New Algorithms and Methodology for Analysing Distances

George Kettleborough

Supervisors:
Dr K.T. Huber
Prof. V.J. Rayward-Smith
Dr B. De La Iglesia

Doctor of Philosophy

University of East Anglia

School of Computing Sciences

June, 2014

Abstract

DISTANCES ARISE in a wide variety of different contexts, one of which is partitional clustering, that is, the problem of finding groups of similar objects within a set of objects. These groups are seemingly very easy to find for humans, but very difficult to find for machines as there are two major difficulties to be overcome: the first defining an objective criterion for the vague notion of "groups of similar objects", and the second is the computational complexity of finding such groups given a criterion. In the first part of this thesis, we focus on the first difficulty and show that even seemingly similar optimisation criteria used for partitional clustering can produce vastly different results. In the process of showing this we develop a new metric for comparing clustering solutions called the assignment metric. We then prove some new NP-completeness results for problems using two related "sum-of-squares" clustering criteria.

Closely related to partitional clustering is the problem of hierarchical clustering. We extend and formalise this problem to the problem of constructing rooted edge-weighted *X-trees*, that is trees with a leafset X. It is well known that an X-tree can be uniquely reconstructed from a distance on X if the distance is an ultrametric. But in practice the complete distance on X may not always be available. In the second part of this thesis we look at some of the circumstances under which a tree can be uniquely reconstructed from incomplete distance information. We use a concept called a *lasso* and give some theoretical properties of a special type of lasso. We then develop an algorithm which can construct a tree together with a lasso from partial distance information and show how this can be applied to various incomplete datasets.

Contents

List of Figures

List of Algorithms

List of Tables

Chapter 1

Introduction

Metric spaces are a generalisation of the world in which we live where physical objects appear to exist in 3-dimensional space and we have a notion of the distance between pairs of objects. The distance that many people think of is the Euclidean distance, the length of the straight line between the objects that one would measure with a ruler. However, taxi drivers are more familiar with the Manhattan distance, the distance one must travel between locations by traversing the grid-like streets of a city, and airline pilots the great circle distance, the shortest distance between two points on a sphere. There are many ways to calculate a distance but those most familiar to humans all share a few key properties, namely they are metrics. We therefore live in a metric space, or at least a close approximation of one.

Increasingly, our world is becoming filled with other, more abstract types of objects that "live" outside of the physical space: data. Regardless of the form that data takes, be it numbers, text or pictures, the presence of data always comes with the need for a method of comparison. Since we are most at ease with thinking about our own world, it seems natural that we should imagine data points as "living" in a space with a distance defined which holds the same key properties as the distances we use every day. Thus, the data points become members of their own metric space.

There are a number of problems that present themselves when we have a metric space. The problem of identifying groups of similar objects based on their relative proximity is called clustering. The problem is very old, ubiquitous and has a rich and diverse set of applications. Despite the fact that intelligent beings seem to be naturally adept at the task, it is a well-known hard problem for a computer, and in more than one sense. The first difficulty is in defining precisely what one actually expects in terms of a metric, rather than the vague objective of "groups of similar objects". The second difficulty is in the sense of computational complexity. In the first part of this thesis we focus mainly on the first difficulty in the context of partitional clustering, which seeks to partition a set into a given number of subsets. Many objective criteria have been developed to measure the usefulness of a partition as a solution to a given clustering problem, but we show that even seemingly similar optimisation criteria can produce vastly different results (see Section 3.5).

In order to show that criteria can produce vastly different partitions it is necessary to be able to compare partitions. Some metrics have been devised for that purpose already but they are all naïve with respect to the fact that our data itself lives in a metric space. We find ourselves with several layers of metric spaces and we introduce the concept of "lifting" a metric space to the space of its power set. In this way we introduce a new metric for comparing partitions that can take into account the fact that the data lives in a metric space (see Section 3.4.2).

Metrics present themselves in the context of that most well-known and useful data structure: the tree. In a rooted edge-weighted X-tree, that is a tree with leaf set X, the graph-theoretic distance between the leaves is a metric called a tree metric. It is well known that a tree is uniquely defined by and can be reconstructed from the tree metric induced on its leafset. Tree construction is appropriate and desirable in many areas of classification such as in evolutionary biology. For example, if we had a dataset corresponding to varieties of some organism, we could find a partitional clustering on them in which each cluster corresponds to a continent, reflecting the expected result that organisms on the same continent are more closely related, and those on different continents are more distantly related. However, a tree can show us not only this information but the relationships and lineages of each variety in our set. In this way, trees can be considered hierarchical clusterings and a step up from partitional clusterings.

In practice, though, we may not always have access to the complete distance information for a set of elements. Rather we might be presented with a partial distance, that is where the distance between some pairs elements is unknown. Studying the properties of partial distances and their ability to uniquely identify a tree has given rise to the theory of "lassoing" a tree. It turns out that certain subsets of all pairwise distances do indeed contain enough information to uniquely identify either the topology of a tree, its edge weights, or both. This theory is reviewed in Section 4.4.

We therefore investigate the problem of constructing an edge-weighted tree from partial distance information. We begin by investigating the properties of a special type of lasso and then we present an algorithm for constructing the unique tree which corresponds to such a lasso. We then turn to the more general problem of reconstruction from partial distance information and present an algorithm which constructs a tree from a partial distance and returns the set of given distances which uniquely determine the constructed tree.

This thesis is in two parts with the first part focusing on partitional clustering and the second on reconstruction of hierarchical clusterings (trees) from partial distance information. It is organised as follows: in Chapter 2, we introduce the concept of metric spaces and review various metrics that exist for various objects including partitions. This leads to a review of partitional clustering, the problem of finding partitions of a dataset. In Chapter 3, we focus on clustering and its difficulties by comparing and contrasting two closely related clustering criteria, called sum-of-squares criteria. In Chapter 4, we turn our attention to the tree reconstruction problem and introduce trees, tree metrics, reconstruction and lassos. In Chapter 5, we present an algorithm called Lasso for reconstruction from partial distances which is consistent according to the theory of lassos and show that it is applicable to construction of trees and supertrees from partial distance information.

Chapter 2

Partitional Clustering

2.1 Summary

In this chapter we first introduce the relevant terminology that is required for much of the work presented in this thesis including metrics and partitions. We then review the field of partitional clustering including the various criteria which have been developed, the computational complexity issues associated with the clustering problem and finally various methods which have been developed to perform partitional clustering.

2.2 Datasets and metric spaces

A dataset, in the most general sense, is simply a collection of objects. For now we will consider the collection to be a set, but later we will look at some generalisations.

Let D be a set. It is important for many applications to be able to compare objects in a set. For this purpose we can define a function:

$$d\colon D \times D \to \mathbb{R}^{\geq 0}.$$

Such a function is called a *dissimilarity on D*. Whenever we have, for any $x, y, z \in D$, that $d(x, y) > d(x, z)$, we interpret it as "x, y are more dissimilar than x, z".

Examples of dissimilarities include Bregman divergences, which are generalisations of Euclidean distance squared for objects in Euclidean space [10], and the Kullback-Leibler divergence for probability distributions [98].

Some dissimilarities are more useful than others. One would generally expect such a function to have the following two properties for all $x, y \in D$:

1. $d(x, y) = d(y, x)$ (symmetry),

2. $d(x, x) = 0$ (identical elements are most similar).

A dissimilarity on D that satisfies these two properties is called a *distance on D*. When we use a distance, objects are usually said to be close or distant instead of similar or dissimilar.

Examples of distances include the Bhattacharyya distance [14] and the *single-linkage* distance that we will see in Section 2.2.4.

Two further properties that one may expect a dissimilarity to satisfy are, for all $x, y, z \in D$:

3. $d(x, y) = 0$ only if $x = y$ (identity of indiscernibles),

4. $d(x, y) + d(y, z) \geq d(x, z)$ (triangle inequality).

A distance on D which satisfies all four properties is called a *metric on D*. If d is a metric on D then the pair (D, d) is called a *metric space*.

A metric is usually the most desirable and intuitive type of dissimilarity. In the remainder of this section we review some of the common metrics used for certain types of objects, namely Euclidean vectors, sequences, mixed vectors and sets.

Metrics exist for many other types of objects including probability distributions, for which metrics include the Hellinger distance and squared Jensen-Shannon divergence [50]. It is also possible to transform some dissimilarities into metrics, for example see Everitt [53, chap. 2.5]. In Section 2.3.2 we look at metrics (and other measures) for comparing partitions and in Chapter 3 we introduce our own metric.

2.2.1 Euclidean space metrics

Let M be m-dimensional Euclidean space, $\mathbb{R}^m$. For all $\mathbf{x} = (x_1, \ldots, x_m), \mathbf{y} = (y_1, \ldots, y_m) \in M$, metrics on M include the familiar *Euclidean distance* d_E defined as:

$$d_E(\mathbf{x}, \mathbf{y}) = \sqrt{\sum_{i=1}^{m} (x_i - y_i)^2},$$

the *Manhattan distance* d_M defined as:

$$d_M(\mathbf{x}, \mathbf{y}) = \sum_{i=1}^{m} |x_i - y_i|,$$

and the *Chebyshev distance* d_C defined as:

$$d_C(\mathbf{x}, \mathbf{y}) = \max_{1 \le i \le m} |x_i - y_i|.$$

It should be noted that all three are special cases of the *Minkowski distance* [41], d_I defined as:

$$d_I(\mathbf{x}, \mathbf{y}) = \left(\sum_{i=1}^{m} |x_i - y_i|^p \right)^{1/p}$$

for all $\mathbf{x} = (x_1, x_2, \ldots, x_m), \mathbf{y} = (y_1, y_2, \ldots, y_m) \in M$ where p is the order. For $p = 1$, $d_I = d_M$, for $p = 2$, $d_I = d_E$ and for $\lim_{p \to \infty}$, $d_I = d_C$.

The Mahalanobis distance [107] is a metric which is equivalent to Euclidean distance on scaled principle components of the dataset. To make this more precise, let $D = \{\mathbf{s}_1, \ldots, \mathbf{s}_n\} \subset \mathbb{R}^m$, then the *covariance* of the ith and jth components in D is defined as:

$$S_{ij} = \frac{1}{n-1} \sum_{k=1}^{n} (s_{ki} - \bar{s}_i)(s_{kj} - \bar{s}_j),$$

where s_{ki} means the ith component of the kth element and $\bar{s}_i$ means the mean of component i across D. In this context, components are often called *features* or *fields*. The covariance matrix associated with D is then the $m \times m$ matrix $S = [S_{ij}]$, which is clearly symmetric. The Mahalanobis distance, $d_P \colon D \times D \to \mathbb{R}^{\geq 0}$, is then defined as:

$$d_P(\mathbf{x}, \mathbf{y}) = \sqrt{(\mathbf{x} - \mathbf{y}) S^{-1} (\mathbf{x} - \mathbf{y})^T},$$

for all $\mathbf{x}, \mathbf{y} \in D$.

2.2.2 Sequence space metrics

Sequences are a frequently occurring type of object found in many areas such as evolutionary biology and the analysis of natural language. A well-known metric on sequences of equal length is the *Hamming distance* [72]. The Hamming distance, for two sequences x and y of equal length, is equal to the number of positions where x and y differ.

Another well-known metric, this time on sequences of arbitrary length, is the Levenshtein distance [155]. The Levenshtein distance is an edit distance, where the allowed edits are insertion, deletion and substitution (for definitions, see [103]). The Levenshtein distance between two sequences x and y is the minimum number of edits required to transform one sequence into the other.

2.2.3 Mixed data metrics

Euclidean space consists of m-tuples of real numbers, where m is some positive integer. This type of data is often called numerical data. Another type of data is categorical data, where the value of each object is one of a fixed, usually small, number of categories, such as true and false or male and female.

The simplest metric for categorical data is the *overlap metric*, which is also variously called the *split metric*, *discrete metric* or $0/1$ *metric*. For a set, D, the overlap metric, $d_O \colon D \times D \to \mathbb{R}^{\geq 0}$, is defined for all $x, y \in D$ as:

$$d_O(x, y) = \begin{cases} 0 & \text{if } x = y, \\ 1 & \text{otherwise.} \end{cases}$$

A frequently occurring type of data appearing in many areas is mixed data, which consists of m-tuples with both numerical and categorical components. Let D be a set of such m-tuples. The *heterogeneous Euclidean-overlap metric*, $d_H: D \times D \to \mathbb{R}^{\geq 0}$, uses normalised Euclidean distance on the numerical components and the overlap metric on categorical components. It is defined as:

$$d_H(\mathbf{x}, \mathbf{y}) = \sqrt{\sum_{i=1}^{m} d_{Hi}^2(x_i, y_i)},$$

for all $\mathbf{x} = (x_1, \ldots, x_m), \mathbf{y} = (y_1, \ldots, y_m) \in D$ where

$$d_{Hi}(x_i, y_i) = \begin{cases} d_O(x_i, y_i) & \text{if } i \text{ is a categorical component in } D, \\ \dfrac{|x_i - y_i|}{range_i} & \text{otherwise,} \end{cases}$$

for all $1 \geq i \geq m$ where $range_i$ is the difference between maximum and minimum observed values for component i in D. There also exist generalised Mahalanobis distances for mixed data, see for example [36].

2.2.4 Set metrics

Metrics are also possible on more complex structures, like sets. Throughout this thesis we use a convention for naming dissimilarities: d is used in general, δ is used for dissimilarities on sets and, later, we will use Δ for dissimilarities on partitions.

Symmetric difference

Let D be a set and 2^D its powerset. The cardinality of the symmetric difference, $\delta_\triangle: 2^D \times 2^D \to \mathbb{R}^{\geq 0}$ is a well-known metric on sets defined for all $X, Y \in 2^D$ as:

$$\delta_\triangle(X, Y) = |X \triangle Y|.$$

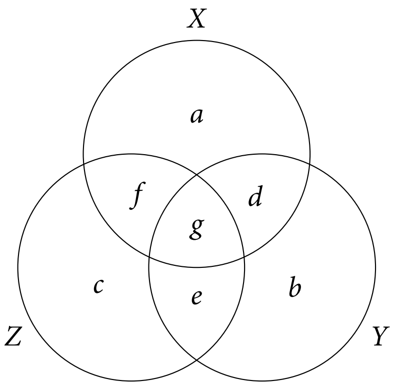


Figure 2.1: Partition of $X \cup Y \cup Z$. The quantities a, b, c, d, e, f, g represent the cardinalities of each disjoint set as shown.

A related dissimilarity is the normalised symmetric difference, $\delta_{\triangle}\colon 2^D \times 2^D \to \mathbb{R}^{\geq 0}$, defined as:

$$\delta_{\triangle_n}(X, Y) = \begin{cases} \dfrac{|X \triangle Y|}{|X \cup Y|} & \text{if } X \cup Y \neq \varnothing, \\ 0 & \text{otherwise,} \end{cases}$$

for all $X, Y \in 2^D$.

Theorem 1. *The normalised symmetric difference is a metric on* 2^D.

The following proof was presented in [169], with some errors. We reproduce the proof here with the errors corrected.

Proof. It is easy to see the function is nonnegative, symmetric and that $\delta_{\triangle_n}(X, Y) = 0$ if and only if $X = Y$ for all $X, Y \in 2^D$ so we will show that the triangle inequality holds which is:

$$\frac{|X \triangle Y|}{|X \cup Y|} + \frac{|Y \triangle Z|}{|Y \cup Z|} \geq \frac{|X \triangle Z|}{|X \cup Y|}$$

or, equivalently:

$$1 - \frac{|X \cap Y|}{|X \cup Y|} + 1 - \frac{|Y \cap Z|}{|Y \cup Z|} \geq 1 - \frac{|X \cap Z|}{|X \cup Z|}. \tag{2.1}$$

We partition $X \cup Y \cup Z$ into disjoint subsets as shown in Figure 2.1 and let

$$\begin{aligned} a &= |X \smallsetminus (Y \cup Z)|, & b &= |Y \smallsetminus (X \cup Z)|, \\ c &= |Z \smallsetminus (X \cup Y)|, & d &= |(X \cap Y) \smallsetminus Z|. \\ e &= |(Y \cap Z) \smallsetminus X|. & f &= |(Z \cap X) \smallsetminus Y|. \\ && g = |X \cap Y \cap Z|, \end{aligned}$$

and for convenience we let

$$\xi = |X \cup Y \cup Z|.$$

We can now write equation (2.1) as

$$1 - \frac{d+g}{\xi - c} + 1 - \frac{e+g}{\xi - a} \geq 1 - \frac{f+g}{\xi - b}$$

which can be rewritten as

$$\frac{d+g}{\xi - c} + \frac{e+g}{\xi - a} \leq \frac{f+g}{\xi - b} + 1.$$

Removing b from the denominators on the left-hand side can only make the left-hand side greater, so it is sufficient to show that

$$\frac{d+g}{\xi - b - c} + \frac{e+g}{\xi - a - b} \leq \frac{f+g}{\xi - b} + 1.$$

Now if we replace 1 with $\frac{\xi-a-b-c}{\xi-a-b-c}$ on the right-hand side and add the fractions on the left-hand side we get

$$\frac{(\xi - a - b)(d+g) + (\xi - b - c)(e+g)}{(\xi - b - c)(\xi - a - b)} \\ \leq \frac{(\xi - a - b - c)(\xi - b) + (\xi - a - b - c)(f+g)}{(\xi - b)(\xi - a - b - c)}$$

which when we expand the denominators becomes

$$\frac{(\xi - a - b)(d + g) + (\xi - b - c)(e + g)}{\xi^2 - \xi a - 2\xi b - \xi c + ab + bc + b^2 + ac} \leq \frac{(\xi - a - b - c)(\xi - b) + (\xi - a - b - c)(f + g)}{\xi^2 - \xi a - 2\xi b - \xi c + ab + bc + b^2}.$$

Notice that the denominator on the left-hand side is equal to the denominator on the right-hand side with the addition of ac so it cannot be less. It is therefore sufficient to show that

$$(\xi - a - b)(d + g) + (\xi - b - c)(e + g) \leq (\xi - a - b - c)(\xi - b) + (\xi - a - b - c)(f + g).$$

Starting with the left-hand side we have

$$\begin{aligned}
&(\xi - a - b)(d + g) + (\xi - c - b)(e + g) \\
&= (\xi - a - b - c)(d + g) + c(d + g) + (\xi - a - b - c)(e + g) + a(e + g) \\
&\leq (\xi - a - b - c)(d + g) + c(\xi - a - b - c) \\
&\qquad + (\xi - a - b - c)(e + g) + a(\xi - a - b - c) \\
&= c(\xi - a - b - c) + (\xi - a - b - c)(d + e + g) \\
&\qquad + g(\xi - a - b - c) + a(\xi - a - b - c) \\
&\leq c(\xi - a - b - c) + (\xi - a - b - c)^2 + g(\xi - a - b - c) + a(\xi - a - b - c) \\
&= (\xi - a - b - c)(\xi - b) + g(\xi - a - b - c) \\
&\leq (\xi - a - b - c)(\xi - b) + (f + g)(\xi - a - b - c).
\end{aligned}$$

□

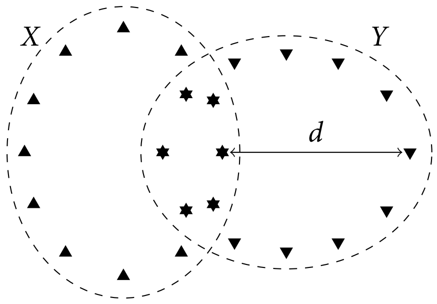


Figure 2.2: X and Y are sets with elements labelled by ▲ and ▼ respectively. Elements in both sets are labelled with ✱. The Hausdorff distance between X and Y is, informally, the greatest distance one must travel if starting from a point in one set and travelling to the closest point in the other. This distance is marked by d in the diagram.

Hausdorff distance

Let (M, d) be a metric space and $\mathcal{M} = 2^M \setminus \{\emptyset\}$. The Hausdorff distance, $\delta_H\colon \mathcal{M} \times \mathcal{M} \to \mathbb{R}^{\geq 0}$, is defined as:

$$\delta_H(X, Y) = \max\left(\max_{x \in X} \min_{y \in Y} d(x, y), \max_{y \in Y} \min_{x \in X} d(x, y)\right),$$

for all $X, Y \in \mathcal{M}$. The Hausdorff distance is a metric on $\mathcal{M}$ [17]. This distance is illustrated in Figure 2.2 by showing the Hausdorff distance between two sets with elements in a metric space. In Chapter 3 we present a new metric on $\mathcal{M}$.

Linkage functions

We conclude by remarking that the linkage functions, which are used in hierarchical clustering, the topic of Section 4.3.2, are not generally metrics. For a metric space, (M, d), and $\mathcal{M} = 2^M \setminus \emptyset$, the *single-linkage distance*, $\delta_{SL}\colon \mathcal{M} \times \mathcal{M} \to \mathbb{R}^{\geq 0}$ is defined as:

$$\delta_{SL}(X, Y) = \min_{x \in X, y \in Y} d(x, y),$$

for all $X, Y \in \mathcal{M}$. While single-linkage is a simple and intuitive distance between sets, it is not a metric since distinct sets can have a distance of zero. The *complete-linkage* and

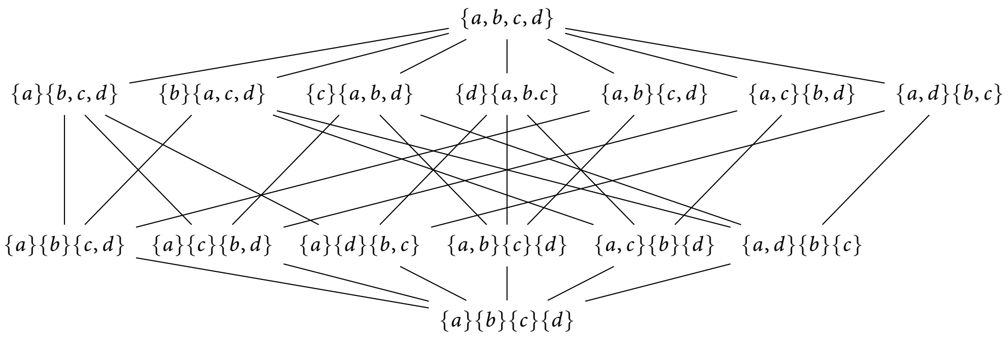


Figure 2.3: The Hasse diagram of the lattice of partitions of a set, $D = \{a, b, c, d\}$ [112]. Each vertex in the graph is a partition $\{C_1, \ldots, C_k\}$ of D which we denote by $C_1, \ldots, C_k$ for clarity.

average-linkage dissimilarities are, in fact, not even distances. These measures will be discussed further in Section 4.3.2.

2.3 Partitions

In this section we look at the properties of partitions of sets, how we can compare partitions and how we can find partitions. Throughout, we assume that $n > 0$ and refer to a set of n elements as an *n-set*.

2.3.1 The space of partitions

Given an n-set D, a k-partition $\mathcal{C} = \{C_1, C_2, \ldots, C_k\}$ of D is a set of $k \in \{1, \ldots, n\}$ nonempty, pairwise-disjoint subsets of D such that $C_1 \cup C_2 \cup \cdots \cup C_k = D$. Following common practice, we will refer to the elements of $\mathcal{C}$ as *clusters*.

