. 235–239. doi: 10.1109/IEEECONF56349.2022.10052105.

[9] G. Matz, "On Generalized Signature Graphs", in *Proc. IEEE Int. Conf. Acoustics, Speech and Signal Processing (ICASSP)*, Seoul (South Korea), Apr. 2024, pp. 13336–13340. doi: 10.1109/ICASSP48485.2024.10446569.

[10] J. Aitchison, "The Statistical Analysis of Compositional Data", *J. Royal Stat. Soc.: Series B (Methodological)*, vol. 44, no. 2, Jan. 1982, doi: 10.1111/j.2517-6161.1982.tb01195.x.

[11] V. Pawlowsky-Glahn, J. J. Egozcue, and R. Tolosana-Delgado, *Lecture Notes on Compositional Data Analysis*. Girona, Barcelona, 2011.

[12] J. J. Egozcue, V. Pawlowsky-Glahn, G. Mateu-Figueras, and C. Barceló-Vidal, "Isometric Logratio Transformations for Compositional Data Analysis", *Mathematical Geology*, vol. 35, Apr. 2003, doi: 10.1023/A:1023818214614.

[13] T. Dittrich and G. Matz, "Signal Processing on Signed Graphs: Fundamentals and Potentials", *IEEE Signal Proc. Mag.*, vol. 37, no. 6, pp. 86–98, Nov. 2020, doi: 10.1109/MSP.2020.3014060.

[14] F. Harary, "On the notion of balance of a signed graph", *Michigan Math. J.*, vol. 2, pp. 143–146, 1953, doi: 10.1307/MMJ/1028989917.

[15] Z. Hammoud and F. Kramer, "Multilayer networks: aspects, implementations, and application in biomedicine", *Big Data Analytics*, vol. 5, no. 2, 2020, doi: 10.1186/s41044-020-00046-0.

[16] G. Matz, C. Verardo, and T. Dittrich, "Efficient Learning of Balanced Signature Graphs", in *Proc. IEEE Int. Conf. Acoustics, Speech and Signal Processing (ICASSP)*, Rhodes Island (Greece), Jun. 2023. doi: 10.1109/ICASSP49357.2023.10095989.

[17] L. Qiao, L. Zhang, S. Chen, and D. Shen, "Data-driven graph construction and graph learning: A review", *Neurocomputing*, vol. 312, pp. 336–351, 2018, doi: 10.1016/j.neucom.2018.05.084.

[18] W. Dong, C. Moses, and K. Li, "Efficient k-nearest neighbor graph construction for generic similarity measures", in *Proc. ACM Int. Conf. World Wide Web (WWW '11)*, New York (NY), 2011, pp. 577–586. doi: 10.1145/1963405.1963487.

[19] S. E. Schaeffer, "Graph clustering", *Comp. Science Rev.*, vol. 1, no. 1, pp. 27–64, 2007, doi: 10.1016/j.cosrev.2007.05.001.

[20] D. R. Karger, "Minimum cuts in near-linear time", *J. of the ACM*, vol. 47, no. 1, pp. 46–76, Jan. 2000, doi: 10.1145/331605.331608.

[21] J. A. Davis, "Clustering and Structural Balance in Graphs", *Human Relations*, vol. 20, no. 2, pp. 181–187, 1967, doi: 10.1177/001872676702000206.

[22] R. Mondal, E. Ignatova, D. Walke, D. Broneske, G. Saake, and R. Heyer, "Clustering graph data: the roadmap to spectral techniques", *Discover Artificial Intelligence*, vol. 4, no. 1, Jan. 2024, doi: 10.1007/s44163-024-00102-x.

[23] S. K. Narang, A. Gadde, and A. Ortega, "Signal processing techniques for interpolation in graph structured data", in *Proc. IEEE Int. Conf. Acoustics, Speech and Signal Processing*, Vancouver (BC), 2013, pp. 5445–5449. doi: 10.1109/ICASSP.2013.6638704.

[24] S. Chen, R. Varma, A. Sandryhaila, and J. Kovačević, "Discrete Signal Processing on Graphs: Sampling Theory", *IEEE Transactions on Signal Processing*, vol. 63, no. 24, pp. 6510–6523, Aug. 2015, doi: 10.1109/TSP.2015.2469645.

[25] D. S. Bernstein, *Matrix Mathematics: Theory, Facts, and Formulas*, 2nd ed. Princeton University Press, 2009.

[26] G. H. Golub and C. F. Van Loan, "An Analysis of the Total Least Squares Problem", *SIAM J. Numerical Analysis*, vol. 17, no. 6, pp. 883–893, 1980, doi: 10.1137/0717073.

[27] C. Jutten and P. Comon, "Handbook of Blind Source Separation". Academic Press, Oxford (England), 2010. doi: 10.1016/B978-0-12-374726-6.00006-0.

[28] G. H. Golub and C. F. Van Loan, *Matrix Computations*. Johns Hopkins University Press, 1996. [Online]. Available: https://books.google.at/books?id=mlOa7wPX6OYC

[29] H. W. Kuhn, "The Hungarian method for the assignment problem", *Naval Research Logistics*, vol. 2, no. 1–2, pp. 83–97, Mar. 1955, doi: 10.1002/nav.3800020109.

[30] F. Heider, "Attitudes and Cognitive Organization", *The J. of Psychology*, vol. 21, no. 1, pp. 107–112, 1946, doi: 10.1080/00223980.1946.9917275.

[31] Z. Diao and Z. Tang, "Approximation Algorithms for Balancing Signed Graphs", in *Algorithmic Aspects in Information and Management*, Jinhua (CN): Springer International Publishing, Aug. 2020, pp. 399–410. doi: 10.1007/978-3-030-57602-8_36.

[32] Y. P. Hou, "Bounds for the Least Laplacian Eigenvalue of a Signed Graph", *Acta Mathematica Sinica*, vol. 21, no. 4, pp. 955–960, Aug. 2005, doi: 10.1007/s10114-004-0437-9.

[33] J. Kunegis, "Exploiting The Structure of Bipartite Graphs for Algebraic and Spectral Graph Theory Applications", *Internet Mathematics*, vol. 11, no. 3, May 2015, doi: 10.1080/15427951.2014.958250.

Statement of Academic Integrity

I hereby declare and confirm with my signature that the thesis is exclusively the result of my own autonomous work except where states otherwise by reference of literature. I also declare that no part submitted has been made in an inappropriate way, whether by plagiarising or infringing on any third person's copyright. Finally, I declare that no part submitted has been plagiarised for any other paper in another higher education institution, research institution or educational institution.

Vienna, 7 May 2024

Dimitrios Kalodikis