Let $\mathcal{P}_D$ be the set of all partitions of D and $\mathcal{C}, \mathcal{C}' \in \mathcal{P}_D$. The partition $\mathcal{C}$ is called a *refinement* of $\mathcal{C}'$ if every element of $\mathcal{C}$ is a subset of some element of $\mathcal{C}'$. We can then say that $\mathcal{C}$ is *finer-than-or-equal-to* $\mathcal{C}'$, which we notate $\mathcal{C} \leq \mathcal{C}'$ (or that $\mathcal{C}'$ is *coarser-than-or-equal-to* $\mathcal{C}$). The relation $\leq$ imposes a partial order on the elements of $\mathcal{P}_D$.

The partially-ordered set $(\mathcal{P}_D, \leq)$ is called the *lattice of partitions* and can be represented in terms of a graph called the *Hasse diagram* associated with $\mathcal{P}_D$. The vertex set of that graph is $\mathcal{P}_D$ and its arc set consists of the pairs $(\mathcal{C}_1, \mathcal{C}_2) \in \mathcal{P}_D \times \mathcal{P}_D$ such that $\mathcal{C}_2 < \mathcal{C}_1$

n	B_n
5	52
10	115,975
15	1,382,958,545
20	5.17×10^{13}
25	4.64×10^{18}

Table 2.1: The value of the Bell number, B_n, for some selected values of n. The value of B_n is equivalent to the number of possible partitions of an n element set.

and there exists no $\mathcal{C}_3 \in \mathcal{P}_D$ such that $\mathcal{C}_2 < \mathcal{C}_3 < \mathcal{C}_1$. Informally, this means that $\mathcal{C}_2$ can be obtained by bisecting one cluster of $\mathcal{C}_1$.

The Hasse diagram of the set $D = \{a, b, c, d\}$ is presented in Figure 2.3. The partition at the top is the coarsest partition (a 1-partition) and the partition at the bottom is the finest partition (an n-partition).

As this example suggests, a set can potentially support a very large number of partitions. For an n-set D the cardinality of $\mathcal{P}_D$ is given by the *Bell number* B_n [12, 151] where $B_0 = 1$ and

$$B_n = \sum_{l=0}^{n-1} B_l \binom{n-1}{l} \qquad \text{for } n \geq 1.$$

Alternatively, B_n is given by Dobiński's formula [44]:

$$B_n = \frac{1}{e} \sum_{l=0}^{\infty} \frac{l^n}{l!} \qquad \text{for } n \geq 0.$$

We illustrate the growth of the Bell number in Table 2.1.

Fuzzy partitions

Partitions can be generalised to fuzzy partitions by letting the clusters be fuzzy sets. Fuzzy sets are collections of objects where each member of a collection has a grade of membership associated to it [171]. Sets are a special case where the grades of membership are binary—objects are either members of a set or they are not. Fuzzy sets and partitions have many applications including classification of data in the natural sciences where membership is not precisely defined.

Formally, a fuzzy set is an ordered pair (C, μ_C) consisting of an underlying set C and a membership function $\mu_C: C \rightarrow [0,1]$. For some fuzzy set (C, μ_C) and element $x \in C$ $\mu_C(x) = 1$ means x is a full member, $\mu_C(x) = 0$ means x is not a member and $0 < \mu_C(x) < 1$ means x is a fuzzy member. If $\mu_C(x) = 0$ or 1 for all $x \in C$ then the object is called by contrast a *crisp set* and may be handled by ordinary set theory [171, 95].

A *k-fuzzy-partition* of an n-set D is a set of $k \in \{1, \ldots, n\}$ fuzzy sets:

$$\{(C_1, \mu_1), (C_2, \mu_2), \ldots, (C_k, \mu_k)\}$$

where $C_1 \cup \cdots \cup C_k = D$, $C_i \neq \varnothing$ for all $i \in \{1, \ldots, k\}$ and $\sum_{i=1}^{k} \mu_i(x) = 1$ for all $x \in D$. Note that the underlying sets of the fuzzy sets (C_i, μ_i) are not necessarily pairwise disjoint, so elements can belong to more than one cluster.

2.3.2 Comparing partitions

An attractive way to find partitions of a set is by means of a partitional clustering method. Many methods exist—we will review some in Section 2.4—and, depending on their respective approach to the problem, they all tend to produce different partitions. To further complicate matters, some methods are nondeterministic so can potentially produce different partitions each time they are used.

For the purposes of comparing and assessing clustering methods it is therefore useful to be able to compare partitions. Many measures have been devised for this purpose, including similarity measures and dissimilarity measures, of which some are metrics. Existing methods fall into four main categories; for two partitions $\mathcal{C}_1$ and $\mathcal{C}_2$ of a set D, these are:

Pair counting which measures the agreement and disagreement between $\mathcal{C}_1$ and $\mathcal{C}_2$ by means of counting pairs of element in D,

Set matching which "matches" clusters in $\mathcal{C}_1$ with clusters in $\mathcal{C}_2$ and measures the similarity between matched sets,

Information theoretic which uses information theory to measure the information and mutual information contained in $\mathcal{C}_1$ and $\mathcal{C}_2$,

Density profile which takes into account the values of the data when computing the measure.

We will review measures belonging to the first three categories in this section. The fourth category consists of a single measure called ADCO which we will analyse in Chapter 3.

To be able to describe the measures in detail, we need to introduce some more terminology. Let $D = \{x_1, x_2, \ldots, x_n\}$ be an n-set with $n \geq 2$ and again, as before, let $\mathcal{C}_1 = \{C_{11}, C_{12}, \ldots, C_{1k}\}$ and $\mathcal{C}_2 = \{C_{21}, C_{22}, \ldots, C_{2k'}\}$ be two partitions of D with k and k' clusters, respectively. We denote, for some $x \in D$, and a partition, $\mathcal{C}$, of D, the cluster in $\mathcal{C}$ that contains x by $\mathcal{C}(x)$. The *confusion matrix* associated with $\mathcal{C}_1$ and $\mathcal{C}_2$ is the $k \times k'$ matrix $[n_{ij}]$ where $n_{ij} = |C_{1i} \cap C_{2j}|$. It is used for the calculation of both pair counting and set matching measures.

To illustrate these definitions and the comparison measures we review below, we will use two example partitions $\mathcal{C}_1^*$ and $\mathcal{C}_2^*$ of the set $D^* = \{1, \ldots, 9\}$ throughout this section:

$$\mathcal{C}_1^* = \{C_{11}^*, C_{12}^*, C_{13}^*\}, \qquad \mathcal{C}_2^* = \{C_{21}^*, C_{22}^*, C_{23}^*, C_{24}^*\} \tag{2.2}$$

where

$$\begin{aligned} &C_{11}^* = \{1,2,3\}, \quad C_{12}^* = \{4,5,6\}, \quad C_{13}^*\{7,8,9\} \quad \text{and} \\ &C_{21}^* = \{1,4\}, \quad C_{22}^* = \{2,3,6\}, \quad C_{23}^* = \{5,8\}, \quad C_{24}^*\{7,9\}. \end{aligned}$$

Then, for example, we have $\mathcal{C}_1^*(3) = C_{11}^*$, and the confusion matrix associated with $\mathcal{C}_1^*$ and

$\mathcal{C}_2^*$ is:

$$[n_{ij}]^* = \begin{bmatrix} 1 & 2 & 0 & 0 \\ 1 & 1 & 1 & 0 \\ 0 & 0 & 1 & 2 \end{bmatrix}.$$

Pair counting

These measures show the agreement and disagreement between $\mathcal{C}_1$ and $\mathcal{C}_2$ by counting pairs of element in D. There are $\binom{n}{2}$ distinct pairs of elements in D. For each distinct pair $(a, b) \in D \times D$ one of the following is true:

$$\begin{aligned} &\mathcal{C}_1(a) = \mathcal{C}_1(b) \text{ and } \mathcal{C}_2(a) = \mathcal{C}_2(b), \\ &\mathcal{C}_1(a) \neq \mathcal{C}_1(b) \text{ and } \mathcal{C}_2(a) \neq \mathcal{C}_2(b), \\ &\mathcal{C}_1(a) = \mathcal{C}_1(b) \text{ and } \mathcal{C}_2(a) \neq \mathcal{C}_2(b), \\ &\mathcal{C}_1(a) \neq \mathcal{C}_1(b) \text{ and } \mathcal{C}_2(a) = \mathcal{C}_2(b). \end{aligned}$$

Counting the number of element in each category gives us four counts which are defined formally as:

$$\begin{aligned} N_{11} &= \left|\{(a,b) \in D \times D : \mathcal{C}_1(a) = \mathcal{C}_1(b) \text{ and } \mathcal{C}_2(a) = \mathcal{C}_2(b)\}\right|, \\ N_{00} &= \left|\{(a,b) \in D \times D : \mathcal{C}_1(a) \neq \mathcal{C}_1(b) \text{ and } \mathcal{C}_2(a) \neq \mathcal{C}_2(b)\}\right|, \\ N_{10} &= \left|\{(a,b) \in D \times D : \mathcal{C}_1(a) = \mathcal{C}_1(b) \text{ and } \mathcal{C}_2(a) \neq \mathcal{C}_2(b)\}\right|, \\ N_{01} &= \left|\{(a,b) \in D \times D : \mathcal{C}_1(a) \neq \mathcal{C}_1(b) \text{ and } \mathcal{C}_2(a) = \mathcal{C}_2(b)\}\right|. \end{aligned}$$

The size of N_{11} and N_{00} are considered to be measurements of *agreement* between $\mathcal{C}_1$ and $\mathcal{C}_2$, while N_{10} and N_{01} are considered measurements of *disagreement*.

The measures based on pair counting can all be expressed in terms of these four counts. Clearly

$$N_{11} + N_{00} + N_{10} + N_{01} = \binom{n}{2}$$

is always satisfied. The quantities N_{11}, N_{00}, N_{10} and N_{01} can all be obtained from the confusion matrix using the following formulæ [83]:

$$\begin{aligned}
N_{11} &= \frac{1}{2}\sum_{i=1}^{k}\sum_{j=1}^{k'} n_{ij}(n_{ij}-1),\\
N_{00} &= \frac{1}{2}\left(n^2 + \sum_{i=1}^{k}\sum_{j=1}^{k'} n_{ij}^2 - \sum_{i=1}^{k}\left(\sum_{j=1}^{k'} n_{ij}\right)^2 - \sum_{j=1}^{k'}\left(\sum_{i=1}^{k} n_{ij}\right)^2\right),\\
N_{10} &= \frac{1}{2}\left(\sum_{i=1}^{k}\left(\sum_{j=1}^{k'} n_{ij}\right)^2 - \sum_{i=1}^{k}\sum_{j=1}^{k'} n_{ij}^2\right),\\
N_{01} &= \frac{1}{2}\left(\sum_{j=1}^{k'}\left(\sum_{i=1}^{k} n_{ij}\right)^2 - \sum_{i=1}^{k}\sum_{j=1}^{k'} n_{ij}^2\right).
\end{aligned}$$

The four counts for our example partitions $\mathcal{C}_1^*$ and $\mathcal{C}_2^*$ are:

$$N_{11} = 2, \quad N_{00} = 23, \quad N_{10} = 7, \quad N_{01} = 4.$$

One of the simplest pair counting measures between $\mathcal{C}_1$ and $\mathcal{C}_2$ is the widely-used *Rand index* $\Delta_{\mathcal{R}}$ introduced in [130] and defined as:

$$\Delta_{\mathcal{R}}(\mathcal{C}_1, \mathcal{C}_2) = (N_{11} + N_{00}) / \binom{n}{2}.$$

The Rand index is a similarity measure with a lower bound of 0 and an upper bound of 1. Hubert and Arabie [83] use the following variation:

$$\Delta_{\mathcal{HA}}(\mathcal{C}_1, \mathcal{C}_2) = (N_{11} + N_{00} - N_{10} - N_{01}) / \binom{n}{2}.$$

Wallace [160] introduced two asymmetric measures for comparing $\mathcal{C}_1$ and $\mathcal{C}_2$ defined as:

$$\mathcal{W}_I(\mathcal{C}_1, \mathcal{C}_2) = \frac{N_{11}}{\sum_{i=1}^{k} |C_{i1}|(|C_{1i}| - 1)/2},$$

Name	Measure
Rand index	$\Delta_{\mathcal{R}}(\mathcal{C}_1^*, \mathcal{C}_2^*) \approx 0.694$
Wallace measures	$\mathcal{W}_I(\mathcal{C}_1^*, \mathcal{C}_2^*) \approx 0.222$
	$\mathcal{W}_{II}(\mathcal{C}_1^*, \mathcal{C}_2^*) \approx 0.333$
Fowlkes & Mallows	$\Delta_{\mathcal{F}}(\mathcal{C}_1^*, \mathcal{C}_2^*) \approx 0.272$
Jaccard coefficient	$\Delta_{\mathcal{J}}(\mathcal{C}_1^*, \mathcal{C}_2^*) \approx 0.154$
Merkin metric	$\Delta_{\mathcal{M}}(\mathcal{C}_1^*, \mathcal{C}_2^*) = 22.00$

Table 2.2: Various pair counting based measures applied to the example partitions $\mathcal{C}_1^*$ and $\mathcal{C}_2^*$ (2.2).

and

$$\mathcal{W}_{II}(\mathcal{C}_1, \mathcal{C}_2) = \frac{N_{11}}{\sum_{j=1}^{k'} |C_{2j}|(|C_{2j}| - 1)/2}.$$

$\mathcal{W}_I$ and $\mathcal{W}_{II}$ represent the probabilities that, for a pair of elements $(a, b) \in D \times D$ with $\mathcal{C}_1(a) = \mathcal{C}_1(b)$, we also have $\mathcal{C}_2(a) = \mathcal{C}_2(b)$, and vice versa. A symmetric similarity measure for $\mathcal{C}_1$ and $\mathcal{C}_2$ can be obtained by taking the geometric mean of the Wallace measures:

$$\Delta_{\mathcal{F}}(\mathcal{C}_1, \mathcal{C}_2) = \sqrt{\mathcal{W}_I(\mathcal{C}_1, \mathcal{C}_2)\mathcal{W}_{II}(\mathcal{C}_1, \mathcal{C}_2)}.$$

This measure was also introduced independently by Fowlkes and Mallows in [59].

The *Jaccard coefficient* $\Delta_{\mathcal{J}}$ is another widely-used similarity measure and is defined, for $\mathcal{C}_1$ and $\mathcal{C}_2$, as:

$$\Delta_{\mathcal{J}}(\mathcal{C}_1, \mathcal{C}_2) = \frac{N_{11}}{N_{10} + N_{01} + N_{11}}.$$

It should be noted that all measures reviewed so far are similarity measures. The *Mirkin metric* $\Delta_{\mathcal{M}}$ for measuring dissimilarity between $\mathcal{C}_1$ and $\mathcal{C}_2$ was originally introduced in [116] and is defined as

$$\Delta_{\mathcal{M}}(\mathcal{C}_1, \mathcal{C}_2) = \sum_{i=1}^{k} |C_{1i}|^2 + \sum_{j=1}^{k'} |C_{1j}|^2 - 2 \sum_{i=1}^{k} \sum_{j=1}^{k'} n_{ij}^2,$$

where $[n_{ij}]$ again denotes the confusion matrix associated with $\mathcal{C}_1$ and $\mathcal{C}_2$. As it turns out,

$$\Delta_{\mathcal{M}}(\mathcal{C}_1, \mathcal{C}_2) = 2(N_{10} + N_{01}).$$

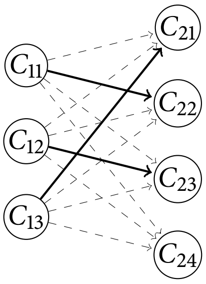


Figure 2.4: The clusters belonging to two partitions $\{C_{11}, \ldots, C_{13}\}$ and $\{C_{21}, \ldots, C_{24}\}$ are shown with all similarities between pairs shown dashed. The solid lines represent one possible matching between the clusters. In this case, the matching is an injection.

The closely related measure:

$$\Delta_{\mathcal{AB}}(\mathcal{C}_1, \mathcal{C}_2) = (N_{10} + N_{01}) / \binom{n}{2}$$

is also a metric on $\mathcal{P}_D$ and was used by Mirkin and Chernyi [117] and Arabie and Boorman [5].

The values of the reviewed pair counting measures when applied to $\mathcal{C}_1^*$ and $\mathcal{C}_2^*$ are given in Table 2.2.

Set matching

Set matching measures for partitions $\mathcal{C}_1$ and $\mathcal{C}_2$ are based on comparisons between matched pairs of clusters. Each pair to be compared consists of one element of $\mathcal{C}_1$ and one of $\mathcal{C}_2$. All of the measures that we review here are based on the confusion matrix, meaning that they use the cardinality of the intersection between two clusters as a similarity measure. The differences between the measures are essentially due to the way that they find the matched pairs of clusters for comparison.

As before, let D be an n-set and $\mathcal{C}_1 = \{C_{11}, \ldots, C_{1k}\}$ and $\mathcal{C}_2 = \{C_{21}, \ldots, C_{1k'}\}$, where $k, k' \geq 1$, denote two partitions of D. Also assume, without loss of generality, that $k' \geq k$. Then, a *matching* between $\mathcal{C}_1$ and $\mathcal{C}_2$ is a function $\sigma \colon \{1, \ldots, k\} \to \{1, \ldots, k'\}$.

Meilă and Heckerman [114] introduced a set matching measure for measuring the similarity between $\mathcal{C}_1$ and $\mathcal{C}_2$. It finds a matching σ using the following heuristic: let $[n_{ij}]$ be the $k \times k'$ confusion matrix associated with $\mathcal{C}_1$ and $\mathcal{C}_2$. Compute $n_{ab} = \arg\max\{n_{ij} : 1 \leq i \leq k, 1 \leq j \leq k'\}$ and put $\sigma(a) \leftarrow b$. Repeat the process for the $(k-1) \times (k'-1)$ submatrix obtained by deleting row a and column b, and so on until k matches have been made. This process clearly finds an injection for σ and then the similarity of $\mathcal{C}_1$ and $\mathcal{C}_2$ based on that matching is:

$$\Delta_{\mathcal{H}}(\mathcal{C}_1, \mathcal{C}_2) = \frac{1}{n} \sum_{i=1}^{k} n_{i\sigma(i)}.$$

A second similarity measure on $\mathcal{P}_D$, denoted $\Delta_{\mathcal{L}}$ and introduced by Larsen and Aone [102] simply computes a maximal match for each cluster in $\mathcal{C}_1$ and $\mathcal{C}_2$:

$$\Delta_{\mathcal{L}}(\mathcal{C}_1, \mathcal{C}_2) = \frac{1}{k} \sum_{i=1}^{k} \max_{1 \leq j \leq k'} \frac{2n_{ij}}{|C_{1i}| + |C_{2j}|}.$$

A dissimilarity measure $\Delta_{\mathcal{V}}$, which is a metric on $\mathcal{P}_D$, was introduced by van Dongen [157]. This measure, again, computes maximal matches for each cluster in $\mathcal{C}_1$ and $\mathcal{C}_2$:

$$\Delta_{\mathcal{V}}(\mathcal{C}_1, \mathcal{C}_2) = 2n - \sum_{i=1}^{k} \max_{1 \leq j \leq k'} n_{ij} - \sum_{j=1}^{k'} \max_{1 \leq i \leq k} n_{ij}.$$

A second metric on $\mathcal{P}_D$ denoted by $\Delta_{\mathcal{CE}}$ is due to Meilă [112] and is known by the name *classification error*. Like $\Delta_{\mathcal{H}}$, this measure computes an injection for σ, but instead of using a heuristic finds a globally optimal injection in the set S_k of all possible injections from $\{1, \ldots, k\}$ to $\{1, \ldots, k'\}$:

$$\Delta_{\mathcal{CE}}(\mathcal{C}_1, \mathcal{C}_2) = 1 - \frac{1}{n} \max_{\sigma \in S_k} \sum_{i=1}^{k} n_{i\sigma(i)}.$$

Note that the injection, σ, can be found in polynomial time. Also,

$$\Delta_{\mathcal{CE}}(\mathcal{C}_1, \mathcal{C}_2) \leq 1$$

for all partitions $\mathcal{C}_1$ and $\mathcal{C}_2$ of D. The values of the set matching measures when applied to

Name	Measure
Meilă & Heckerman	$\Delta_{\mathcal{H}}(\mathcal{C}_1^*, \mathcal{C}_2^*) \approx 0.556$
Larsen & Aone	$\Delta_{\mathcal{L}}(\mathcal{C}_1^*, \mathcal{C}_2^*) \approx 0.622$
Van Dongen metric	$\Delta_{\mathcal{V}}(\mathcal{C}_1^*, \mathcal{C}_2^*) = 7.000$
Classification Error	$\Delta_{\mathcal{CE}}(\mathcal{C}_1^*, \mathcal{C}_2^*) \approx 0.445$

Table 2.3: Various set matching based measures applied to the example partitions $\mathcal{C}_1^*$ and $\mathcal{C}_2^*$ (2.2).

our example partitions $\mathcal{C}_1^*$ and $\mathcal{C}_2^*$ are given in Table 2.3.

Meilă [113] and Bae et al. [8] point out that all of these measures suffer from the so-called "problem of matching" which we will now illustrate. Given a partition $\mathcal{C} = \{C_1, \ldots, C_k\}$ with $k \geq 2$ equally sized clusters, we can obtain a partition $\mathcal{C}'$ from $\mathcal{C}$ by moving a fraction f of the objects from each cluster C_i to C_{i+1}, with the indices taken modulo k. We can obtain a further partition $\mathcal{C}''$ from $\mathcal{C}$ by taking the same fraction from each cluster in $C \in \mathcal{C}$ and distributing the objects evenly among all other clusters in $\mathcal{C} \setminus \{C\}$. Our intuition would be that the similarity between $\mathcal{C}$ and $\mathcal{C}'$ is not the same as the similarity between $\mathcal{C}$ and $\mathcal{C}''$. However,

$$\Delta_{\mathcal{H}}(\mathcal{C}, \mathcal{C}') = \Delta_{\mathcal{H}}(\mathcal{C}, \mathcal{C}''), \quad \Delta_{\mathcal{L}}(\mathcal{C}, \mathcal{C}') = \Delta_{\mathcal{L}}(\mathcal{C}, \mathcal{C}''),$$
$$\Delta_{\mathcal{V}}(\mathcal{C}, \mathcal{C}') = \Delta_{\mathcal{V}}(\mathcal{C}, \mathcal{C}''), \quad \Delta_{\mathcal{CE}}(\mathcal{C}, \mathcal{C}') = \Delta_{\mathcal{CE}}(\mathcal{C}, \mathcal{C}'')$$

whenever $0 < f < 1/2$.

To give a numerical example, let $D = \{1, \ldots, 15\}$,

$$\mathcal{C} = \big\{\{1,2,3,4,5\}, \{6,7,8,9,10\}, \{11,12,13,14,15\}\big\}$$

be a partition of D and $f = 2/5$. Then

$$\mathcal{C}' = \big\{\{1,2,3,14,15\}, \{6,7,8,4,5\}, \{11,12,13,9,10\}\big\}$$

and

$$\mathcal{C}'' = \big\{\{1,2,3,9,14\},\{6,7,8,4,15\},\{11,12,13,5,10\}\big\}$$

are partitions of D obtained from $\mathcal{C}$ as described above. In this case we have

$$\Delta_{\mathcal{H}}(\mathcal{C},\mathcal{C}') = \Delta_{\mathcal{H}}(\mathcal{C},\mathcal{C}'') = 3/5, \quad \Delta_{\mathcal{L}}(\mathcal{C},\mathcal{C}') = \Delta_{\mathcal{L}}(\mathcal{C},\mathcal{C}'') = 3/5,$$
$$\Delta_{\mathcal{V}}(\mathcal{C},\mathcal{C}') = \Delta_{\mathcal{V}}(\mathcal{C},\mathcal{C}'') = 12, \quad \Delta_{\mathcal{CE}}(\mathcal{C},\mathcal{C}') = \Delta_{\mathcal{CE}}(\mathcal{C},\mathcal{C}'') = 2/5.$$

Information theoretic

Two measures which use information theory are *Normalized Mutual Information* [60] and *Variation of Information* [113]. For the remainder of this section we will focus on the latter and refer the reader to [60] for details on the former.

Variation of Information is based on both how much information is contained in each partition and how much information one partition contains about the other (their mutual information).

Let $\mathcal{C} = \{C_1,\dots,C_k\}$, where $k \geq 1$, be partition of an n-set D. Then the *information* contained in $\mathcal{C}$ is measured by:

$$H(\mathcal{C}) = -\sum_{i=1}^{k} P_{\mathcal{C}}(i)\log_b P_{\mathcal{C}}(i), \tag{2.3}$$

where

$$P_{\mathcal{C}}(i) = \frac{|C_i|}{k}, \qquad \text{for } i = 1,\dots,k.$$

Informally, $P_1(i)$, is the probability that an object picked randomly from D is in cluster C_{1i}. This measure is sometimes called *entropy*. The base of the logarithm determines the unit of information; for example, the bases $b = 2$, $b = e$ and $b = 10$ give the information in so-called *bits*, *nits* and *Hartleys*, respectively [see 98].

Now, with $\mathcal{C}_1 = \{C_{11},\dots,C_{1k}\}$ and $\mathcal{C}_2 = \{C_{21},\dots,C_{1k'}\}$, where $k,k' \geq 1$, the mutual information, $I(\mathcal{C}_1, \mathcal{C}_2)$, between $\mathcal{C}_1$ and $\mathcal{C}_2$ is given by:

$$I(\mathcal{C}_1, \mathcal{C}_2) = \sum_{i=1}^{k} \sum_{i=1}^{k'} P_{12}(i, j) \log_b \frac{P_{12}(i, j)}{P_{\mathcal{C}_1}(i) P_{\mathcal{C}_2}(j)}, \tag{2.4}$$

where

$$P_{12}(i, j) = \frac{|C_{1i} \cap C_{2j}|}{n}, \qquad \text{for } i = 1, \ldots, k, j = 1, \ldots, k'.$$

Informally, $P_{12}(i, j)$ is the probability that an object picked randomly from D is in both C_{1i} and C_{2j}.

The *Variation of Information*, $\Delta_{\mathcal{VI}}$, between $\mathcal{C}_1$ and $\mathcal{C}_2$ is then defined as:

$$\Delta_{\mathcal{VI}}(\mathcal{C}_1, \mathcal{C}_2) = H(\mathcal{C}_1) + H(\mathcal{C}_2) - 2I(\mathcal{C}_1, \mathcal{C}_2).$$

As it turns out, $\Delta_{\mathcal{VI}}$ is a metric on $\mathcal{P}_D$ and has some attractive properties. These include that it is n-invariant, meaning its value depends only on the relative sizes of the clusters in $\mathcal{C}_1$ and $\mathcal{C}_2$ and not on the size of n. Further, it is bounded for all n by

$$\Delta_{\mathcal{VI}}(\mathcal{C}_1, \mathcal{C}_2) \leq \log_b n.$$

Finally, if $\max(k, k') \leq k^*$ where $k^* \leq \sqrt{n}$ then

$$\Delta_{\mathcal{VI}}(\mathcal{C}_1, \mathcal{C}_2) \leq 2 \log_b k^*$$

holds.

The values of Variation of Information and its components applied to our example partitions, $\mathcal{C}_1^*$ and $\mathcal{C}_2^*$, are given in Table 2.4.

We conclude this section with mentioning a further measure called ADCO, which is a density profile based measure. We will save discussion of this measure until we introduce our own Assignment Metric later since these two metrics share a unique feature possessed by no other measure discussed in this section. Namely, they take into account that the

Name	Measure
Information	$H(\mathcal{C}_1^*) \approx 1.585$
	$H(\mathcal{C}_2^*) \approx 1.975$
Mutual information	$I(\mathcal{C}_1^*, \mathcal{C}_2^*) \approx 0.834$
Variation of Information	$\Delta_{\mathcal{VI}}(\mathcal{C}_1^*, \mathcal{C}_2^*) \approx 1.891$

Table 2.4: Variation of Information and its components applied to the example partitions $\mathcal{C}_1^*$ and $\mathcal{C}_2^*$ (2.2) using base 2 for the logarithms in equations (2.3) and (2.4).

partitions to be compared are partitions of a dataset themselves, with elements that have a metric defined on them.

2.4 Partitional clustering

The problem of finding meaningful partitions of a dataset is called *partitional clustering*. Broadly speaking, the applications of partitional clustering fall into two categories: *data reduction* and *object classification*.

Data reduction may be necessary when a dataset is large and it is deemed that only an essence of the data is wanted for a particular application. For example, if geographical data is to be displayed on a map then a large dataset may not be desirable due to the visual clutter it would create. Clustering can be used to reduce the dataset into a more visually appealing and usable subset.

Object classification is concerned with grouping data into a number of classes. For example, given a dataset obtained by market research one may wish to find different classes of consumers in order to observe their habits and predict future behaviour. A further example is document clustering which is an important area concerned with clustering on datasets consisting of objects written in a natural language. Search engines such as those found on the World Wide Web use document clustering to suggest, among other things, “similar” documents to the one in which a user is currently interested [154].

2.4.1 Criteria

Informally, a meaningful partition of a dataset (D, d) is one which contains clusters that are homogeneous—meaning objects belonging to the same cluster are similar—and well-separated—meaning objects belonging to different clusters are dissimilar, according to d.

The task of judging a particular partition of a dataset based on these informal standards is often a highly subjective one but, nevertheless, many objective criteria have been devised for the purposes of automatic clustering. The aim of a partitional clustering algorithm is to find a globally optimal solution, that is a partition which has maximum homogeneity or separation or both, according to a particular criterion.

Dissimilarity based criteria

The most general criteria for homogeneity and separation are defined only using dissimilarities. To make this more precise, let D be an n-set with $n \geq 1$, $d \colon D \times D \to \mathbb{R}^{\geq 0}$ be a dissimilarity on D and $C \subseteq D$ be some cluster.

Criteria that measure the homogeneity of C include the *diameter* of C, which is defined as the maximum dissimilarity between two members of C:

$$\max_{x,y \in C} d(x, y),$$

the *radius* of C, which is defined as the minimum of the maximum dissimilarities between each member and another member of C:

$$\min_{x \in C_i} \max_{y \in C} d(x, y)$$

the *star* of C, which is defined as the minimum of the sums of dissimilarities between each member and every other member of C:

$$\min_{c \in C} \sum_{x \in C} d(x, c),$$

and the *clique* of C, which is defined as the sum of dissimilarities between each pair of members of C:

$$\sum_{x,y \in C} d(x, y).$$

Criteria that measure the separation of C from every other cluster include the *split* of C, which is defined as the minimum dissimilarity between a member of C and an element in $D \setminus C$:

$$\min_{x \in C, y \in D \setminus C} d(x, y),$$

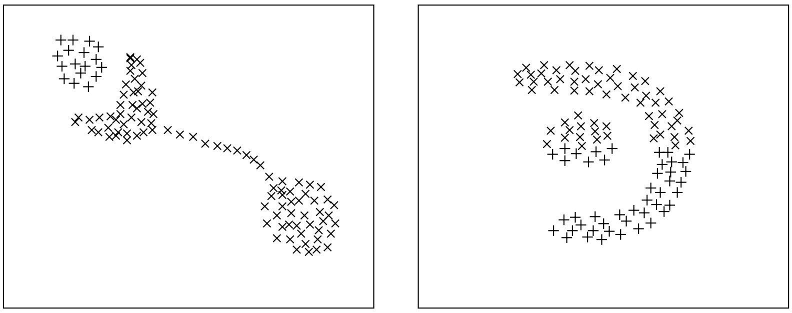
and the *cut* of C, which is defined as the sum of dissimilarities between all members of C and all elements in $D \setminus C$:

$$\sum_{x \in C} \sum_{y \in D \setminus C} d(x, y).$$

Dissimilarity based criteria for partitions can then be obtained from the above measures for clusters by simply taking the sum over all clusters in a partition. We call these criteria *sum-of-diameters*, *sum-of-radii*, *sum-of-stars* and so on. Alternatively one could simply take the maximum or minimum value, as appropriate, over the clusters which we would call *max-diameter*, *max-radius*, *min-cut* and so on. The aim of a clustering method is then to minimise a criterion for homogeneity or maximise a criterion for separation.

Some criteria for homogeneity are equivalent to criteria for separation. Most notably, minimising sum-of-cliques is equivalent to maximising sum-of-cuts. Such criteria are therefore criteria for both homogeneity and separation.

As it turns out, criteria which only measure one or the other, like min-split and max-diameter, are often conflicting. For example, maximising min-split produces clusters with



(i) Chaining effect.

(ii) Dissection effect.

Figure 2.5: The chaining effect can produce clusters with poor homogeneity and the dissection effect can produce clusters with poor separation.

poor homogeneity, called the *chaining effect*, and minimising max-diameter results in clusters with poor separation, called the *dissection effect.* Figure 2.5 (i) shows a 2-clustering displaying poor cluster homogeneity due to the chaining effect and Figure 2.5 (ii) shows a 2-clustering displaying poor separation due to the dissection effect. These are the clusterings produced by maximising min-split and minimising max-diameter, respectively.

Such criteria are therefore not so useful on their own, but one way to overcome this problem is to consider a *bicriterion*, that is simultaneously consider a criterion for homogeneity and a criterion for separation [40].

Scatter matrix based criteria

If the dataset of interest is embedded in m-dimensional Euclidean space, with $m \geq 1$, then the following criteria based on the scatter or dispersion matrix of Wilks [164] are possible. To make this more precise, let d_E denote the Euclidean distance on D and denote a vector $\mathbf{v} \in \mathbb{R}^m$ by $\mathbf{v} = (v_1, \ldots, v_m)$. Suppose $C = \{\mathbf{x}_1, \ldots, \mathbf{x}_n\} \subset \mathbb{R}^m$, then the *total scatter matrix* of C, $\mathbf{T}_C$, is the $m \times m$ matrix defined as:

$$\mathbf{T}_C = \sum_{i=1}^{n} (\mathbf{x}_i - \mathbf{c})^{\mathrm{T}} (\mathbf{x}_i - \mathbf{c})$$

where $\mathbf{c}$ is the mean value of D.

For $\mathcal{C} = \{C_1, \ldots, C_k\}$, a partition of an n-set $D = \{\mathbf{x}_1, \ldots, \mathbf{x}_n\}$, two further matrices are associated. The *within-cluster scatter matrix*, $\mathbf{W}_{\mathcal{C}}$, is defined as:

$$\mathbf{W}_{\mathcal{C}} = \sum_{i=1}^{k} \mathbf{T}_{C_i},$$

where $\mathbf{T}_{C_i}$ is the total scatter matrix for cluster C_i of $\mathcal{C}$. The *between-cluster scatter matrix*, $\mathbf{B}_{\mathcal{C}}$, is defined as:

$$\mathbf{B}_{\mathcal{C}} = \sum_{i=1}^{k} |C_i|(\mathbf{c}_i - \bar{\mathbf{x}})(\mathbf{c}_i - \bar{\mathbf{x}})^{\mathrm{T}}$$

where $\mathbf{c}_i$ is the mean of cluster C_i and $\bar{\mathbf{x}}$ is the mean of D. These matrices are related by the equality

$$\mathbf{T}_D = \mathbf{W}_{\mathcal{C}} + \mathbf{B}_{\mathcal{C}}.$$

A popular criterion used in clustering algorithms is the minimisation of the trace of $\mathbf{W}_{\mathcal{C}}$, denoted $\mathrm{tr}(\mathbf{W}_{\mathcal{C}})$. Since

$$\mathrm{tr}(\mathbf{T_D}) = \mathrm{tr}(\mathbf{W}_{\mathcal{C}}) + \mathrm{tr}(\mathbf{B}_{\mathcal{C}}),$$

and $\mathrm{tr}(\mathbf{T}_D)$ is a constant, it follows that minimising $\mathrm{tr}(\mathbf{W}_{\mathcal{C}})$ is equivalent to maximising $\mathrm{tr}(\mathbf{B}_{\mathcal{C}})$.

Intriguingly, $\mathrm{tr}(\mathbf{W}_{\mathcal{C}})$ is also equivalent to the sum over all clusters $C \in \mathcal{C}$ of the sum of Euclidean distances squared between each element $\mathbf{x} \in C$ and the mean $\mathbf{c}$ of C:

$$\mathrm{tr}(\mathbf{W}_{\mathcal{C}}) = \sum_{i=1}^{k} \sum_{\mathbf{x} \in C_i} d_E^2(\mathbf{x}, \mathbf{c}_i). \tag{2.5}$$

This can be seen easily by examining the main diagonal of the total scatter matrix for each cluster $C \in \mathcal{C}$, as illustrated here:

$$\mathbf{T}_C = \sum_{\mathbf{x}\in C} \begin{bmatrix} (x_1 - c_1)^2 & (x_1 - c_1)(x_2 - c_2) & \cdots & (x_1 - c_1)(x_m - c_m) \\ (x_2 - c_2)(x_1 - c_1) & (x_2 - c_2)^2 & \cdots & (x_2 - c_2)(x_m - c_m) \\ \vdots & \vdots & \ddots & \vdots \\ (x_m - c_m)(x_1 - c_1) & (x_m - c_m)(x_2 - c_2) & \cdots & (x_m - c_m)^2 \end{bmatrix}$$

where $\mathbf{c} = (c_1, \ldots, c_m)$ is the mean of C.

Similarly, $\mathrm{tr}(\mathbf{B}_{\mathcal{C}})$ is equivalent to the sum of Euclidean distances squares between the mean $\mathbf{c}$ of a cluster C and $\bar{\mathbf{x}}$, the mean of D, multiplied by the size of C:

$$\mathrm{tr}(\mathbf{B}_{\mathcal{C}}) = \sum_{i=1}^{k} |C_i| d_E^2(\mathbf{c}_i, \bar{\mathbf{x}}). \tag{2.6}$$

For this reason, the trace of $\mathbf{W}_{\mathcal{C}}$ is often called "sum-of-squares", but this name is ambiguous since any criterion utilising sums of distances squared could have this name. We therefore call the criterion shown in equation (2.5) the *centroid-distance* of $\mathcal{C}$, due to measuring the distances between elements and centroids, and in general call any criterion using sums of distances squared a *sum-of-squares criterion*. Note that when the centroid-distance of a partition is calculated using equation (2.5) we can use any metric in place of d_E. But if it is calculated using the scatter matrices then it is implicitly using Euclidean distance.

The relationship between equations (2.5) and (2.6) can also be established by the *Huygens-Steiner*, or *parallel-axis*, theorem. This theorem states that, for any cluster $C \subset \mathbb{R}^m$ with mean $\mathbf{c}$ and any element $\mathbf{s} \in \mathbb{R}^m$:

$$\sum_{\mathbf{x}\in C} d_E^2(\mathbf{x}, \mathbf{s}) = d_E^2(\mathbf{c}, \mathbf{s}) \cdot |C| + \sum_{\mathbf{x}\in C} d_E^2(\mathbf{x}, \mathbf{c}). \tag{2.7}$$

We now use this theorem to establish a relationship between centroid-distance and sum-of-cliques. With some $C \subset \mathbb{R}^m$ as before, we beginning with equation (2.7) and assign some $\mathbf{y} \in C$ to $\mathbf{s}$ and sum over all $\mathbf{y} \in C$:

$$\sum_{\mathbf{x},\mathbf{y} \in C} d_E^2(\mathbf{x},\mathbf{y}) = \sum_{\mathbf{y} \in C} d_E^2(\mathbf{c},\mathbf{y}) \cdot |C| + \sum_{\mathbf{x} \in C} d_E^2(\mathbf{x},\mathbf{c}) \cdot |C|$$

and, since d_E is symmetric,

$$\frac{\sum_{\mathbf{x},\mathbf{y} \in C} d_E^2(\mathbf{x},\mathbf{y})}{|C|} = 2 \sum_{\mathbf{x} \in C} d_E^2(\mathbf{x},\mathbf{c}). \tag{2.8}$$

So when we use squared Euclidean distance as a dissimilarity measure, centroid-distance for a cluster $C \subset \mathbb{R}^m$ is equivalent to the sum-of-cliques of C over twice its cardinality. Since d_E is symmetric this equates to only counting pairwise distances once each.

It should also be noted that centroid-distance is, in fact, simply a more general version of sum-of-stars where the centre need not be a member of the dataset under consideration but is a member of some underlying metric space, which in the above case was $(\mathbb{R}^m, d_E)$.

As mentioned, we generally call any criterion involving a metric squared a sum-of-squares criterion. We call the numerator of the left-hand side of equation (2.8) *all-squares*, since we sum over all pairs of distances. Since sum-of-cliques can generally be used with any dissimilarity measure we generally count each pair of elements twice, once in each direction, but for a metric this is obviously redundant.

Friedman and Rubin [61] suggested two further criteria based on scatter matrices, these are the minimisation of the determinant of $\mathbf{W}_\mathcal{C}$ (denoted $\det(\mathbf{W}_\mathcal{C})$), which is sometimes called *generalised variance*, and the maximisation of the trace of $\mathbf{B}_\mathcal{C}{\mathbf{W}_\mathcal{C}}^{-1}$. These criteria tend to produce clusters of similar shape and size as centroid-distance with the Euclidean distance [111].

A further three criteria based on scatter matrices are discussed in [111], these are $\prod_{i=1}^k \det(\mathbf{W}_i)^{|C_i|}$, which is a generalisation of $\det(\mathbf{W})$ that allows cluster of different shapes; $n \log \det(\mathbf{W}) - 2\sum_{i=1}^k |C_i| \log |C_i|$; and $\sum_{i=1}^k (|C_i| \log \det(\mathbf{W}_i) - 2|C_i| \log |C_i|)$. Finally, in

[110] the criterion $\sum_{i=1}^{k} \det(\mathbf{W}_i)^{\frac{1}{m}}$ was introduced.

It is worth noting that all of the criteria based on the scatter matrix tend to produce clusters of an ellipsoidal shape; clusters with other shapes will either not be found or end up divided into smaller clusters. Most of them also tend to produce clusters which are *linearly-separable*—meaning they are separated by hyperplanes [111].

Other criteria

Some criteria are based on similarities instead of dissimilarities. One such example is the *average entity stability* [137]. This criterion considers an object to be stable if it is more attracted to the rest of the cluster it is currently in than to any other cluster. Give, a similarity measure, s, the attraction between an object and a cluster is defined as the average similarity between the object and the members of that cluster, with respect to s. The clustering criterion is to maximise the average entity stability over the whole dataset.

Criteria based on information theory are also possible. Wallace and Boulton [159] introduce a clustering program called SNOB which attempts to optimise one such information measure.

2.4.2 Computational complexity

Given the potentially huge space of possible partitions of a set, partitional clustering is intuitively a hard problem. In fact, as we will see below, it has been shown that many partitional clustering problems are NP-complete.

We first need to make precise what we mean by a "partitional clustering problem". Given a set D and a criterion, a partition in $\mathcal{P}_D$ that is a global minimum or maximum, whichever is appropriate, according to that criterion is called an *optimal partition*. When we speak of a particular criterion we qualify this name appropriately, so for example a *centroid-distance optimal partition* is a partition which globally minimises the centroid-distance criterion.

For brevity, we will refer to the problem of finding an optimal partition according to some criterion simply as CLUSTERING and, similarly, whenever we speak of a particular criterion we will qualify the name appropriately, so for example CENTROID-DISTANCE CLUSTERING is the problem of finding an optimal partition with respect to the centroid-distance criterion.

An NP-completeness result refers to a decision problem. Since clustering problems are optimisation problems, whenever one is said to be NP-complete we are actually referring to the decision problem derived from the optimisation problem. The general decision problem version of CLUSTERING is simply stated, the specific clustering problems are identical except the value for the criterion f is fixed:

CLUSTERING

INSTANCE: An n-set D, the number of clusters desired $k \in \{1, \ldots, n\}$, a criterion $f\colon \mathcal{P}_D \to \mathbb{R}^+$, which involves k in some way, and a bound $B \in \mathbb{R}^+$.

QUESTION: Does there exist a k-partition $\mathcal{C} \in \mathcal{P}_D$ such that $f(\mathcal{C}) \leq B$?

Optimisation problems also have corresponding *approximation problems*. The n-approximation problem corresponding to an optimisation problem is the problem of finding solutions with an objective criterion value within n times the value for an optimal solution. There is also a decision problem corresponding to the approximation problem which, again, we will be referring to implicitly when speaking of NP-completeness results.

SUM-OF-CUTS CLUSTERING (S-CUTS) is seen to be NP-complete by considering the classic NP-complete graph problem MAX CUT [89, 67]. It turns out that MAX CUT is simply a special case of S-CUTS with $k = 2$ and where a dissimilarity with values in $\{0, 1\}$ is used [63]. Since an optimal partition according to the sum-of-cuts criterion is also an optimal partition according to the sum-of-cliques criterion, it is clear that SUM-OF-CLIQUES CLUSTERING (S-CLIQUES) is NP-complete also. An approximate solution to S-CUTS is possible to find in polynomial time, but, interestingly, the approximation problem of

S-Cliques remains NP-complete [138].

A number of partitional clustering problems were shown to be in NP-complete by Brücker [22]. Among these was Max-Diameter Clustering (M-Diam), a result which, as it turns out, is directly deducible from the earlier results of Sahni and Gonzalez [138]. M-Diam remains NP-complete for $k = 3$ and when all dissimilarities are in $\{0,1\}$ [62].

It was shown in Gonzalez [68] that M-Diam is NP-complete when the dataset is in 2-dimensional Euclidean space and Euclidean distance is used. It is also shown that for a general dissimilarity function, even the n-approximation problem is NP-complete for all $n \geq 1$.

For a dataset in 1-dimensional Euclidean space M-Diam is solvable in polynomial time. Further, whenever the dissimilarity measure is a metric it is possible to find an approximate solution efficiently in general. Brücker [22] provides an algorithm for finding solutions with a max-diameter value within two times the max-diameter value of the optimal solution. However, Bern and Eppstein [13] show that, under the Euclidean distance, the 1.969-approximation problem associated with M-Diam is NP-complete. It is also shown that, for the similar Max-Radius Clustering problem, the 1.822-approximation problem is NP-complete.

Another result of Brücker [22] is that Sum-of-Diameters Clustering (S-Diam) is NP-complete for $k \geq 3$. Doddi et al. [45] show that the associated 2-approximation problem can be solved efficiently when the dissimilarity used is a metric. But if the triangle inequality is not satisfied by the dissimilarity used then it is shown that, if $P \neq NP$, no efficient approximation algorithm is possible, even for $k = 3$.

In Hansen and Jaumard [74] it is shown that, when $k = 2$, S-Diam is solvable in $O(n^2 \log n)$ time. It is also shown that minimising any function of the diameters can be done in $O(n^5)$ time when $k = 2$.

In Garey et al. [64] it is shown that the *quantization problem* is NP-complete. It turns out that this problem is a special case of Sum-of-Stars Clustering (S-Stars) where a dissimilarity with values in $\{0,1\}$ is used. Further, it is shown that the problem is

NP-complete even when the parameter corresponding to k is set to 2.

The above complexity results, especially the last one for S-STARS, has led many authors to believe that CENTROID-DISTANCE CLUSTERING in Euclidean space is also NP-complete. In fact this was proved only relatively recently. It is in fact NP-complete even when $k = 2$ [3] and for general k in 2-dimensional Euclidean space [106]. If both k and the number of dimensions, m, are fixed, then the problem is exactly solvable in $O(n^{mk+1} \log n)$ time [84].

We conclude by remarking that not all clustering problems are hard. One example is MIN-SPLIT CLUSTERING for which a polynomial time algorithm exists (for the optimisation problem). In particular, the problem is solved with the SLINK hierarchical clustering algorithm of Johnson [86]. The runtime of this algorithm is $O(n^2)$ [40]. However, the min-split criterion does suffer from the previously mentioned chaining effect and for this reason it is often paired with a criterion for homogeneity in a bicriterion, effectively making it a hard problem again [40].

2.4.3 Methods

The consequence of the complexity issues discussed in the previous section are that heuristic methods for approximating optimal solutions are very prevalent. There are only a small handful of algorithms which are either guaranteed to find a globally optimal solution or guaranteed to find a solution within a known degree of the optimal.

Many of the heuristic methods happen in two stages. Given an n element dataset, D, the number of clusters desired, $k \in \{1, \dots, n\}$, and a criterion, $f \colon \mathcal{P}_D \to \mathbb{R}^+$, we proceed as follows:

1. (Initialisation) Select some initial partition, $\mathcal{C} \in \mathcal{P}_D$,

2. (Improvement) Try to improve the partition with respect to the criterion, so find a second partition $\mathcal{C}'$ such that $f(\mathcal{C}') < f(\mathcal{C})$.

We will look at each stage in turn now.

Initialisation

A common way to select an initial partition, $\mathcal{C}$, is to choose k elements to be initial estimates for the "centre" of each cluster. The elements of the dataset are then grouped with the centre that is closest. Sometimes the centres are updated each time a new element is grouped with them, for example they could become the actual centroid of the current cluster. The k centres are usually selected from the dataset, but they could also be taken from the underlying metric space.

A great number of methods for selecting centres have been suggested and we present a far from exhaustive review next. Further examples can be found in the following [78], [93], [24], [168], [26], [51] and [131].

The most simple ways to select k centres include taking the first k elements of the dataset [105], taking k random elements of the dataset [58] or taking k elements regularly spaced across the dataset [11].

More sophisticated methods include the Kennard-Stone algorithm (shown in Algorithm 1) which aims to find k centres that are maximally separated in the metric space. It is noted by De Groot et al. [35] that this method is sensitive to outliers and it is suggested that, instead of picking maximally separated elements as the first two centres as in the original algorithm (see Algorithm 1), the first two centres should be distant, but not outliers.

Algorithm 1 Kennard-Stone initial centres algorithm.

```
Input: Number of centres desired, $k \geq 2$, and a dataset, $D = \{x_1, x_2, \ldots, x_n\}$ with $n \geq k$,
   with dissimilarity $d\colon D \times D \to \mathbb{R}^{\geq 0}$.
Output: Centres $\{c_1, c_2, \ldots, c_k\} \subseteq D$.
  $(c_1, c_2) \leftarrow \operatorname*{arg\,max}_{(c,c') \in D \times D} d(c, c')$          ▷ First two centres
  $S \leftarrow \{c_1, c_2\}$
  $m \leftarrow 2$
  while $m < k$ do
      $c_{m+1} \leftarrow \operatorname*{arg\,max}_{c \in D \setminus S} \min_{c' \in S} d(c, c')$
      $S \leftarrow S \cup \{c_{m+1}\}$
      $m \leftarrow m + 1$
  end while
  return $S$.
```

Another method, due to Yuan et al. [170], is shown in Algorithm 2. The algorithm has a parameter, α, for which the authors suggest a value of 0.75 since this gave good results in their experiments.

Algorithm 2 Yuan-Meng-Zhang-Dong initial centres algorithm.

```
**Input:** Number of centres desired, $k > 0$, $0 < \alpha \leq 1$, and a dataset $D = \{x_1, x_2, \dots, x_k\}$,
  with $n \geq k$, embedded in a metric space $(M, d)$.
**Output:** Centres $\{c_1, c_2, \dots, c_k\} \subseteq D$.
  $m \leftarrow 1$
  while $m < k$ do
      $(x_i, x_j) \leftarrow \operatorname*{arg\,min}_{(x,x') \in D \times D} d(x, x')$
      $C_m \leftarrow \{x_i, x_j\}$
      $D \leftarrow D \setminus \{x_i, x_j\}$
      while $|C_m| < \alpha \cdot \frac{n}{k}$ do
          $x_i \leftarrow \operatorname*{arg\,min}_{x \in D} \min_{x' \in C_m} d(x, x')$
          $C_m \leftarrow C_m \cup \{x_i\}$
          $D \leftarrow D \setminus \{x_i\}$
      end while
  end while
  for all $1 \leq i \leq k$ do
      $c_i \leftarrow \operatorname*{arg\,min}_{c \in M} \sum_{x \in C_i} d^2(x, c)$
  end for
  return $\{c_1, c_2, \dots, c_k\}$.
```

A further method, introduced by Arthur and Vassilvitskii [6] is called k-means++ and is shown in Algorithm 3. Its name reflects the fact that it was designed to be used in conjunction with an improvement method that is often referred to as k-means.

Improvement

Given an initial partition, $\mathcal{C}$, of a set D and a criterion $f\colon \mathcal{P}_D \to \mathbb{R}^+$, the aim is now to find a partition that is an *improvement* of the initial partition, if possible. A partition $\mathcal{C}'$ of D is an improvement of $\mathcal{C}$ if $f(\mathcal{C}') < f(\mathcal{C})$ or $f(\mathcal{C}') > f(\mathcal{C})$, whichever is appropriate for the criterion. We will now look at some of the methods that have been devised for making improvements.

Algorithm 3 k-means++ initial centres algorithm.

```
Input: Number of centres desired, $k > 0$, dataset $D = \{x_1, x_2, \dots, x_n\}$, with $n \geq k$, and
  dissimilarity $d\colon D \times D \to \mathbb{R}^{\geq 0}$.
Output: Centres $\{c_1, c_2, \dots, c_k\}$.
  $S \leftarrow \{\text{element chosen at random from } D\}$
  $m \leftarrow 2$
  while $m < k$ do
    $S \leftarrow S \cup \left\{ \text{an element } x' \in D \text{ chosen with probability } \dfrac{\min_{c \in S} d^2(x', c)}{\sum_{x \in D} \min_{c \in S} d^2(x, c)} \right\}$
  end while
  return $S$.
```

Hartigan's method [77] is a simple heuristic that is general in the sense that it takes a criterion as a parameter and attempts to produce an improvement with respect to that criterion. This is unlike most methods which are specialised to a particular criterion and often even to a particular type of dataset and metric. Many of the earliest clustering methods, such as those described in [53], simply consisted of a particular initialisation step followed by Hartigan's method with respect to a particular criterion.

Given a set D, a partition $\mathcal{C}$ on D and a criterion, Hartigan's method produces a new partition, $\mathcal{C}'$, by selecting an element in D and moving it to a different cluster, if such a move would produce an improvement. The new cluster is chosen such that the criterion is optimised at each stage. This is called an optimal reassignment. Elements are repeatedly optimally reassigned until no more reassignments would produce an improvement. Thus, upon termination, a locally optimal partition has been found.

Lloyd's method, which is specialised to the centroid-distance criterion with the Euclidean distance, is probably the most well-known and widely used clustering method today. The centroid-distance problem is also commonly called the k-means problem. The popularity of Lloyd's method had led to many authors referring to it simply as "the k-means algorithm" (although, as we note below, calling it an algorithm may be erroneous). There is some confusion here, though, as some authors also refer to Hartigan's method with respect to the centroid-distance criterion as k-means and others refer Lloyd's method as H-means.

Lloyd's method is simple and easy to understand, given an initial k partition of a dataset D it proceeds as follows:

1. Call the current centroid of each cluster the "centre". Move each element to the cluster with the closest centre.

2. Set each cluster's centre equal to the centroid. If any centres changed then go to step 1, otherwise terminate.

The version written here is as suggested by Ball and Hall [9]. MacQueen [105] prefers a variation which recomputes centroids every time new elements are added to the clusters, instead of only once during each iteration. To enable quicker termination, usually another stopping condition is added, for example a maximum number of iterations or a minimum threshold for the changes made after each iteration.

Note also that we are careful not to call this method an algorithm. It is possible that step 1 will results in empty clusters, which has two problems: we no longer have a partition, and centroids are not defined for empty sets. Some modifications have been suggested to ensure that clusters remain nonempty, for example in [124].

The major advantage of Lloyd's method is that it is simple, easy to implement (assuming centroids are easy to compute) and it generally converges very quickly to a solution. There are some drawbacks, however. As mentioned already, it is implicitly assumed that centroids are easy to compute. This is the case when Euclidean distance is used, or in fact any Bregman divergence, since centroids are then equal to the mean of the elements [156, 10]. Even calculating means presents a drawback, since it requires that an implementation has access to the dataset while other methods only require a matrix of dissimilarities. A final drawback is that the method actually requires, in the worse case, exponentially many iterations to converge, even in the Euclidean plane [158].

But despite these drawbacks, Lloyd's method remains very popular. In a brief survey of some popular data mining and machine learning software it was found that Lloyd's method is used in around three-quarters of implementations for centroid-distance clustering.

Hartigan's method is used the rest of the time including in some of the more prominent software, for example in R and Weka. A comparison of Lloyd's method and Hartigan's method, in which Hartigan's method is shown to produce more attractive partitions in fewer iterations, is given in [156].

We turn now to sum-of-stars clustering, which is also known as k-medoids clustering analogous to k-means clustering and reflecting the fact that medoids are calculated in the process of calculating the sum-of-stars criterion. The oldest and simplest heuristic method for sum-of-stars clustering is Partitioning Around Medoids (PAM), shown in Algorithm 4, was introduced by Kaufman and Rousseeuw [90].

Algorithm 4 Partition Around Medoids (PAM).

Input: A dataset, $D = \{x_1, \ldots, x_n\}$ with a dissimilarity $d\colon D \times D \to \mathbb{R}^{\geq 0}$ and an integer $k > n$.

Output: A k partition $\{C_1, \ldots, C_k\}$ of D.

- Initialise a partition $\{C_1, \ldots, C_k\}$ by selecting k elements at random from D and placing one in each cluster. Call the set of k selected elements D_s and the set of unselected elements D_u.
- Assign each element in D_u to the cluster containing a maximally similar element of D_s, according to d. Calculate the sum-of-stars value for this partition.
- Repeatedly swap elements of D_s with elements of D_u and reassign the elements of D_u, whenever the swap would improve the sum-of-stars value. Continue to swap until no swap will improve the sum-of-stars.

Since PAM can require the sum-of-stars value to be calculated a very large number of times, Kaufman and Rousseeuw [90] suggested a version with enhanced performance for large datasets called CLARA (CLustering LArge Applications). For a dataset, D, the basic idea is to run PAM on a subset, D', of the D which is selected at random. The centres found by PAM in D' should be a good approximation for the centres that would be found by PAM in D. After the centres are found in D', the remaining elements of D are assigned to the cluster with the closest centre. Usually multiple samples are taken for D' and the partition with the smallest sum-of-stars value is taken. As a guidance, Kaufman and Rousseeuw [90] suggest taking five samples of size $40 + 2k$.

Compared with PAM, CLARA performs well for larger datasets. For an n element dataset where k clusters are desired, each iteration of PAM has runtime complexity of $O(k(n-k)^2)$ while for CLARA it is only $O(k(40+k)^2 + k(n-k))$ with the suggested sample size [122].

CLARANS (Clustering Large Applications based on RANdomized Search) takes influence from both PAM and CLARA [122]. The algorithm considers the graph G_{nk} with vertex set $V_{nk} = \{\{m_1, m_2, \ldots, m_k\} | m_1, m_2, \ldots, m_k \in D\}$, so all possible k-medoids, which each define a partition. Two vertices, $M_1, M_2 \in V_{nk}$ are connected by an edge if and only if $|M_1 \cap M_2| = k-1$, meaning M_1 and M_2 differ by only one object.

With regards to this graph, PAM can be thought of as a search through G_{nk} to find a vertex for which no adjacent vertices have a smaller sum-of-stars value. For a found vertex in G_{nk}, PAM works by considering every vertex adjacent to it to continue the search. CLARA instead searches through only a subgraph of G_{nk} which has far fewer vertices, so effectively, when a vertex is found in G_{nk}, it considers only a subset of the vertices adjacent to it.

CLARANS, shown in Algorithm 5, searches through the entire graph G_{nk} but, for a found vertex $v \in V_{nk}$, does not consider every vertex adjacent to v, instead it considers only a random subset of the set of vertices adjacent to v. In experiments, CLARANS was shown to run faster than PAM while producing partitions with a smaller sum-of-stars value than those produced by CLARA [122].

Heuristics remain the most popular choice for clustering applications in general, probably due to their ability to produce acceptable solutions for a wide variety of input. But it is often more desirable to have an algorithm with a proven worst-case runtime complexity and which is guaranteed to produce either optimal solutions or solutions within some known degree of the optimal. The former type are called *exact algorithms* and the latter *approximation algorithms*.

Approximation algorithms are a popular way to deal with many NP-hard optimisation problems. An algorithm is called an n-approximation algorithm if the solutions produced

Algorithm 5 CLARANS.

```
Input: A dataset, $D = \{x_1, \dots, x_n\}$, with a dissimilarity $d\colon D \times D \to \mathbb{R}^{\geq 0}$, an integer $k \geq n$
    and parameters $numlocal$ and $maxneighbour$.
Output: A set of $k$ medoids $\{m_1, \dots, m_k\} \subseteq D$.
    • Set $i \leftarrow 1$ and $mincost$ to a large number,
    • Set $current$ to a random vertex of $G_{nk}$,
    • Set $j \leftarrow 1$,
    • Compare the sum-of-stars of $current$ and a random vertex, $S$, adjacent to $current$,
      If $S$ has a lower cost, set $current \leftarrow S$ and go to 3,
      Otherwise, set $j \leftarrow j + 1$,
    • If $j \leq maxneighbour$, go to 4,
      Otherwise, if $ss(current) < mincost$, set $mincost \leftarrow ss(current)$ and
      $bestnode \leftarrow current$,
    • Set $i \leftarrow i + 1$m
      If $i \leq numlocal$, go to 2,
      Otherwise return $bestnode$.
```

are guaranteed to have a cost, with respect to the objective criterion function, of no more than n times (a minimisation problem) or at least $1/n$ times (a maximisation problem) the cost of the optimal solution, for some $n > 1$.

Gonzalez [68] provides a 2-approximation algorithm for max-diameter which, for an n element dataset and k cluster desired, has worst-case runtime complexity of $O(nk)$. The algorithm requires that the dissimilarity used is a metric. Doddi et al. [45] provide a 2-approximate algorithm for sum-of-diameters which, again, requires that the dissimilarity is a metric. They also provide a version which produces partitions of no more than $O(k)$ clusters that have sum-of-diameters values within $O(\ln(n/k))$ times the optimal for a k-partition. Other approximation algorithms have been devised for max-radius [166] and centroid-distance [30].

Some work has been done on exact algorithms for certain criteria. For those problems which are NP-complete, these algorithms are only efficient under very constrained input conditions. Algorithms for centroid-distance exist including methods using branch-and-bound [19], branch-and-cut [3] and column generation [115]. Some of these algorithms are able to find exact solutions to instances in the plane with up to 2000 objects, in reasonable time [3].

The all-squares problem, or more generally sum-of-cliques, has received attention also. An exact algorithm using branch-and-bound [94] can be used to solve problems with $n \leq 50$, $k \leq 5$ in reasonable time [75]. Cutting planes have also been applied with problems with $n \leq 158$ solved quickly [75, 125].

In Hansen and Delattre [73] an exact algorithm using graph-colouring for the max-diameter problem is given which is shown to exactly cluster 270 objects in reasonable time.

Not many clustering problems are in P, but one problem which has already been mentioned is min-split which can be exactly solved using the SLINK algorithm [86, 148]. This is actually a hierarchical clustering algorithm, but can be used to find partitions simply by taking the layer of the hierarchy with the desired number of clusters.

Chapter 3

Sum-of-Squares Clustering

This chapter is largely based on the following paper:

G. Kettleborough and V.J. Rayward-Smith. Optimising sum-of-squares measures for clustering multisets defined over a metric space. *Discrete Applied Mathematics*, 161(16-17): 2499–2513, 2013. doi: 10.1016/j.dam.2013.04.015

∼

My contributions include: the examples showing that equations (3.4) and (3.5) do not hold for all metrics, the example to show that linear separability does not hold for all-squares clustering in Section 3.2.2, the proof for Lemma 1, the proof of Theorems 5, 6 and 10, the examples of using the assignment metric in Section 3.4.2, and the examples leading to Theorem 14. I was also responsible for preparation of the published manuscript and implementation of the assignment metric for testing.

3.1 Introduction

3.1.1 Summary

In this chapter we examine two sum-of-squares criteria for clustering in a metric space and compare their performance. It has recently been shown that partitional clustering under the centroid-distance criterion is an NP-hard problem in Euclidean space. We show that the associated decision problem is NP-complete even in a highly constrained 2-valued metric space, as well as general p-valued metric spaces. We also show that the problem is NP-complete under the related all-squares criterion both in Euclidean space and for a p-valued metric. We propose a new metric for comparing clustering called the assignment metric which allows us to use information about the underlying metric space of a clustering to help distinguish different clustering solutions. Using this metric we finally show that optimal clustering solutions under our two different sum-of-squares criteria can be as different as any two solutions can possibly be.

3.1.2 Multiset datasets

The first stage of clustering is to build a dataset in which clusters are to be found. These data are often sampled from some set, M. If the objects in M have a metric, d, defined on them then we say that (M, d) is a metric space. The same metric is, of course, then defined for any elements sampled from M, so given a dataset, $D \subseteq M$, (D, d) is itself a metric space.

However, there is usually no restriction on how many times an element from M may be sampled so, contrary to the name, a dataset is not usually a set, but is often a multiset. A multiset can be considered a generalisation of a fuzzy set, that is an ordered pair (D, μ_D) where D is the *underlying set* and is the *membership function* but with $\mu_D\colon D \to \mathbb{R}^+$. For some element $x \in D$ $\mu_D(x) = n$ means that n copies x appear in (D, μ_D) (note that $x \in D$ is ordinary set notation).

3.1.3 Multiset clusterings

A clustering is usually considered to be a set of subsets of the dataset but if the dataset is a multiset then the clusters must also be multisets. A k-clustering of (D, μ_D) is therefore $\mathcal{C} = \{(C_1, \mu_1), (C_2, \mu_2), \ldots, (C_k, \mu_k)\}$ where μ_i is the membership function for cluster C_i. For any such clustering we have that $C_1 \cup C_2 \cup \cdots \cup C_k = D$ and for all $x \in D$, $\sum_{i=1}^{k} \mu_i(x) = \mu_D(x)$. We will see later that there are some surprising differences between normal clusterings and multiset clusterings (see Theorem 8).

3.2 Clustering criteria

As discussed in Section 2.4.1, the problem of clustering is finding a solution where the clusters are both homogeneous, meaning elements belonging to the same cluster are similar, and well-separated, meaning elements belonging to different clusters are dissimilar. We have seen that there are a large number of possible k-clustering solutions for any given dataset and many criteria for judging the homogeneity, separation or both of a given solution.

In this chapter we will look at the two sum-of-squares criteria which judge the suitability of a k-clustering based on both homogeneity and separation. We derived the two criteria, called centroid-distance and all-squares, from the scatter matrix in Section 2.4.1. We state them again here defined as cost functions along with definitions of their corresponding optimal partitions.

The *all-squares cost* of a clustering is defined as:

$$cost_{as}(\mathcal{C}) = \sum_{i=1}^{k} \sum_{x,y \in C_i} \mu_i(x)\mu_i(y)d^2(x,y). \tag{3.1}$$

Definition 1. *A multiset k-clustering which minimises the all-squares cost for a particular D is called an all-squares optimal k-clustering of D.*

The *centroid-distance cost* is defined as:

$$cost_{cd}(\mathcal{C}) = \sum_{i=1}^{k} \sum_{x \in C_i} \mu_i(x) d^2(x, c_i), \tag{3.2}$$

where c_i is the centroid of C_i and is defined as

$$c_i = \arg\min_{c \in M} \sum_{x \in C_i} \mu_i(x) d^2(x, c).$$

Calculation of the centroid depends on the metric; for example, with the Euclidean metric the centroid is the mean, with the overlap metric it is the mode and with the heterogeneous Euclidean-overlap metric (HEOM, as defined in Section 2.2.3) it is a mixture of means and modes.

Definition 2. *A multiset k-clustering which minimises the centroid-distance cost for a particular D is called a centroid-distance optimal k-clustering of D.*

Centroid-distance cost is a well known criterion which is often called sum-of-squares without qualification in the literature (see, for example, [4, 115, 150, 85, 75]). We believe that both of the criteria which we have stated can equally well be called sum-of-squares criteria, so we qualify them as all-squares and centroid-distance.

If d is the Euclidean metric, centroid-distance is also equivalent to a criterion for separation, although it is not immediately obvious why. It is due to the parallel axis theorem, or Huygens-Steiner theorem, which we saw in Section 2.4.1. The theorem states that when (M, d_E) is the Euclidean space with the Euclidean metric

$$\sum_{x \in A} \mu_A(x) d_E^2(x, s) = \sum_{x \in A} \mu_A(x) d_E^2(a, s) + \sum_{x \in A} \mu_A(x) d_E^2(x, a) \tag{3.3}$$

for each $A \subseteq M$, where $a \in M$ is the centroid of A, and $s \in M$ [150]. By summing over each clustering, $C_1, \ldots, C_k$ in some clustering of D we obtain the equation:

$$\sum_{x \in D} \mu_D(x) d_E^2(x, s) = \sum_{i=1}^{k} \sum_{x \in C_i} \mu_i(x) d_E^2(c_i, s) + \sum_{i=1}^{k} \sum_{x \in C_i} \mu_i(x) d_E^2(x, c_i). \tag{3.4}$$

The left-hand side is a constant for any s and D and the second term on the right-hand side is the centroid-distance cost. Therefore, minimising the centroid-distance cost is equivalent to maximising squared-separation

$$sep(\mathcal{C}) = \sum_{i=1}^{k} \sum_{x \in C_i} \mu_i(x) d_E^2(c_i, s).$$

For convenience, s is usually taken to be the centroid of D.

We can also use equation (3.3) to show an alternative formula for centroid-distance cost. We substitute some $y \in C_i$ for s in (3.3) and sum over all $y \in C_i$, multiplying each term by $\mu_i(y)$:

$$\begin{aligned} \sum_{y \in C_i} \sum_{x \in C_i} \mu_i(x) \mu_i(y) d_E^2(x, y) &= \sum_{y \in C_i} \mu_i(y) d_E^2(c_i, y) \sum_{x \in C_i} \mu_i(x) \\ &+ \sum_{x \in C_i} \mu_i(x) d_E^2(c_i, x) \sum_{y \in C_i} \mu_i(y). \end{aligned}$$

Rearranging we get

$$\frac{1}{2} \frac{\sum\limits_{x, y \in C_i} \mu_i(x) \mu_i(y) d_E^2(x, y)}{\sum\limits_{x \in C_i} \mu_i(x)} = \sum_{x \in C_i} \mu_i(x) d_E^2(x, c_i).$$

So,

$$\frac{1}{2} \sum_{i=1}^{k} \frac{\sum\limits_{x, y \in C_i} \mu_i(x) \mu_i(y) d_E^2(x, y)}{\sum\limits_{x \in C_i} \mu_i(x)} = \sum_{i=1}^{k} \sum_{x \in C_i} \mu_i(x) d_E^2(x, c_i). \tag{3.5}$$

The right-hand side here is centroid-distance cost, while the left-hand side bears a resemblance to all-squares cost.

It is important to note that equation (3.3) does not hold for a general metric space, so neither equation (3.4) nor equation (3.5) necessarily hold for metrics other than the Euclidean metric.

An example to show that (3.4) does not hold for the HEOM follows. Let D be a dataset with two numerical attributes and one categorical attribute. The dataset consists of four distinct elements, $a = (0, 0, p), b = (1, 0, q), c = (0, 1, r), d = (1, 1, s)$ with the following multiplicities, shown in multiset notation:

$$D = \{a, a, a, a, b, b, b, c, c, d\}.$$

Under the HEOM, the centroid-distance optimal multiset 2-clustering is

$$\{\{a, a, a, a, c, c\}, \{b, b, b, d\}\},$$

which has centroids $(0, \frac{1}{3}, p)$ and $(1, \frac{1}{4}, q)$. The centroid-distance cost is $4(\frac{1}{3})^2 + 2((\frac{2}{3})^2 + 1) + 3(\frac{1}{4})^2 + (\frac{3}{4})^2 + 1 = 5\frac{1}{12}$ and the squared-separation is $6((\frac{2}{5})^2 + (\frac{1}{30})^2) + 4((\frac{3}{5})^2 + (\frac{1}{20})^2 + 1) = 6\frac{5}{12}$. But this is not the maximum squared-separation possible since the clustering

$$\{\{a, a, a, a\}, \{b, b, b, c, c, d\}\},$$

with centroids $(0, 0, p)$ and $(\frac{2}{3}, \frac{1}{2}, q)$, has a squared-separation of $4((\frac{2}{5})^2 + (\frac{3}{10})^2) + 6((\frac{4}{15})^2 + (\frac{1}{5})^2 + 1) = 7\frac{2}{3}$.

Similarly, we can show that (3.5) does not hold with the overlap metric: let $M = (D, d_1) = (\{a, b, c\}, overlap)$. We measure the cost of the multiset clustering $\{C_1\}$ where $C_1 = (\{a, b, c\}, \mu_1)$ and $\mu_1(a) = \mu_1(b) = \mu_1(c) = 1$. The centroid is equal to either a, b or c, this gives a cost using the left-hand side of equation (3.5) of $\frac{1}{2}$ while the right-hand side gives 2.

So, although it may seem sensible to minimise centroid-distance cost for other metrics, in general it should not be considered to be a criterion for separation.

3.2.1 Consistency

Definition 3. *A clustering on a multiset is called consistent if and only if it satisfies the condition that for all $x \in D, \mu_i(x) = \mu_D(x)$ whenever $x \in C_i$. In other words, all of the copies of x belong to the same cluster.*

Theorem 2. *There exists an all-squares optimal k-clustering that is consistent.*

Proof. Assume there exists an all-squares optimal k-clustering

$$\mathcal{C} = \{(C_1, \mu_1), (C_2, \mu_2), \dots, (C_k, \mu_k)\}$$

where two identical points are in different clusters: $x \in C_i$ and $x \in C_j$ with $\mu_i(x) \geq 1$ and $\mu_j(x) \geq 1$.

Consider the clustering $\mathcal{C}'$ constructed from $\mathcal{C}$ by removing one copy of x from C_i and placing it in C_j. Then

$$cost_{as}(\mathcal{C}') = cost_{as}(\mathcal{C}) - \sum_{y \in C_i} \mu_i(y) d^2(x, y) + \sum_{y \in C_j} \mu_j(y) d^2(x, y).$$

Since $cost_{as}(\mathcal{C})$ is minimal we thus deduce

$$\sum_{y \in C_j} \mu_j(y) d^2(x, y) \geq \sum_{y \in C_i} \mu_i(y) d^2(x, y). \tag{3.6}$$

Similarly we can construct a clustering $\mathcal{C}''$ by removing one copy of x from C_j and placing it in C_i and deduce that

$$cost_{as}(\mathcal{C}'') = cost_{as}(\mathcal{C}) - \sum_{y \in C_j} \mu_j(y) d^2(x, y) + \sum_{y \in C_i} \mu_i(y) d^2(x, y).$$

So

$$\sum_{y \in C_i} \mu_i(y) d^2(x, y) \geq \sum_{y \in C_j} \mu_j(y) d^2(x, y). \tag{3.7}$$

Therefore due to (3.6) and (3.7)

$$\sum_{y \in C_i} \mu_i(y) d^2(x, y) = \sum_{y \in C_j} \mu_j(y) d^2(x, y),$$

and so

$$cost_{as}(\mathcal{C}) = cost_{as}(\mathcal{C}') = cost_{as}(\mathcal{C}'').$$

So, for an optimal clustering, moving elements from one cluster where they exist to another cluster where they exist does not change the all-squares cost. Thus all copies of an element can be safely moved to the same cluster and the result follows. □

Theorem 3. *There exists a centroid-distance optimal k-clustering which is consistent.*

Proof. Assume there exists a centroid-distance optimal k-clustering

$$\mathcal{C} = \{(C_1, \mu_1), (C_2, \mu_2), \ldots, (C_k, \mu_k)\},$$

where two identical points are in different clusters: $x \in C_i$ and $x \in C_j$ with $\mu_i(x) \geq 1$ and $\mu_j(x) \geq 1$.

The clustering $\mathcal{C}'$ is constructed from $\mathcal{C}$ by removing one copy of x from C_i and placing it in C_j. We call the new clusters C_i' and C_j'. Let the centroids of C_i, C_j, C_i' and C_j' be c_i, c_j, c_i' and c_j' respectively with membership functions μ_i, μ_j, μ_i' and μ_j'.

Note that C_i' must be nonempty in order for c_i' to be well defined. This is always the case since the clustering where $C_i = \{x\}$ is trivially a suboptimal centroid-distance clustering.

Then

$$\begin{aligned} cost_{cd}(\mathcal{C}') = cost_{cd}(\mathcal{C}) &- \sum_{y \in C_i} \mu_i(y) d^2(y, c_i) - \sum_{y \in C_j} \mu_j(y) d^2(y, c_j) \\ &+ \sum_{y \in C_i'} \mu_i'(y) d^2(y, c_i') + \sum_{y \in C_j'} \mu_j'(y) d^2(y, c_j'). \end{aligned}$$

Now, by the definition of a centroid,

$$\sum_{y\in C'_j} \mu'_j(y)d^2(y,c'_j) \le \sum_{y\in C_j} \mu_j(y)d^2(y,c_j) + \mu_i(x_1)d^2(x,c_j),$$

and

$$\sum_{y\in C'_i} \mu'_i(y)d^2(y,c'_i) \le \sum_{y\in C_i} \mu_i(y)d^2(y,c_i) - \mu_i(x_1)d^2(x,c_i).$$

So

$$cost_{cd}(\mathcal{C}') \le cost_{cd}(\mathcal{C}) + \mu_i(x_1)(d^2(x,c_j) - d^2(x,c_i)).$$

Since $cost_{cd}(\mathcal{C})$ is optimal we deduce that

$$d^2(x,c_j) \ge d^2(x,c_i).$$

Similarly we construct a clustering $\mathcal{C}''$ by removing x_2 from C_j and placing it in C_i and deduce that

$$d^2(x,c_i) \ge d^2(x,c_j).$$

Therefore

$$d^2(x,c_i) = d^2(x,c_j)$$

and

$$cost_{cd}(\mathcal{C}) = cost_{cd}(\mathcal{C}') = cost_{cd}(\mathcal{C}'').$$

So, for an optimal clustering, moving elements from one cluster where they exist to another cluster where they exist does not change the centroid-distance cost and, again, all copies of an element can be moved to the same cluster and the result follows. □

3.2.2 Linear separability

If our dataset is a subset of n dimensional Euclidean space, $\mathbb{R}^n$, then we can consider whether a clustering solution will be linearly separable or not. Consider a clustering

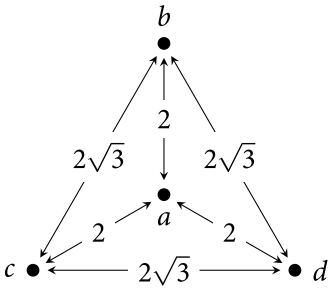


Figure 3.1: A dataset in Euclidean space consisting of three points arranged in an equilateral triangle with another point at the centre.

$\mathcal{C} = \{C_1, C_2, \ldots, C_k\}$ of points in $\mathbb{R}^n$. Let H_i denote the convex hull of C_i for all $1 \geq i \geq k$.

Definition 4. *$\mathcal{C}$ is linearly separable if $H_i \cap H_j = \varnothing$ for all $1 \leq i < j \leq k$.*

It is well known that a centroid-distance optimal clustering is necessarily linearly separable. This can be seen by considering the Voronoi tessellation defined by the centroids of the clustering; each cluster must lie wholly within the region defined by the tessellation.

An all-squares optimal clustering, on the other hand, is not necessarily linearly separable. This can be shown by a simple example. Consider a dataset consisting of distinct points a, b, c, d arranged in the Euclidean plane as illustrated in Figure 3.1. There are n copies of a and one copy each of b, c, d. The clustering $\mathcal{C} = \{\{(a,n)\}, \{(b,1),(c,1),(d,1)\}\}$ has an all-squares cost of 72. But if we move one of b, c, d into the first cluster, then we have a cost of $8n + 24$ which is greater whenever $n > 6$. Similarly if we move two of b, c, d into the first cluster we get a cost of $16n + 24$. So whenever $n > 6$ the all-squares optimal clustering is $\mathcal{C}$ and hence is not linearly separable.

3.3 Complexity issues

We will now formally define the problems of finding an optimal clustering according to either criteria and analyse the complexity of these problems.

3.3.1 All-squares clustering

We state the all-squares problem formally as a decision problem:

All-Squares Clustering (ASC)

Instance: A multiset of nodes (D, μ_D), where $D = \{S_1, S_2, \ldots, S_n\}$; a metric, d, which is defined for all elements in D; the number of clusters desired, $k \in \mathbb{Z}^+$ and a bound $B \in \mathbb{R}^+$.
Question: Is there a k-clustering, $\mathcal{C} = \{C_1, C_2, \ldots, C_k\}$, such that

$$cost_{as}(\mathcal{C}) = \sum_{i=1}^{k} \sum_{x,y \in C_i} \mu_{C_i}(x) \mu_{C_i}(y) d^2(x, y) \leq B \quad ?$$

The problem is defined for multisets but, unless stated otherwise, the NP-completeness proofs which follow are for sets, since this establishes the result for multisets too. When dealing with sets, the membership functions are omitted since they are always equal to 1.

Euclidean space

An important special case of ASC is when the nodes are in Euclidean space and we use the Euclidean metric d_E. We call this special case Euclidean all-squares clustering (EASC).

Theorem 4. *EASC is NP-complete.*

Proof. We observe that a guessed solution, $\mathcal{C}_g$, can be checked in polynomial time by simply calculating $cost_{as}(\mathcal{C}_g)$ and comparing the value to the bound. Therefore EASC $\in$ NP.

Now to show that EASC is NP-complete we construct a transformation from a known NP-complete problem to our problem. The NP-complete problem we will use is the following [62]:

Partition into Triangles (PT)

Instance: Graph $G = (V, E)$, with $|V| = 3q$ for some integer q.

Question: Can each of the vertices of G be partitioned into q disjoint sets $V_1, V_2, \ldots, V_q$, each containing exactly 3 vertices, such that for each $V_i = \{u_i, v_i, w_i\}$, $1 \leq i \leq q$, all three of the edges $\{u_i, v_i\}$, $\{u_i, w_i\}$ and $\{v_i, w_i\}$ belong to E?

We transform an instance $I = G = (V, E)$ of PT, where $|V| = 3p$, $V = \{v_0, \ldots, v_{3p-1}\}$ and $|E| = m$ into an instance $f(I)$ of EASC: first we assign to D a set of $n = 3p$ points in $(3p + m)$-dimensional Euclidean space. Each element $v_i \in V$ corresponds to a point where

1. the ith coordinate is $N = 2m$,
2. all other coordinates $1 \leq i \leq 3p$ are 0,
3. for $1 \leq j \leq m$, if v_i is incident with $e_j \in E$ then the $(3p + j)$th coordinate is 1, otherwise it is 0.

We then set k to p and B to $8m - 12p + 12N^2p$. It is easy to see that this transformation can be computed in polynomial time.

Lemma 1. *Let D be a dataset with $|D| = n$ where the distance squared between each pair of distinct elements is s. The sum-of-squares minimal k-clustering will contain clusters with cardinality of either $\lceil \frac{n}{k} \rceil$ or $\lfloor \frac{n}{k} \rfloor$ only.*

Proof. The cost associated with a cluster, C, which contains p elements is $p(p-1)s = (p^2 - p)s$. The extra cost of adding an element to C is $((p+1)^2 - (p+1))s - (p^2 - p)s = 2ps$, and the saving when removing an element is $(p^2 - p)s - ((p-1)^2 - (p-1))s = 2(p-1)s$.

We proceed by contradiction. Assume that we have an optimal k-clustering, where $k \geq 2$ (the case where $k = 1$ is trivial), including two clusters C_i and C_j, where $|C_i| = p_i$, $|C_j| = p_j$ and $p_i > p_j + 1$. We move an element from C_i to C_j; the difference in overall cost due to the move is $2p_js - 2p_is + 2s < 0$, so we have a saving. Therefore our original clustering could not have been optimal. □

Corollary 1. *Since $p_i \geq p_j + 2$, the difference in cost will be $(2p_j - 2p_i + 2)s \leq 2s$. Therefore, the cost of any suboptimal clustering will be at least $2s$ greater than the cost of the optimal clustering.*

In our reduction $\lceil \frac{n}{k} \rceil = \lfloor \frac{n}{k} \rfloor = 3$, so the optimal clustering when all nonzero distances are equal will be one where each cluster contains three elements.

Lemma 2. *I is a YES instance of PT if and only if $f(I)$ is a YES instance of EASC.*

Proof. If $I \in Y_{\mathrm{PT}}$ then we can construct a clustering by letting each cluster correspond to one of the triangles in the partition of G. Let w, x, y be the vertices of such a triangle, and therefore a cluster. The distance squared between a pair of these points is $d^2(w,x) = 2N^2 + \deg(w) + \deg(x) - 2$; the $2N^2$ comes from the coordinates governed by rules 1 and 2 in the transformation and the $\deg(w) + \deg(x) - 2$ comes from the coordinates governed by rule 3. The overall cost of the clustering is therefore $2(2\deg(w) + 2\deg(x) + 2\deg(y) + 6N^2 - 6)$. Hence, the set of p triangles has cost

$$4\sum_{v \in V} \deg(v) + 12N^2p - 12p = 8m + 12N^2p - 12p = B, \tag{3.8}$$

so $f(I) \in Y_{\mathrm{EASC}}$.

If $f(I) \in Y_{\mathrm{EASC}}$ then we have a k-clustering which fits the bound. The distance between each pair of distinct points in the dataset is at least $2N^2$. Assume that the distances are all exactly $2N^2$: due to Lemma 1, the optimal clustering in this case is one where each cluster contains 3 elements, and this clustering has an overall cost of $12N^2p$. Due to Corollary 1, any suboptimal clustering would have an overall cost of at least $(12p + 4)N^2$ and, since $N^2 = 4m^2$, will not meet the bound. Clearly, if one of these suboptimal clusterings contained distances greater than $2N^2$ in the sum their overall cost would be greater still. Therefore, each cluster must contain exactly 3 elements.

As shown in equation (3.8), the cost of a clustering where each cluster corresponds to a triangle in G equals the bound. If some cluster does not correspond to a triangle, then its cost is increased, and therefore the overall cost of the clustering will be greater than the bound. So each cluster must be a triangle, so $I \in Y_{\mathrm{PT}}$. □

Hence, due to Lemma 2, the theorem is established. □

p-valued metric

Definition 5. *A p-**valued metric** is a metric where the cardinality of the codomain is equal to some positive integer p.*

Theorem 5. *ASC is NP-complete even with a 3-valued metric.*

Proof. We observe that a guessed solution can be checked in polynomial time, therefore ASC $\in$ NP.

Now we choose again to construct a transformation from Partition into Triangles (PT). An instance I of PT is transformed into an instance $f(I)$ of ASC as follows: first we construct a 3-valued metric space by setting D to V and defining a function $d\colon D \times D \to \{0, \alpha, \beta\}$ where $0 > \alpha > \beta$ by

$$d(u,v) = \begin{cases} 0 & \text{if } u = v, \\ \alpha & \text{if there exists an edge in } G \text{ between } u \text{ and } v, \\ \beta & \text{otherwise.} \end{cases}$$

Lemma 3. *(D, d) is a metric space.*

Proof. We show that d satisfies each condition required for a metric for all $u, v, w \in D$:

1. $d(u,v) = 0$ if and only if $u = v$ by definition,

2. $d(u,v) = d(v,u)$ by definition,

3. $d(u,v) + d(v,w) \geq d(u,w)$ (triangle inequality)

   If $u = w$ then this is trivially true.

If $d(u,w) = \alpha$ then $u \neq w$ so either $u \neq v$ or $v \neq w$ or both.

If $d(u,w) = \beta$ then $u \neq w$ and there is no edge between u and w, we then have two cases: either both $u \neq v$ and $v \neq w$ which satisfies the inequality, or one of $u = v$ or $v = w$, but we know there is no edge between u and w so therefore there is no edge between v and w or v and u respectively, so the inequality is satisfied.

□

We then set B to $6q\alpha$ and k to q to complete the transformation. It is easy to see that this transformation can be computed in polynomial time.

Lemma 4. *I is a YES instance of PT if and only if $f(I)$ is a YES instance of ASC.*

Proof. If $I \in Y_{\mathrm{PT}}$ then observe that we can construct a clustering, $\mathcal{C}$, by assigning to each cluster C_i the set V_i. Since each V_i has an edge between each pair of vertices, the cost of each cluster C_i is 6α, and therefore the cost of $\mathcal{C}$ is $6q\alpha$. So therefore $f(I) \in Y_{\mathrm{ASC}}$.

If $I \in N_{\mathrm{PT}}$ then we cannot have a clustering where all clusters have cardinality 3 as this will contain at least one distance greater than α and therefore not meet the bound. We must consider clusterings where all within cluster distances are equal to α, but due to Lemma 1, any such clustering with clusters of different sizes always has a higher cost so they also cannot meet the bound. Therefore $f(I) \in N_{\mathrm{ASC}}$. □

Hence, due to Lemmata 3 and 4, the theorem is established.

□

Theorem 6. *For all $n \geq 3$ there exists a metric such that ASC is hard, therefore ASC is NP-complete with an n-valued metric when $n \geq 3$.*

Proof. We proceed by induction. Theorem 5 establishes the base case, so we now assume that the problem is NP-complete with an n-valued metric and show that it is NP-complete with an $(n+1)$-valued metric.

We transform an instance, I, of the n-valued problem into an instance, $f(I)$ of the $(n+1)$-valued problem. Let D^*, d^*, k^* and B^* be the dataset, metric, number of clusters and bound of $f(I)$, respectively. We set D^* to $D \cup \{a\}$, k^* to $k+1$ and B^* to B. We define a function $d^*\colon D \times D$ by

$$d^*(x,y) = \begin{cases} 0 & \text{if } x = y, \\ s & \text{if } x = a \text{ or } y = a, \\ d(x,y) & \text{otherwise,} \end{cases}$$

where $s = \sum_{p,q \in D} d^2(p,q)$. Notice that the distance between a and each element in D is the complete sum of all distances squared in D. It is easy to see that this transformation can be computed in polynomial time.

Lemma 5. *I is a YES instance of n-valued ASC if and only if $f(I)$ is a YES instance of $(n+1)$-valued ASC.*

Proof. If $I \in Y_{ASC}$ then there is some clustering $\mathcal{C} = \{C_1, C_2, \ldots, C_k\}$ of D with a cost less than or equal to B. We can construct a similar clustering of $D^* : \mathcal{C}^* = \mathcal{C} \cup \{\{a\}\}$. The extra cluster will not contribute to the cost, so the cost of $\mathcal{C}^*$ is also less than or equal to B, so $f(I) \in Y_{ASC}$.

If $f(I) \in Y_{ASC}$ then there is some clustering, $\mathcal{C}^* = \{C_1^*, C_2^*, \ldots, C_{k^*}^*\}$, with cost less than or equal to B. Let C_l^* be the cluster which contains the element a. If some other elements belong to this cluster then we can move them into any other cluster: each element we move makes a saving of at least s^2 but the final overall cost of the cluster we move them to can be at worst s, so clearly this will result in a saving. Now, C_l contains only a so does not contribute to the overall cost, so $\mathcal{C}^* \setminus C_l^*$ has an overall cost less than or equal to B and therefore $I \in Y_{ASC}$. □

Hence, due to Lemma 5, the theorem is established. □

If our dataset is a set and the metric is 2-valued then the problem is solvable in polynomial time due to Lemma 1. We need only solve the equation

$$x\left\lceil\frac{n}{k}\right\rceil + (k-x)\left\lfloor\frac{n}{k}\right\rfloor = n$$

for x and an optimal clustering is then $x = n - k\lfloor\frac{n}{k}\rfloor$ clusters of $\lceil\frac{n}{k}\rceil$ elements and $k - x$ clusters of $\lfloor\frac{n}{k}\rfloor$ elements. For the decision version we would then simply calculate the cost of the clustering to see if it is less than the bound. We conclude with:

Theorem 7. *When the dataset is a set with a 2-valued metric, an optimal clustering can be found in polynomial time.*

However, if the dataset is a multiset this is not the case; the problem becomes NP-complete.

Theorem 8. *When the dataset is a multiset with a 2-valued metric, ASC is an NP-complete problem.*

Proof. We construct a transformation from the NP-complete problem [62]:

Minimum Sum-of-Squares (MSS)

Instance: Finite set A, a size $s(a) \in \mathbb{Z}^+$ for each $a \in A$, positive integers $K \leq |A|$ and J.

Question: Can A be partitioned into K disjoint sets $A_1, A_2, \ldots, A_k$ such that

$$\sum_{i=1}^{K}\left(\sum_{a\in A_i} s(a)\right)^2 \leq J \quad ?$$

We construct our dataset by setting D to A and μ_D to s and we set k to K. We define a metric d for all $u, v \in D$ as

$$d(u,v) = \begin{cases} 0 & \text{if } u = v, \\ 1 & \text{otherwise.} \end{cases}$$

To find the value for B, consider an optimal multiset clustering $\mathcal{C} = \{C_1, \ldots, C_k\}$ which is consistent, so $\mu_{C_i}(x) = \mu_D(x)$ for all $x \in D$ and $1 \leq i \leq k$. The cost of a single cluster, C_i, is

$$\sum_{x,y \in C_i} \mu_D(x)\mu_D(y) = \left(\sum_{x \in C_i} \mu_D(x) \right)^2 - \sum_{x \in C_i} (\mu_D(x))^2,$$

so the total cost of $\mathcal{C}$ is

$$\sum_{i=1}^{k} \left(\sum_{x \in C_i} \mu_D(x) \right)^2 - \sum_{i=1}^{k} \sum_{x \in C_i} (\mu_D(x))^2. \tag{3.9}$$

The second term is a constant and is equivalent to

$$\Gamma = \sum_{x \in D} (\mu_D(x))^2.$$

We set B to $J - \Gamma$ and the transformation is complete. It is now easy to see that the cost of the optimal clustering, $\mathcal{C}$, will meet the bound, B, if and only if the first term in expression (3.9) is less than or equal to J. So $I \in Y_{\mathrm{MSS}}$ if and only if $f(I) \in Y_{\mathrm{ASC}}$ and the theorem is established. □

3.3.2 Centroid-distance clustering

Again, we state the problem formally as a decision problem:

Centroid-Distance Clustering (CDC)

Instance: A metric space (M, d), a set of nodes $D \subseteq M$, the number of clusters desired, $k \in \mathbb{Z}^+$, and a bound, $B \in \mathbb{R}^+$.

Question: Is there a k-clustering, $\mathcal{C} = \{C_1, C_2, \ldots, C_k\}$ such that

$$cost_{cd}(\mathcal{C}) = \sum_{i=1}^{k} \sum_{x \in C_i} \mu_i(x) d^2(x, c_i) \leq B,$$

where $c_i \in M$ is the centroid of cluster C_i?

Again, the problem is defined for multisets, but the following proof is for sets so the membership function has been omitted.

Theorem 9. *CDC is NP-complete.*

Proof. We observe that a guessed solution, $\mathcal{C}_g$, can be checked in polynomial time, therefore CDC $\in$ NP.

To show that CDC is NP-complete we will construct a transformation from a known NP-complete problem to our problem. The problem we will use is the following [62]:

Dominating Set (DS)

Instance: A graph $G = (V, E)$ and a positive integer $K \leq |V|$.
Question: Is there a dominating set of size K or less for G or, in other words, a subset $V' \subseteq V$ with $|V'| \leq K$ such that for all $u \in V \setminus V'$ there is a $v \in V'$ for which $\{u, v\} \in E$?

We transform an instance I of DS into an instance $f(I)$ of CDC. Fist we construct a 3-valued metric space by setting $M = D$ to V and define a function $d \colon M \times M \to \mathbb{R}$ by

$$d(u,v) = \begin{cases} 0 & \text{if } u = v, \\ 1 & \text{if there exists an edge in } G \text{ between } u \text{ and } v, \\ 2 & \text{otherwise.} \end{cases}$$

This is a metric space by Lemma 3. We then set k to K and B to $n - k$.

Lemma 6. *I is a YES instance of DS if and only if $f(I)$ is a YES instance of CDC.*

Proof. If $I \in Y_{DS}$ then $|V'| \leq K = k$. We can construct a k-clustering by first picking $k - |V'|$ arbitrary elements from $V \setminus V'$ and adding each of these to separate clusters. These are final clusters and will not contribute to the overall cost. We then add one element of the dominating set each to the remaining $|V'|$ clusters. These elements are the tentative centroids of the clusters. So far we still have an overall cost of zero. The remaining $n - k$

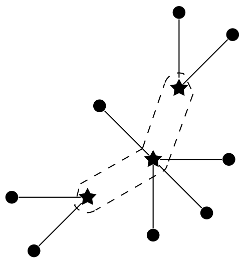


Figure 3.2: Each of the k clusters correspond to star graphs with the centroid at the centre, so there is a dominating set of size k as outlined.

elements are added one by one to any cluster where they share an edge with the tentative centroid. Thus, each of these $n - k$ elements contributes to the cost by 1. If one of the final centroids of these constructed clusters turns out to be different to the tentative centroids, this can only decrease the cost of that cluster, by the definition of a centroid. Therefore the overall cost is less than or equal to $B = n - k$, so $f(I) \in Y_{CDC}$.

If $f(I) \in Y_{\text{CDC}}$ then we have an overall cost of less than or equal to $n - k$. Since $D = M$, each cluster must contain an element equal to the centroid of that cluster, so there are exactly k elements which do not contribute to the overall cost. Each of the $n - k$ remaining elements must contribute to the cost by at least 1, so therefore must contribute by exactly 1 giving an overall cost of exactly $n - k$. Each cluster therefore corresponds to a star in G, as illustrated in Figure 3.2, so $I \in Y_{\text{DS}}$. □

Hence, due to Lemmata 3 and 6, the theorem is established. □

Theorem 9 has already been established using Euclidean space. In fact, the problem has been shown to be NP-hard for both $k = 2$ in general Euclidean space [4] and for general k in only 2 dimensions [106]. If both k and d, the number of dimensions, are fixed, the problem is exactly solvable in $O(n^{dk+1} \log n)$ time[84].

Our proof establishes the following new result:

Corollary 2. *Even if the metric, d, is a 3-valued metric, centroid-distance clustering remains an NP-complete problem.*

Theorem 10. *For all $n \geq 3$ there exists a metric such that CDC is hard, therefore CDC is NP-complete with an n-valued metric when $n \geq 3$.*

Proof. We proceed by induction. Theorem 9 establishes the base case, so we now assume that the problem is NP-complete with an n-valued metric and show that it is NP-complete with an $(n+1)$-valued metric.

We use the same transformation as used for the proof of Theorem 6.

Lemma 7. *I is a YES instance of n-valued CDC if and only if $f(I)$ is a YES instance of $(n+1)$-valued CDC.*

Proof. If $I \in Y_{CDC}$ then there is some clustering $\mathcal{C} = \{C_1, C_2, \ldots, C_k\}$ of D with a cost less than or equal to B. We can construct a similar clustering of D^*: $\mathcal{C}^* = \mathcal{C} \cup \{\{a\}\}$. The extra cluster will not contribute to the cost, so the cost of $\mathcal{C}^*$ is also less than or equal to B, so $f(I) \in Y_{CDC}$.

If $f(I) \in Y_{CDC}$ then there is some clustering $\mathcal{C}^* = \{C_1^*, C_2^*, \ldots, C_{k^*}^*\}$ which has cost less than or equal to B. Let C_l^* be the cluster which contains element a. If other elements belong to this cluster then the cluster will have cost greater than or equal to s^2. We can move all other elements to any other cluster, say C_p^*, which will reduce the cost of C_l^* to 0, but the cost of C_p^* must be less than s, so we have an overall saving. Now C_l^* does not contribute to the overall cost, so $\mathcal{C}^* \setminus C_l^*$ has an overall cost less than or equal to B and therefore $I \in Y_{CDC}$. □

Hence, due to Lemma 7, the theorem is established. □

However, when the metric is 2-valued, the problem is solvable in polynomial time, even in the multiset case. To see this, let our metric be

$$d(u,v) = \begin{cases} 0 & \text{if } u = v, \\ s & \text{otherwise.} \end{cases}$$

The overall cost of a clustering will be

$$\sum_{i=1}^{k} \sum_{\{x \in C_i, x \neq c_i\}} \mu_i(x) s^2$$

or, equivalently,

$$\sum_{x \in D} \mu_D(x) s^2 - \sum_{i=1}^{k} \mu_i(c_i) s^2.$$

The first term is a constant, so the problem is to simply maximise the second term. This can be done by ordering the elements of D by their membership count in descending order and selecting the first k elements. We move all copies of each of the selected elements to their own cluster; these will become the centroids. The remaining elements may then be moved to an arbitrary cluster as they will all contribute to the cost by s each in any case.

3.4 The assignment metric

There are many existing methods for comparing clusterings. Many do not provide us with, or do not have established, bounds so are not very useful for our purposes. We will look at two metrics which have upper bounds: the Variation of Information (VI) (see Section 2.3.2) and our assignment metric which we present here. The assignment metric is based on set matching like those discussed in Section 2.3.2, but allows any metric to be used for comparing the matched sets.

Let $\mathcal{P}_k$ be the set of all possible k-clusterings of D. We define the assignment metric as a function $\Delta\colon \mathcal{P}_k \times \mathcal{P}_k \to \mathbb{R}$ where

$$\Delta(\mathcal{C}_1, \mathcal{C}_2) = \min_{\sigma \in S_k} \sum_{i=1}^{k} \delta(C_{1i}, C_{2\sigma(i)})$$

for some $\delta\colon (2^D \setminus \varnothing) \times (2^D \setminus \varnothing) \to \mathbb{R}$ and where S_k is the set of all possible functions $\sigma\colon \{1, \ldots, k\} \to \{1, \ldots, k\}$.

Theorem 11. *The measure Δ is a metric on $\mathcal{P}_k$ whenever δ is a metric on $(2^D \setminus \varnothing)$.*

Proof. We show that Δ satisfies all conditions required for a metric:

1. $\Delta(\mathcal{C}_1, \mathcal{C}_2) \geq 0$ trivially since $\delta(C_{1i}, C_{2j}) \geq 0$ for all $1 \leq i, j \leq k$ since δ is itself a metric,

2. If $\mathcal{C}_1 = \mathcal{C}_2$ then there exists some σ for which $C_{1i} = C_{2\sigma(i)}$ so $\delta(C_{1i}, C_{2\sigma(i)}) = 0$ for all $1 \leq i \leq k$, so $\Delta(\mathcal{C}_1, \mathcal{C}_2) = 0$.

   If $\Delta(\mathcal{C}_1, \mathcal{C}_2) = 0$ then $\delta(C_{1i}, C_{2\sigma(i)}) = 0$ and therefore $C_{1i} = C_{2\sigma(i)}$ for some σ and all $1 \leq i \leq k$, so $\mathcal{C}_1 = \mathcal{C}_2$,

3. $\Delta(\mathcal{C}_1, \mathcal{C}_2) = \Delta(\mathcal{C}_2, \mathcal{C}_1)$ trivially since $\delta(C_{1i}, C_{2j}) = \delta(C_{2j}, C_{1i})$ for all $1 \leq i, j \leq k$,

4. Let $\Delta(\mathcal{C}_1, \mathcal{C}_2) = \sum_{i=1}^{k} \delta(C_{1i}, C_{2\sigma(i)})$ for some $\sigma \in S_k$
   and $\Delta(\mathcal{C}_2, \mathcal{C}_3) = \sum_{i=1}^{k} \delta(C_{2i}, C_{\tau(i)})$ for some $\tau \in S_k$.

   Then,

$$\begin{aligned}
\Delta(\mathcal{C}_1, \mathcal{C}_2) + \Delta(\mathcal{C}_2, \mathcal{C}_3) &= \sum_{i=1}^{k} \delta(C_{1i}, C_{2\sigma(i)}) + \delta(C_{2\sigma(i)}, C_{3\tau(\sigma(i))}) \\
&\geq \sum_{i=1}^{k} \delta(C_{1i}, C_{3\tau(\sigma(i))}) \quad \text{(due to triangle inequality of } \delta) \\
&\geq \min_{\sigma \in S_k} \sum_{i=1}^{k} \delta(C_{1i}, C_{3\sigma(i)}) \\
&= \Delta(\mathcal{C}_1, \mathcal{C}_3).
\end{aligned}$$

□

Possible choices for the δ metric are the cardinality of the symmetric difference, $\delta(A, B) = |A \triangle B|$, and the normalised symmetric difference, also known as the Jaccard distance, $\delta(A, B) = \frac{|A \triangle B|}{|A \cup B|}$. These are well known metrics with the former being used often in the literature for comparing sets, for example in [134]. These two metrics extend naturally to multisets; the multiset version of symmetric difference is $\delta((A, \mu_A), (B, \mu_B)) = \sum_{x,y \in A \cup B} |\mu_A(x) - \mu_B(y)|$.

Calculating Δ amounts to calculating the minimum cost matching between the clusters. There are $k!$ possible matchings, but the minimum can be found in $O(k^3)$ time using the

Hungarian algorithm [97]. Since $k! < k^3$ when $k < 6$, it may be more efficient to simply enumerate all solutions when k is small.

Clusterings with different k can be compared by the addition of the pseudometric $||\mathcal{C}_1| - |\mathcal{C}_2||$ (ie. the absolute difference between the set cardinalities). Let $\mathcal{P}$ be the set of all partitions of D. The assignment metric then becomes a function $\Delta\colon \mathcal{P} \times \mathcal{P} \to \mathbb{R}$ defined as

$$\Delta(\mathcal{C}_1, \mathcal{C}_2) = \min_{\sigma \in S_k} \sum_{i=1}^{k} \delta(C_{1i}, C_{2\sigma(i)}) + \lambda ||\mathcal{C}_1| - |\mathcal{C}_2||,$$

where $k = \min(|C_1|, |C_2|)$, $\delta\colon (2^D \setminus \varnothing) \times (2^D \setminus \varnothing) \to \mathbb{R}$ and λ is a positive real number.

This is most sensible when the metric δ is bounded or normalised and λ is greater than or equal to the upper bound on δ. Calculation of the metric using the Hungarian algorithm can then be performed by a simple modification, namely by setting the cost of matching a set to nothing as λ and finding the minimum matching in the same way.

Using a bounded metric does not limit our choice since any metric can be bounded. One general formula for bounding a metric by $[0, 1]$ is

$$\delta_b(A, B) = \frac{\delta(A, B)}{1 + \delta(A, B)}. \tag{3.10}$$

Alternatively, it may be possible to normalise the metric, as with the normalised symmetric difference metric.

This version of the assignment metric is bounded by $\lambda \cdot \max(|\mathcal{C}_1|, |\mathcal{C}_2|)$, and this bound is approached arbitrarily closely by $\mathcal{C}_1 = \{\{1, 2, \ldots, n\}\}$. $\mathcal{C}_2 = \{\{1\}, \{2\}, \ldots, \{n\}\}$ in the limit of large n for any metric δ. Tighter bounds may exist for specific choices of δ and fixed k (we prove one in Section 3.4.3).

In Section 3.4.1 we show how the assignment metric can be used to compare fuzzy partitions and in Section 3.4.2 we discuss the use of metrics which are aware of the underlying metric space of a clustering. In Section 3.5 we use the symmetric difference to prove a worst case result for our two clustering criteria.

3.4.1 Comparing fuzzy partitions

The assignment metric extends easily to fuzzy partitions. All we need is a metric on fuzzy sets. Here we present such a metric which is analogous to the symmetric difference.

Let $\mathcal{F}_f$ be the set of all fuzzy sets and $\mathcal{F}_c \subset \mathcal{F}_f$ the set of all crisp sets. Note that a crisp set is a special case of a fuzzy set where

$$\mu_A(x) = \begin{cases} 1 & \text{if } x \in A \\ 0 & \text{if } x \notin A \end{cases}$$

for all $A \in \mathcal{F}_c$.

We now define a function $\delta_f \colon \mathcal{F}_f \times \mathcal{F}_f \to \mathbb{R}$ by

$$\delta_f(A, B) = \sum_{x \in \mathcal{R}} |\mu(x, A) - \mu(x, B)|.$$

Theorem 12. *The measure δ_f is equivalent to the symmetric difference $\delta(A, B) = |A \triangle B|$ for all $A, B \in \mathcal{F}_c$.*

Proof. For each $x \in \mathcal{R}$:

1. If $x \notin A$ and $x \notin B$ then $\mu(x, A) = \mu(x, B) = 0$ and therefore this does not contribute to the sum in $\delta_f(A, B)$,

2. if $x \in A$ and $x \notin B$ then $\mu(x, A) = 1$ and $\mu(x, B) = 0$ so $|\mu(x, A) - \mu(x, B)| = |1 - 0| = 1$,

3. similarly, if $x \notin A$ and $x \in B$ then $|\mu(x, A) - \mu(x, B)| = |0 - 1| = 1$,

4. if $x \in A$ and $x \in B$ then $|\mu(x, A) - \mu(x, B)| = |1 - 1| = 0$ and therefore also does not contribute to the sum.

□

For fuzzy sets this measure does correspond to $|A \triangle B| = |A \cup B| - |A \cap B|$ since

$$\mu(x, A \cup B) = \max(\mu(x, A), \mu(x, B))$$
$$\mu(x, A \cap B) = \min(\mu(x, A), \mu(x, B))$$

so the cardinality $|A \cup B| - |A \cap B|$ is

$$\begin{aligned}&\sum_{x \in \mathcal{R}} \big(\max(\mu(x, A), \mu(x, B)) - \min(\mu(x, A), \mu(x, B))\big)\\ &= \sum_{x \in \mathcal{R}} |\mu(x, A) - \mu(x, B)|.\end{aligned}$$

Theorem 13. *$(\mathcal{F}_f, \delta_f)$ is a metric space.*

Proof.

1. $\delta_f(A, B) = 0 \iff A = B$:

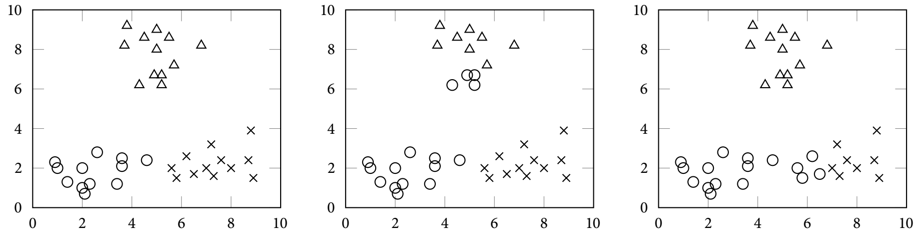
$$\begin{aligned}\delta_f(A, B) = 0 &\iff \mu(x, A) = \mu(x, B) \quad \forall x \in \mathcal{R}\\ &\iff A = B,\end{aligned}$$

2. $\delta_f(A, B) = \delta_f(B, A)$ by definition,

3. $\delta_f(A, B) + \delta_f(B, C) \geq \delta_f(A, C)$ (the triangle inequality):

$$\begin{aligned}\delta_f(A, B) + \delta_f(B, C) &= \sum_{x \in \mathcal{R}} |\mu(x, A) - \mu(x, B)| + \sum_{x \in \mathcal{R}} |\mu(x, B) - \mu(x, C)|\\ &= \sum_{x \in \mathcal{R}} \big(|\mu(x, A) - \mu(x, B)| + |\mu(x, B) - \mu(x, C)|\big)\\ &\geq \sum_{x \in \mathcal{R}} |\mu(x, A) - \mu(x, C)|\\ &= \delta_f(A, C)\end{aligned}$$

□



(i) Clustering $\mathcal{C}_1$. (ii) Clustering $\mathcal{C}_2$. (iii) Clustering $\mathcal{C}_3$.

Figure 3.3: Three clusterings on the same dataset.

There is also a corresponding normalised version of this metric for use with the second version of the assignment metric:

$$\delta_{f_n}(A, B) = \frac{|A \triangle B|}{|A \cup B|}.$$

3.4.2 Lifting the underlying metric space

Another possibility that the assignment metric gives us is to use a metric which is aware of the metric space underlying our clustering. This overcomes some of the limitations that are present in most comparison methods [8].

Figure 3.3 shows three possible clusterings, $\mathcal{C}_1, \mathcal{C}_2$ and $\mathcal{C}_3$ on a given dataset. The elements in this dataset exist in the Euclidean plane and are shown in their relative positions on the page. Imagine that $\mathcal{C}_1$ represents the standard clustering and $\mathcal{C}_2$ and $\mathcal{C}_3$ are two alternative clusterings. We would like to know which of the alternative clusterings is closest to the standard so we measure the distance between $\mathcal{C}_1$ and $\mathcal{C}_2$ and $\mathcal{C}_1$ and $\mathcal{C}_3$. Under the VI metric we get $\Delta(\mathcal{C}_1, \mathcal{C}_2) = \Delta(\mathcal{C}_1, \mathcal{C}_3) = 3.1154556$. Under the assignment metric with the symmetric difference we similarly get $\Delta(\mathcal{C}_1, \mathcal{C}_2) = \Delta(\mathcal{C}_1, \mathcal{C}_3) = 72$. This seems contrary to our intuition: we would expect that $\mathcal{C}_2$ is further from the standard since the clusters are of very different shapes. The assignment metric combined with the Hausdorff metric δ_H and Euclidean metric d_E reflects this intuition: we get $\Delta(\mathcal{C}_1, \mathcal{C}_2) = 6.0621233$ and $\Delta(\mathcal{C}_1, \mathcal{C}_3) = 3.424846$. We could, of course, have used a metric other than the Euclidean metric to plug in to the Hausdorff metric if this made more sense.

The Hausdorff metric can be normalised in a natural way for use with the second version of the assignment metric. One way would be to divide by the maximum distance observed in the dataset:

$$\delta_{H_n}(X, Y) = \frac{\delta_H(X, Y)}{\max_{x,y \in D} d(x, y)}.$$

When we use a metric like the Hausdorff metric we have three "layers" of metric space that we are dealing with. The underlying layer is our dataset and metric used for clustering (D, d). We have the cluster layer $(2^D \setminus \varnothing, \delta)$ and the clustering layer $(\mathcal{P}, \Delta)$. When we take into account all three layers we say that we are "lifting" the underlying metric space to the space of partitions.

It should be noted that two commonly used functions for comparing sets in a metric space

$$f_1(X, Y) = \min_{x \in X, y \in Y} d(x, y)$$

and

$$f_2(X, Y) = \sum_{x \in X} \sum_{y \in Y} d(x, y)$$

are *not* metrics. This can be shown by considering a simple counterexample for each: $f_1(\{x, y\}, \{x, z\}) = 0$ violates property 3 of a metric (identity of indiscernibles) and $f_2(\{x, y\}, \{x, y\}) > 0$ violates property 2 (identical elements are most similar).

3.4.3 Upper bound

An upper bound for Δ can be established when $\delta(A, B) = |A \triangle B|$. We show the upper bound using sets for clarity, but the result is still valid for multisets using the multiset version of symmetric difference.

There are k^2 possible matchings between clusters, the costs of which can be shown in a matrix:

$$\begin{pmatrix} |C_{11} \triangle C_{21}| & |C_{12} \triangle C_{21}| & \dots & |C_{1k} \triangle C_{21}| \\ |C_{11} \triangle C_{22}| & |C_{12} \triangle C_{22}| & \dots & |C_{1k} \triangle C_{22}| \\ \vdots & \vdots & \ddots & \vdots \\ |C_{11} \triangle C_{2k}| & |C_{12} \triangle C_{2k}| & \dots & |C_{1k} \triangle C_{2k}| \end{pmatrix}.$$

Let $p_i = |C_{1i}|$, $p_j = |C_{2j}|$ and $p_{ij} = |C_{1i} \cap C_{2j}|$. We can calculate the sum of the values in the jth row of the matrix for some j:

$$\begin{aligned} \sum_{i=1}^{k} |C_{1i} \triangle C_{2j}| &= \sum_{i=1}^{k} (|C_{1i} \cup C_{2j}| - |C_{1i} \cap C_{2j}|) \\ &= \sum_{i=1}^{k} (p_j + p_i - 2p_{ij}). \end{aligned}$$

Noting that $\sum_{i=1}^{k} p_i = m$ and $\sum_{i=1}^{k} p_{ij} = p_j$ this can be written as

$$kp_j + m - 2p_j = (k-2)p_j + m.$$

The total sum of values in the matrix is therefore

$$\sum_{j=1}^{k} ((k-2)p_j + m),$$

and noting that $\sum_{j=1}^{k} p_j = m$ we get

$$(k-2)m + mk.$$

There are $k!$ possible solutions to the assignment problem where each solution is a combination of k assignments. Let $S = \{s_1, s_2, \dots, s_{k!}\}$ be the set of costs for each solution so each element is equal to $\sum_{i=1}^{k} \delta(C_{1i}, C_{2\sigma(i)})$ for some σ. The mean value in the matrix is

$$\frac{(k-2)m + mk}{k^2},$$

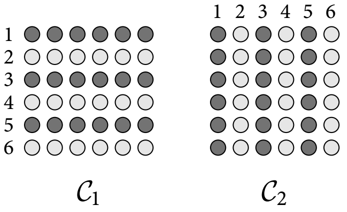


Figure 3.4: Two clusterings, $\mathcal{C}_1$ and $\mathcal{C}_2$, formed of six clusters each. These clusterings are optimally different under both the assignment metric and variation of information.

so the mean solution value is

$$\begin{aligned}\bar{s} &= k\left(\frac{(k-2)m + mk}{k^2}\right)\\ &= 2m - \frac{2m}{k}.\end{aligned}$$

We can split S into three disjoint subsets $S_<$, $S_=$ and $S_>$ which contain the solutions which are less than, equal to and greater than the mean, respectively. We have two cases: either both $S_< = \varnothing$ and $S_> = \varnothing$ or both $S_< \neq \varnothing$ and $S_> \neq \varnothing$. In the first case, all solutions are equal to the mean and therefore any of these will be picked; in the second case, a solution from $S_<$ will be picked.

Therefore

$$\Delta(\mathcal{C}_1, \mathcal{C}_2) \leq 2m - \left\lceil \frac{2m}{k} \right\rceil.$$

This bound is tight and can be met with the worst case which is illustrated in Figure 3.4:

$$\begin{aligned}\mathcal{C}_1 &= \{\{1, \ldots, k\}, \{k+1, \ldots, 2k\}, \ldots, \{(k-1)k+1, \ldots, k^2\}\}\\ \mathcal{C}_2 &= \{\{1, k+1, \ldots, (k-1)k+1\},\\ &\qquad \{2, k+2, \ldots, (k-1)k+2\},\\ &\qquad \ldots,\\ &\qquad \{k, k+k, \ldots, (k-1)k+k\}\}.\end{aligned}$$

This also produces the worst case under the VI metric, as shown in [113].

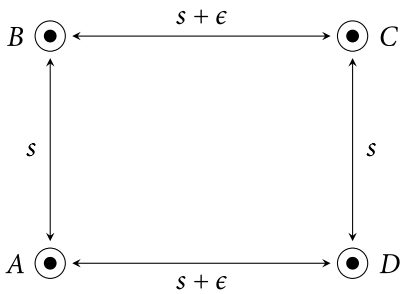


Figure 3.5: Relative positions of elements to be clustered. A and B contain N elements each, C and D contain $N + 1$ elements each.

3.5 Worst case performance

We have so far seen that our two clustering criteria have a number of similarities, and a number of differences. Using our metrics for comparing clusterings, we will now show just how different an all-squares optimal clustering and a centroid-distance optimal clustering of the same dataset can be.

Let (M, d_E) be 2-dimensional Euclidean space with the Euclidean metric and our dataset be (D, μ_D) where $D \subset \mathbb{R}^2$, $D = A \cup B \cup C \cup D$, $\mu_D(A) = \mu_D(B) = N$ and $\mu_D(C) = \mu_D(D) = N + 1$. The relative positions of A, B, C, D in the Euclidean plane are shown in Figure 3.5. Two possible 2-clusterings of (D, μ_D) are $\{A \cup B, C \cup D\}$ and $\{A \cup D, B \cup C\}$ which we will call $\mathcal{C}_1$ and $\mathcal{C}_2$ respectively.

Formulae for the all-squares and centroid-distance costs of $\mathcal{C}_1$ and $\mathcal{C}_2$ can now be given, first for centroid-distance cost:

$$\begin{aligned} cost_{cd}(\mathcal{C}_1) &= \frac{1}{2}\left(Ns^2 + (N+1)s^2\right), \\ cost_{cd}(\mathcal{C}_2) &= \frac{1}{2}\left(N(s+\epsilon)^2 + (N+1)(s+\epsilon)^2\right). \end{aligned}$$

It is clear that $\mathcal{C}_1$ is the optimal clustering under the centroid-distance criterion whenever

Table 3.1: The costs of possible 2-clusterings of D, with minimum costs underlined.

Clustering	Centroid-distance cost	All-squares cost
$\{A \cup B, C \cup D\}$	$\underline{2.940}$	19.60
$\{A \cup D, B \cup C\}$	3.375	$\underline{18.00}$
$\{A \cup C, B \cup D\}$	5.613	33.68
$\{A, B \cup C \cup D\}$	4.152	41.52
$\{B, A \cup C \cup D\}$	4.152	41.52
$\{C, A \cup B \cup D\}$	4.283	29.76
$\{D, A \cup B \cup C\}$	4.283	29.76

$\epsilon > 0$. Now, for all-squares cost,

$$\begin{aligned} cost_{as}(\mathcal{C}_1) &= 2N^2s^2 + 2(N+1)^2s^2, \\ cost_{as}(\mathcal{C}_2) &= 4N(N+1)(s+\epsilon)^2. \end{aligned}$$

So $\mathcal{C}_2$ is the optimal clustering under the all-squares criterion whenever

$$\epsilon < \sqrt{s^2 + \frac{s^2}{2N(N+1)}} - s.$$

Other clusterings of D are possible, but these are trivially more expensive.

A simple numerical example can now be constructed; let $s = 1.4$, $\epsilon = 0.1$ and $N = 1$. The costs of all possible 2-clusterings are shown in Table 3.1. Again we see that the optimal clustering under each criterion is different.

$\mathcal{C}_1$ and $\mathcal{C}_2$ are not just slightly different clusterings, they are in fact optimally different, that is, no two clusterings can be more different, according to both the VI metric and assignment metric, as well as our basic intuition. This leads us to:

Theorem 14. *A centroid-distance optimal clustering and an all-squares optimal clustering can be optimally different under both the VI metric and the assignment metric.*

3.6 Conclusion

In this chapter we have shown that the two sum-of-squares criteria, centroid-distance and all-squares, share some similarities but also some differences. Optimal clusterings according to both criteria may be consistent but, while centroid-distance always produces linearly separable solutions, all-squares does not.

Both criteria simultaneously measure both homogeneity and separation. For all-squares, the relationship between the homogeneity measure and separation measure is trivial and independent of the choice of metric. However, for centroid-distance we have shown that the homogeneity measure is not necessarily equivalent to the separation measure when using something other than the Euclidean metric. The example we used is the homogeneous Euclidean-overlap metric for mixed data.

It has recently been shown that the centroid-distance problem is NP-hard using Euclidean space. We have shown that both problems are NP-complete even when using a simple 3-valued metric. We also show that all-squares is NP-complete in Euclidean space. When using a 2-valued metric, both problems are in P, except for all-squares on a multiset, which remains NP-complete.

We have introduced a new metric for comparing clusterings, called the assignment metric. It is, in fact, a family of metrics since any metric for comparing matched sets can be used. This allows for some interesting choices of metric, namely we can use it to compare fuzzy clusterings and take into account the underlying metric space of the dataset which gives the measure a more intuitive feel. We have used this metric to show just how different optimal clusterings according to the two criteria can be. It turns out that they can be optimally different, according to both our metric and the VI metric.

Much of the work here which we present for multisets also applies naturally to fuzzy multisets. The consistency results also extend to crispness, that is to say that optimal fuzzy clusterings according to either criteria need not have been fuzzy in the first place. The assignment metric also applies to fuzzy clusterings simply by using a metric for fuzzy sets.

Chapter 4

Hierarchical Clustering

4.1 Summary

In this chapter we review the relevant terminology and background that is required for the remainder of this thesis. We begin with the general theory of graphs, trees and their relationship to distances and special types of metrics. We then look at the problem of reconstructing trees from distances and finally at the theory of "lassoing" a tree from incomplete distance information.

4.2 Graphs, Trees and Distances

4.2.1 History

Trees have been used to represent hierarchical structures for many hundreds of years [96]. One of the most well-known and ubiquitous occurrences is that of the family tree. The use of a tree to represent lineages was widespread in Europe by the 14th Century in Christian artwork depicting the ancestors of Jesus of Nazareth. This depiction is known as the Tree of Jesse [29].

Darwin would later popularise the concept of the more general "tree of life" in his seminal work popularly known as *On the Origin of Species* [34]. Darwin's first tree (Figure 4.1)

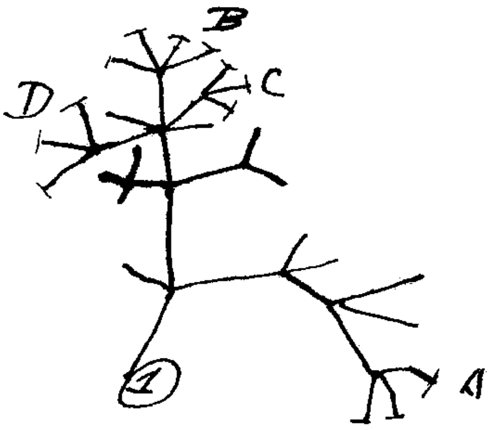


Figure 4.1: Charles Darwin's first diagram of an evolutionary tree from his notebook *Transmutation of species*, 1837 [33].

shows the theoretical relationship between an ancestral species (1) and its descendant species [147]. Extant species are shown by tips on the endpoints of some branches with the remaining branches possibly representing extinct species. His idea was that groups of species would have diverged at different times and therefore some groups will be more closely related than others. For example, the species labelled by B and C would be more closely related than those labelled by A and D.

Trees as a formally defined mathematical entity, such as the ones will see in the following sections, appeared as early as 1847 with the name "tree" appearing in the literature shortly afterwards [96]. The more general theory of graphs and topology was developed earlier and is considered to have begun with Euler who in 1735 presented the paper "Solutio problematis ad geometriam situs pertinentis" [52] in which it was shown that it was not possible to walk through the city of Königsburg crossing each of its seven bridges exactly once.

Tree structures are of great importance in computer programming. One of the earliest programs to make explicit use of tree structures used them to represent algebraic formulæ for the purpose of symbolic differentiation [96, 88]. This would later become the field of computer algebra which would drive the development of computer science and especially programming languages for many years.

Intuitively, it makes sense to arrange many different types of information in trees, particularly that for which we can define a distance. A first question to ask is whether a representation in terms of a tree exists for a given dataset and distance function. As we will see, when the distance function satisfies certain properties there always exists a tree representation and this representation is also unique (in a well-defined sense). We focus then on a slightly different problem: imagine we know the distance between only certain pairs of objects in our dataset. We call this information a partial distance. This presents two separate problems. First, does there exist a tree representation of a given partial distance and if so, can we find one? Second, does there exist a unique tree representation of a given partial distance, and if so can we find it? These issues are the subject of the remainder of this thesis.

4.2.2 Basic terminology and assumptions

In this section we introduce much of the terminology that is required for dealing with trees. Since trees are special cases of graphs, we begin with general graph theory before moving on to trees and, in particular, a special type of tree that is of greatest interest to us: the equidistant tree. As noted by Knuth [96] there is a large degree of variation in the terminology used by different authors in graph theory. We try to follow the terminology used in current phylogenetics literature such as [147] and [46]. Throughout this section, let X denote a finite nonempty set.

Graphs

A *graph* is an ordered pair (V, E) where V is a set of *vertices* and E is a set (or multiset) of *edges*, each of the form $\{x, y\}$ such that $x, y \in V$ are distinct. Unless specified otherwise, all graphs in this thesis are *simple* and finite meaning that V and E are finite sets and there are no loops. Suppose $G = (V, E)$ is a graph. Two vertices $v, v' \in V$ are said to be *adjacent* if there exists an edge in G joining v and v'. An edge $\{x, y\} \in E$ is said to be *incident* with

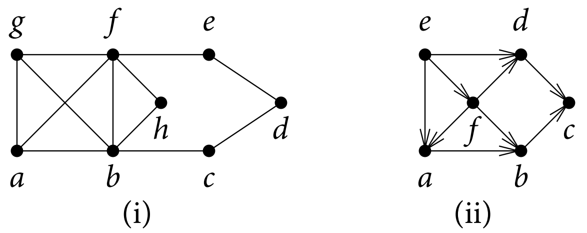


Figure 4.2: A graph (i) and a directed graph (ii).

the vertices x and y. The *degree* of a vertex $v \in V$ is the number of edges in G incident with it.

A *path* is a sequence of distinct vertices $v_1, v_2, \ldots, v_k$ where $k \geq 2$ such that for all $i \in \{1, \ldots, k-1\}, v_i$ and v_{i+1} are adjacent. If $k \geq 3$ and v_1 and v_k are also adjacent then the path is called a *cycle*. A graph is *connected* if between each pair of distinct vertices there exists a path joining them. A graph is *complete* if there is an edge joining each pair of distinct vertices. A subset $V' \subseteq V$ is called a *clique* if there is an edge joining each pair of distinct vertices in V'. Figure 4.2 helps to illustrate these definitions. In the graph shown in (i) an example of a path is g, f, e, d and an example of a cycle is g, f, b, a, g. The graph is connected, but not complete, and the set $\{a, b, g, f\}$ is a clique.

A *directed graph* is an ordered pair (V, E) where E is a set of *directed edges* or *arcs*. An arc $(v, v') \in E$ is said to lead from v to v'. The *out-degree* of a vertex is the number of arcs leading out from it and the *in-degree* is the number leading in. The degree of a vertex is therefore the sum of its out-degree and in-degree. An *oriented path* is a sequence of distinct vertices $v_1, \ldots, v_k$ such that there exists an arc from v_i to v_j for all $1 \leq i \leq k-1$. Figure 4.2 (ii) shows a directed graph to help illustrate these definitions. Vertex f has out-degree 3, in-degree 1 and therefore degree 4. An example of an oriented path is e, f, b, c.

Two graphs (V, E) and (V', E') are called *isomorphic* if there exists a bijection $\phi\colon V \to V'$ such that $\{v, v'\} \in E$ if and only if $\{\phi(v), \phi(v')\} \in E'$. So, in other words, adjacency of vertices is preserved. A graph (V', E') is a *subgraph* of a graph (V, E) if $V' \subseteq V$ and $E' \subseteq E$. Further if $V' \subseteq V$ and E' contains all edges $\{v, v'\} \in E$ whenever $v, v' \in V'$ the graph (V', E') is an *induced subgraph* of (V, E).

Trees

In this section we formally define trees as a special type of graph and introduce all the relevant terminology for a special type of tree which we will focus on.

A graph that is connected and has no cycles is called a *tree*. Trees can be characterised in many ways, some of which are given in the following theorem (see [96, Section 2.3.4.1] for a proof):

Theorem 15. *If G is a finite graph with $n > 0$ vertices, the following statements are equivalent:*

a) G is a tree,

b) G is connected, but if any edge is deleted the resulting graph is no longer connected,

c) There is exactly one path between any two distinct vertices of G,

d) G contains no cycles and has $n - 1$ edges,

e) G is connected and has $n - 1$ edges.

For example, Figure 4.3 shows two graphs which are trees. The tree depicted in (ii) is a special type of tree called a *rooted tree*. A rooted tree (or *oriented tree*) is a directed graph G with a distinguished vertex ρ_G (the root) such that [96]:

a) The root ρ_G has in-degree 0,

b) Each vertex v of G apart from ρ_G has in-degree 1,

c) There is a path between ρ_G and any vertex of G that is not the root.

Suppose T is a rooted tree. A vertex in T is called a *leaf* if it has degree 1. All other vertices are called *interior* vertices. An edge which is incident with a leaf is called a *pendant* edge. All other edges are called *interior* edges. The set of all leaves of T is called the *leafset* of T which we denote by $L(T)$.

Suppose X is a finite, nonempty set, a *rooted phylogenetic X-tree* is a rooted tree with no vertices of in-degree one and out-degree one and with leafset X. Since the remainder of this

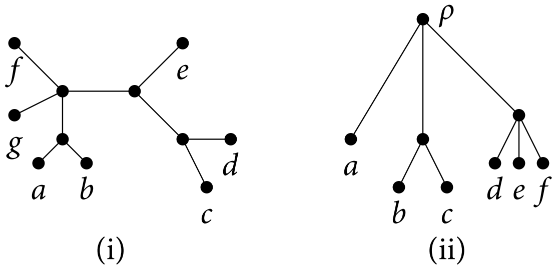


Figure 4.3: A tree (i) and a rooted phylogenetic X-tree (ii).

thesis is concerned with rooted trees we will from now on refer to a rooted phylogenetic X-tree as simply an *X-tree*. Figure 4.3 (ii) shows an X-tree with $X = \{a, \dots, f\}$.

An X-tree is *binary* if all interior vertices have degree 3 apart from the root which has degree 2. We therefore have that each vertex has out-degree zero or two and each vertex apart from the root has in-degree 1. By considering the directed paths from the root to leaves it becomes clear why such a tree is called binary [147].

An *edge-weighted graph* (G, ω) is a graph $G = (V, E)$ paired with an edge-weighting function $\omega\colon E \to \mathbb{R}$. For an edge-weighted X-tree T we call an edge-weighting *proper* if $w(e) > 0$ for every interior edge e of T.

An X-tree has a natural partial order on the vertices. Given an X-tree $T = (V, E)$, a vertex $v \in V$ is an *ancestor* of a vertex $v' \in V$ if and only if v is on the directed path from the root of T to v'. The vertex v' is then said to be a *descendant* of v. If v and v' are adjacent then we also call v the *parent* of v' and v' a *child* of v. The number of children of a particular vertex equals its out-degree.

The *lowest common ancestor* of two vertices v and v' in an X-tree is the unique vertex u such that u is an ancestor of both v and v' and the path from the root ρ to u is longer than the path to any other ancestor of both v and v'. The lowest common ancestor of v and v' is denoted $\mathrm{lca}(v, v')$. In the X'-tree in Figure 4.3 (ii) the lowest common ancestor of a and d is the root ρ.

A connected subgraph of a tree is called a *subtree*. If T is an X-tree and $X' \subseteq X$ then we denote by $T|X'$ the subtree of T whose leafset is X', with degree 2 vertices suppressed.

$T|X'$ is then called a *restricted subtree.* If T is binary and $|X'| = 3$ then $T|X'$ is called a *triplet.*

We call two X-trees T and T' *equivalent* (written $T \simeq T'$) if there exists a bijection $\phi\colon V(T) \to V(T')$ that extends to a graph isomorphism that is the identity on X. Since an X-tree is rooted, ϕ must also map ρ_T to $\rho_{T'}$.

For a set X the number of possible non-equivalent binary X-trees is $(2|X|-3)!!$ (where !! means double factorial). Therefore to find a tree in this space with particular properties we cannot look at each possible tree and must instead use a special tree reconstruction method. Next we look at how distances are related to trees and then at the problem of reconstructing a tree from distance information.

Ultrametrics

Given an edge-weighted X-tree (T, ω) where $T = (V, E)$ and $\omega\colon E \to \mathbb{R}_{\geq 0}$ we can associate a distance $D_{(T,\omega)}\colon X \times X \to \mathbb{R}_{\geq 0}$ to (T, ω) by setting for all $x, y \in X$:

$$D_{(T,\omega)}(x, y) = \begin{cases} \sum\limits_{e \in P(x,y)} \omega(e) & \text{if } x \neq y, \\ 0 & \text{otherwise,} \end{cases}$$

where $P(x, y)$ is the set of edges on the path from x to y. We call $D_{(T,\omega)}$ the distance *induced* by (T, ω).

The distance induced by a tree is a special type of metric called a *tree metric.* These metrics have some interesting properties which are discussed in [147, Chapter 7]. For our purposes we are most interested in a special type of tree metric called an *ultrametric.* A distance $\delta\colon X \times X \to \mathbb{R}$ is called an ultrametric if for every distinct $x, y, z \in X$ the following stronger form of the triangle inequality holds:

$$\delta(x, y) \leq \max(\delta(x, z), \delta(y, z)).$$

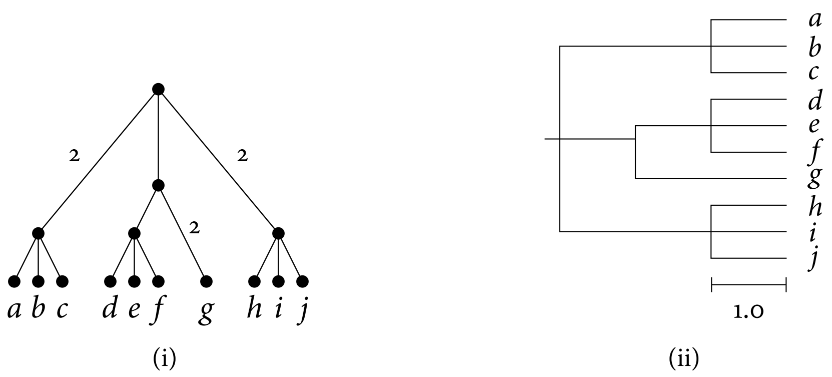


Figure 4.4: A dendrogram is a visual representation of a tree. Here we see two different ways to draw the same edge-weighted X-tree.

An edge-weighting $\omega\colon E \to \mathbb{R}$ for an X-tree $T = (V, E)$ with root ρ_T is called *equidistant* if it satisfies the following properties:

(i) $D_{(T,\omega)}(\rho_T, x) = D_{(T,\omega)}(\rho_T, y)$ for all $x, y \in X$,

(ii) $D_{(T,\omega)}(x, y) \geq D_{(T,\omega)}(x, u)$ for all $x \in X$ and any $u, v \in V$ whenever u is encountered before v on the directed path from x to ρ.

If ω is an equidistant edge-weighting then $D_{(T,w)}$ is an ultrametric [147, Lemma 7.2.4]. X-trees with equidistant edge-weightings arise naturally in many places, for example in phylogenetics. For convenience we will from now on refer to such an X-tree with equidistant edge-weighting as simply an *equidistant X-tree*.

Figure 4.4 shows an example of an equidistant X-tree with $X = \{a, \ldots, j\}$ using two common display formats (dendrograms). In (i) the edges are simply labelled with their weights (edges with weight 1 are left unlabelled for clarity). In (ii) the weight of each edge is proportional to the horizontal component of its length on the bar with the actual length shown with a scale bar.

For any ultrametric $\delta\colon X \times X \to \mathbb{R}$ there is, up to isomorphism, a unique equidistant X-tree (T, ω) such that $\delta(x, y) = D_{(T,\omega)}(x, y)$ for all $x, y \in X$. This tree can be recovered from the ultrametric in polynomial time, as we will see in the next section.

4.3 Tree reconstruction

In this section, we look at various methods of reconstructing trees. Of most interest to us is the problem of reconstructing an X-tree from a distance on X. This is generally done by one of two broad classes of hierarchical clustering methods which we discuss in Section 4.3.2.

A further problem is that of reconstructing an edge-weighted X-tree from a distance on X. This is desirable in many cases, for example in phylogenetics the edge-weights might represent the time or number of point mutations [55].

Closely related to this is the problem of reconstructing X-trees from only partial distance information. This means that we do not have the complete distance d, but rather only have the values $d(x, y)$ for some, but not all, pairs $x, y \in X$. This problem is motivated by the fact that accurate distance measurements are often difficult or impossible to obtain in practice but one would still like to build an edge-weighted X-tree given only what is known.

As with complete distance information, it is possible for partial distance information to uniquely determine an X-tree. This is the subject of Section 4.4. This allows us to define a consistency property for partial distance reconstruction methods.

4.3.1 Reconstruction from subtrees

Before we look at reconstructing trees from distances, it is interesting to note that we do not actually need a distance to reconstruct only the topology of an X-tree. It suffices merely to know the set of subtrees displayed by the tree [147].

A pair of leaves of an X-tree that share the same parent is called a *cherry* and a set of leaves (of any size) that share the same parent is called a *pseudocherry*. As before, an X-tree with $|X| = 3$ that contains a cherry is called a triplet. If P is an $\{a, b, c\}$-tree a, b is a cherry of P then we denote the triplet P by $ab|c$. We say that a triplet $ab|c$ is *displayed* by an X-tree T if the restriction $T|\{a, b, c\}$ of T to $\{a, b, c\}$ is binary and $lca_T(a, c) = lca_T(b, c)$ is an

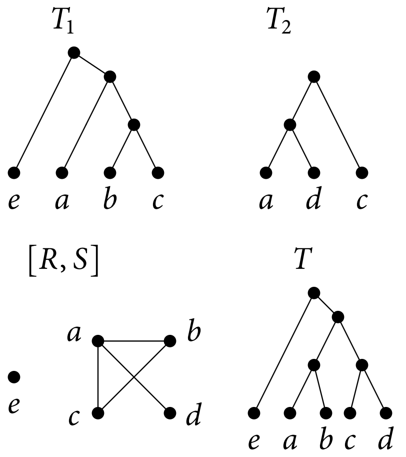


Figure 4.5: An illustration of BUILD with an input of $R = \{T_1, T_2\}$. $[R, S]$ is the auxiliary graph built in the first iteration of the algorithm and T is the final tree constructed.

ancestor of $lca_T(a, b)$ in T. For example, the tree T_2 in Figure 4.5 is a triplet and can be written as $ad|c$.

The BUILD algorithm [1] can be used to reconstruct an X-tree T from the set of all triplets displayed by T. We show the BUILD algorithm in pseudocode in Algorithm 6 and give an example in Figure 4.5. The algorithm uses a set R of rooted trees $\{T_1, \dots, T_n\}$ where $L(T_1) \cup \cdots \cup L(T_n) = X$ and an auxiliary graph $[R, S]$ which has vertex set $S \subseteq X$ and an edge $\{a, b\}$ for each $a, b \in S$ whenever there exists some $c \in S$ and some $T' \in R$ such that $T'|\{a, b, c\}$ and $ab|c$ are equivalent.

This algorithm has some nice properties, for example if BUILD returns a tree then the set of input trees is said to be *compatible*. The definition of compatible is independent of BUILD (see [147]) but BUILD can be used to check compatibility. Furthermore, if we input the set of all triplets displayed by a tree T, then BUILD is guaranteed to output T and therefore T is uniquely determined by the set of triplets it displays.

If the inputted set of triplets R is some subset of all the triplets displayed by a tree then the set is still compatible but this set is displayed by possibly many trees. In this case the output of BUILD is deterministic and the tree reconstructed is known as *the BUILD tree for R*. Some interesting properties of these trees can be found in [20, Section 2.5.2].

Algorithm 6 BUILD.

```
Input: A set of rooted trees $R = \{T_1, \dots, T_n\}$.
Output: An $X$-tree $T$ where $X = L(T_1) \cup \dots \cup L(T_n)$.
  $S = \{x_1, \dots, x_m\} \leftarrow X$.
  if $|S| = 1$ then return the tree $(\{x_1\}, \varnothing)$.
  end if
  if $|S| = 2$ then let $\rho$ be a new vertex and return the tree with root node $\rho$ obtained by
  attaching $x_1$ and $x_2$ to $\rho$.
  end if
  Construct the auxiliary graph $[R, S]$.
  Let $S_1, \dots, S_k$ be the vertex sets of the connected components of $[R, S]$.
  for all $1 \leq i \leq k$ do
      Let $T_i$ be the output of BUILD on $R_i = \{T|S_i : T \in R\}$.
  end for
  Let $\rho$ be a new vertex and return the tree with root node $\rho$ obtained by attaching the
  root of each $T_i$ to $\rho$.
```

The BUILD algorithm was initially developed to construct a tree satisfying a given set of constraints and was applied to problems arising in the theory of relational databases. It was later applied to phylogenetics but remained relatively unknown in the field for a number of years [152, 21].

When the inputted trees are not compatible, BUILD returns an error. Since it is often the case that trees will not be compatible, the MINCUTSUPERTREE algorithm was developed which will construct a tree even if the input trees are not compatible [146]. The algorithm builds a graph similar to $[R, S]$ and ensures that it is disconnected in each step by using a minimum cut. This algorithm was developed primarily for construction of phylogenetic supertrees.

4.3.2 Reconstruction from distances

We now turn to the problem of reconstructing an X-tree from a distance. We begin with a brief overview of general hierarchical clustering methods and then look at the UPGMA method [149] and its optimal runtime algorithm. Hierarchical clustering methods in general build unweighted X-trees, but UPGMA can be considered an extension which

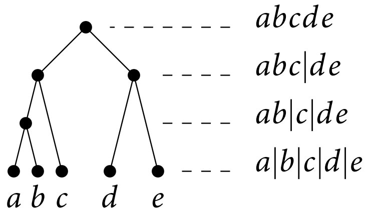


Figure 4.6: A tree can be viewed as a hierarchy of partitions.

also constructs an edge-weighting for the constructed tree.

For a distance-based reconstruction method the following property is desirable: if we input a distance function $D_{(T,\omega)}$ induced by an edge-weighted X-tree (T,ω) we get the tree T and its edge-weighting ω. If a method enjoys this property then we call it *consistent*. For many methods this property holds only under certain conditions. For UPGMA this holds if (T,ω) is a binary equidistant X-tree [49].

Hierarchical clustering methods

Hierarchical clustering is used for the classification of information in much the same way as partitional clustering. The aim of a hierarchical clustering method is to produce an X-tree corresponding to a given distance D on a set X. While a partition of a set is merely a set of disjoint subsets (clusters) which "cover" the set, an X-tree can be viewed as a hierarchy of such clusters which induces several partitions on X. This natural relationship between trees and partitions is illustrated in Figure 4.6. In the example we have $X = \{a,\dots,e\}$. The root of the tree corresponds to the single cluster X (denoted $abcde$) and as we move down the tree the set is partitioned further until we have cluster of one leaf each ($\{\{a\},\dots,\{e\}\}$, denoted $a|b|c|d|e$ for short).

There are two main methods in use for building hierarchies. These are the *agglomerative* or "bottom-up" methods which begin with each element of X in its own cluster and successively merges clusters until there is only one cluster left, and the *divisive* or "top-down" methods which begin with a single cluster and successively splits clusters until each element is on its own. The resulting hierarchy depends upon which merges or splits have been chosen at each stage which, in turn, depends on finding a local optimum according to some criterion.

Algorithm 7 Agglomerative hierarchical clustering algorithm.

```
Input: A set $X$ and a linkage function $D\colon 2^X \times 2^X \to \mathbb{R}$ with an underlying distance
  $d\colon X \times X \to \mathbb{R}$.
Output: A rooted $X$-tree $T$.
  Let $F$ be a forest of $|X|$ trees each containing a unique element of $X$ as the only vertex,
  which is also the root,
  while $|F| > 1$ do
      Let $v$ be a new vertex and $(x, y) \leftarrow \arg\min_{x,y \in F} D(L(x), L(y))$,
      Remove trees $x$ and $y$ from $F$ and add the tree obtained by attaching the roots of $x$
  and $y$ to $v$ and letting $v$ be the root,
  end while
  return the single tree contained in $F$.
```

The general algorithm for agglomerative clustering is shown in Algorithm 7. The choice of linkage function determines how the distances are recomputed at each stage and therefore which trees are joined in each subsequent stage. Given a set X and distance $d\colon X \times X \to \mathbb{R}$ and two nonempty subsets $C_1, C_2 \subseteq X$, some common choices for linkage functions include:

single-linkage:

$$D_{SL}(C_1, C_2) = \min_{x \in C_1, y \in C_2} d(x, y),$$

complete-linkage:

$$D_{CL}(C_1, C_2) = \max_{x \in C_1, y \in C_2} d(x, y),$$

and average-linkage:

$$D_{AL}(C_1, C_2) = \left(\sum_{x \in C_1} \sum_{y \in C_2} d(x, y) \right) / |C_1||C_2|.$$

The general algorithm for agglomerative clustering has runtime complexity of $O(n^3)$ where $n = |X|$ since there are $O(n)$ merges to be done and finding the arg min takes $O(n^2)$ time. However, for many linkage criteria, including the three above, it is possible to use an $O(n^2)$ algorithm. Two well known examples are SLINK [148] and CLINK [39] for using single-linkage and complete-linkage respectively. In the next section we will see that hierarchical clustering using average-linkage may be done in quadratic time too.

The divisive method works in the opposite direction: we begin with a single cluster and successively split clusters until we end up with one cluster per leaf. Divisive methods are usually significantly more complicated than agglomerative methods. Each step requires an initial decision about which cluster to split, and then a decision about how to split that cluster [141]. A naïve approach for deciding which cluster to split next is to always split the largest cluster, but there are more sophisticated approaches, for example based on cluster homogeneity. Splitting a cluster then essentially requires a partitional clustering method [43]. In general the time complexity of a divisive method is $O(2^n)$ where $n = |X|$ making it prohibitively expensive for most applications [27].

UPGMA

Unweighted Pair Group Method with Arithmetic Mean (UPGMA) [149] is in fact a modified agglomerative clustering method using average-linkage which reconstructs an equidistant X-tree. For an equidistant X-tree (T, ω), let $height((T, \omega))$ be the sum of the edge weights on the path from the root of T to any leaf, or 0 if the root is the sole vertex. The algorithm for UPGMA is shown in Algorithm 8.

Since only two trees are joined in each iteration, the output of UPGMA is always a binary tree. If the input distance d is equal to the induced distance $D_{(T,\omega)}$ of some equidistant binary X-tree (T, ω) then UPGMA is guaranteed to return (T, ω) [49].

In the general agglomerative algorithm, each step involves identifying a pair of elements such that no other pair of elements in X are closer according to the chosen linkage criterion. In other words, we are finding a global minimum. In more efficient algorithms we instead look only for a local minimum at each stage. A local minimum in this context means a pair of elements which are *mutual nearest neighbours* (MNNs). Given a set X and a distance d on X, a pair $x, y \in X$ are MNNs if there exists no element $z \in X$ for which

Algorithm 8 UPGMA.

```
Input: A set $X$, and a distance $d\colon X \times X \to \mathbb{R}$.
Output: A binary equidistant $X$-tree $(T, \omega)$.
  Let $F$ be a forest of $|X|$ trees each containing a unique element of $X$ as the only vertex,
  which is also the root.
  while $|F| > 1$ do
      Let $v$ be a new vertex,
      put $m \leftarrow \min_{x,y \in F} D_{AL}(L(x), L(y))$,
      and $(x, y) \leftarrow \arg\min_{x,y \in F} D_{AL}(L(x), L(y))$.
      Remove trees $x$ and $y$ from $F$ and add the tree obtained by attaching the root of $x$
  to $v$ with an edge of length $m/2 - height(x)$, attaching the root of $y$ to $v$ with an edge
  of length $m/2 - height(y)$ and letting $v$ be the root.
  end while
  return the $X$-tree contained in $F$ and its edge-weighting.
```

$d(x,z) < d(x,y)$ or $d(y,z) < d(x,y)$. Such a pair can be safely agglomerated as soon as it is found, just as in the general algorithm [121].

Finding MNNs can be done quickly by building a chain of nearest neighbours in the following way: we begin with an arbitrary cluster in X and find its nearest neighbour giving us a chain of length two. We then repeatedly add to the chain the nearest neighbour of the last element in the chain until we find a pair of MNNs. The MNNs are removed from the chain and agglomerated. The distances between new clusters and all other clusters are calculated recursively using the Lance-Williams update formula in constant time [100]. The process is then continued from the end of the chain, or from an arbitrary cluster if the chain is empty. This method works because for certain linkage criteria, including average-linkage, the nearest neighbour chain remains valid after an agglomeration (see [69] for details).

This mutual nearest neighbour method, also called the *algorithme des célibataires*, was developed in the early 1980s and initially appeared in [37] and [87] (in French) and later in [119] and [120] (in English). It leads to the optimal $O(n^2)$ version of UPGMA and other hierarchical clustering methods [69].

4.3.3 Reconstruction from partial distances

Partial distances arise often in practice, that is a distance where the distance between some pairs of elements is missing. Potential reasons for this might be that it is sometimes difficult or impossible to make measurements between certain pairs due to those pairs not sharing any information from which a distance can be inferred. For example, in biological studies involving large numbers of species and genes it is often the case that species do not share any genes in the available data [31].

For this reason it becomes useful to ask whether a partial distance can be extended into a complete distance that is also an ultrametric. More formally, a *partial distance* on X is a function $d^*\colon (X \times X) - M \to \mathbb{R}_{\geq 0}$ where M is the set of pairs for which the distance value is missing. The problem is to find a complete distance $d\colon X \times X \to \mathbb{R}_{\geq 0}$ where $d(x, y) = d^*(x, y)$ for all $x, y \in (X \times X) - M$ and which is an ultrametric. In [54] it was shown that it is possible to decide in polynomial time whether an extension to an ultrametric exists (and to compute such an extension), although the corresponding decision problem for unrooted trees (that is, deciding if an extension to a general tree metric exists) is NP-complete.

Below we describe three different methods proposed in the literature for extending a partial distance on X to an ultrametric on X and reconstructing the corresponding equidistant X-tree. We restrict ourselves to equidistant X-trees but similar results on unrooted trees may be found in, for example, [70], [54], [108] and [71].

An optimisation method

The approach taken by de Soete [38] is to consider a least squares constrained optimisation problem. Given a partial distance $d^*\colon (X \times X) - M \to \mathbb{R}_{\geq 0}$, a function $Loss\colon \mathbb{R}_{\geq 0}^{X \times X} \to \mathbb{R}_{\geq 0}$ (where $\mathbb{R}_{\geq 0}^{X \times X}$ is the set of all dissimilarities on X) is defined as:

$$Loss(d) = \sum_{(x,y)\in(X\times X)-M} \left(d^*(x, y) - d(x, y)\right)^2.$$

This is called the *loss function*. The problem is now to find a function $d \colon X \times X \to \mathbb{R}_{\geq 0}$ such that $Loss(d)$ is minimised and with the constraint that d is an ultrametric.

To solve the constrained minimisation problem the authors propose to use the sequential unconstrained minimisation technique (SUMT) [57]. Under this technique the constraint that d be an ultrametric is removed and instead we find a distance $d_n \colon X \times X \to \mathbb{R}_{\geq 0}$ which minimises the augmented function $\Phi \colon \mathbb{R}_{\geq 0}^{X \times X} \times \mathbb{R}_{\geq 0} \to \mathbb{R}_{\geq 0}$ defined as:

$$\Phi(d_n, \sigma) = Loss(d_n) + \sigma Pen(d_n), \qquad (\sigma > 0),$$

where $Loss$ is the loss function and $Pen \colon \mathbb{R}_{\geq 0}^{X \times X} \to \mathbb{R}_{\geq 0}$ is called the *penalty function* which is meant to enforce the ultrametric condition. Pen is defined as:

$$Pen(d_n) = \sum_{(i,j,k) \in \Omega(d_n)} \left(d_n(i,k) - d_n(j,k) \right)^2$$

where

$$\Omega(d_n) = \{(i,j,k) \in X^3 \colon d(i,j) \leq \min\left(d_n(i,k), d_n(j,k)\right) \text{ and } d_n(i,k) \neq d_n(j,k)\}.$$

In other words, $\Omega(d_n)$ denotes the set of 3-tuples for which the ultrametric condition is violated. The unconstrained minimisation is performed successively with an increasing value for σ, each time using the previous result d_{n-1} to begin the search for the next d_n. A method by [127] is used to perform the unconstrained minimisation.

An agglomerative method

Missing Values Linkage (MVL) is an example of an agglomerative approach proposed by [143]. It is actually identical to the agglomerative hierarchical clustering algorithm (see Section 4.3.2) but using a linkage criterion modified to take into account missing values.

The modified average-linkage criterion for two nonempty sets S_1, S_2 is:

$$D_{MVL}(S_1, S_2) = \begin{cases} \dfrac{\sum\limits_{(x,y)\in(S_1\times S_2)-M} d(x,y)}{|(S_1 \times S_2) - M|} & \text{if } (S_1 \times S_2) - M \neq \varnothing, \\ \infty & \text{otherwise,} \end{cases}$$

where $M = \{(i,j) \in X \times X : d(i,j) \text{ is unknown}\}$. Modified versions of other linkage criteria are possible; some more examples are given in the original paper.

The authors do not provide an algorithm with better time complexity than that of the naïve $O(n^3)$ agglomerative algorithm (where $n = |X|$). They also show that the MVL method produces very similar results to de Soete's method.

Figure 4.7 (ii) shows the tree constructed by MVL from a partial distance d on a set $\{a, \dots, e\}$ containing only the distances $d(a,b) = 1, d(b,c) = 1, d(c,d) = 2, d(c,e) = 2, d(d,e) = 3$.

A divisive method

Farach et al. [54] use a top down, divisive approach. Given a partial distance function $d^*: (X \times X) - M \to \mathbb{R}_{\geq 0}$ let $G = (V, E)$ be a graph with vertex set X and an edge $\{x, y\}$ with $x, y \in X$ but $(x, y) \notin M$ (so the distance between x and y is known). We also define an edge-weighting $\omega: E \to \mathbb{R}$ by $\omega(\{x, y\}) = d^*(x, y)$ for all $x, y \in E$.

The tree reconstruction algorithm proceeds as follows:

1. If $E = \varnothing$, return the single element in V.
2. Otherwise, put $m \leftarrow max_{e\in E}\omega(e)$.
3. Let $G^* = (V, E^*)$ be a graph and put $E^* \leftarrow \{e \in E : \omega(e) < m\}$.
4. Let u be a new vertex and return the tree with root node u obtained by recursing on each connected component of G^* and attaching the results of each to u.

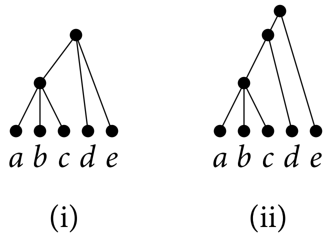


Figure 4.7: The two different trees constructed by Farach's method (i) and the MVL method (ii) from the same partial distance information.

This basic algorithm shown here requires $O(|V||E|)$ time but a quicker algorithm requiring only $O(|E| + |V|\log|V|)$ time is also given by [54].

Figure 4.7 (i) shows the tree constructed by this method from the same partial distance d on $\{a, \ldots, e\}$ containing only the distances $d(a, b) = 1, d(b, c) = 1, d(c, d) = 2, d(c, e) = 2, d(d, e) = 3$. Note that the tree constructed is different to the tree constructed by MVL.

4.4 Lassos

As we saw in the example shown in Figure 4.7, the existing partial distance reconstruction methods can construct different trees from the same input. The problem with the example partial distance in that case was that no ultrametric extension exists so the methods each modified the given distances in different ways to obtain an ultrametric. This means that for some pairs of leaves in the resulting tree the induced tree distance is not equal to the inputted distance. This may be undesirable.

A second problem is that often more than one extension to an ultrametric is possible. We will see this in the example in the following section. In the case where more than one extension is possible the existing algorithms will provide one of many solutions. But we may wish to know not only that an ultrametric extension exists but that it is unique. This section provides us with some theory for deciding when this is possible.

Certain partial distances on X have the property that they uniquely determine the topology or edge-weights of an X-tree. In other words, only one extension to an ultrametric

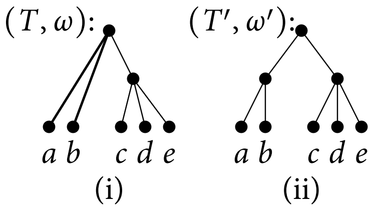


Figure 4.8: Two equidistant X-trees. All edges have weight 1 except bold edges which have weight 2. For $\mathcal{L} = \{ac, de, bc, ce, cd\}$ both trees induce the same distances over the cords in $\mathcal{L}$ despite having different topologies. In this case $\mathcal{L}$ is not a topological lasso.

on X is possible. To enable us to understand when partial distances have this uniqueness property, the theory of lassos was developed. We next define certain types of lassos as applied to equidistant X-trees and look at some important characterisations.

4.4.1 Definitions and basic properties

Let (T,ω) and (T',ω') be two edge-weighted X-trees and $\mathcal{L} \subseteq \binom{X}{2}$ be some subset of pairs of elements in X. For convenience we denote an element $\{x,y\} \in \mathcal{L}$ as simply xy. Then (T,ω) and (T',ω') are called *$\mathcal{L}$-isometric* if $D_{(T,\omega)}(x,y) = D_{(T',\omega')}(x,y)$ for all $xy \in \mathcal{L}$.

Now, given an X-tree T and a subset $\mathcal{L} \subseteq \binom{X}{2}$ we define $\mathcal{L}$ to be:

(i) an *equidistant lasso* for T if $\omega = \omega'$ holds for all equidistant edge-weightings ω, ω' of T where (T,ω) and (T,ω') are $\mathcal{L}$-isometric,

(ii) a *topological lasso* for T if T is equivalent to T' for any X-tree T' for which there exist proper edge-weightings ω and ω' of T and T' respectively where (T,ω) and (T',ω') are $\mathcal{L}$-isometric,

(iii) a *strong lasso* T if $\mathcal{L}$ is both an equidistant and topological lasso for T.

To illustrate these concepts consider the two equidistant X-trees (T,ω) and (T',ω') depicted in Figure 4.8. The edge-weights are proportional to the length of the edges in the figure and induce distances $D_{(T,\omega)}\colon X \times X \to \mathbb{R}$ and $D_{(T',\omega')}\colon X \times X \to \mathbb{R}$ respectively. The trees are therefore equidistant. If we have the set of cords $\mathcal{L} = \{ac, de, bc, ce, cd\}$ then notice that the distances between each of these pairs of leaves is the same according to the induced distances for both trees (for example $D_{(T,\omega)}(a, c) = D_{(T',\omega')}(a, c)$). Therefore $\mathcal{L}$ is not a topological lasso. Notice that this property of $\mathcal{L}$ is dependent solely on the structure of the set itself and not on the actual distances on the trees. For further information and discussion about lassos, especially those concerning more general trees, the reader is referred to [46] and [82].

4.4.2 Characterising lassos: the child-edge graph

Lassos for X-trees can be characterised using a graph associated to each interior vertex of a tree [81]. First, a lasso $\mathcal{L}$ of X which satisfies the property that $\bigcup_{c \in \mathcal{L}} c = X$ is called a *covering* of X. Let T be an X-tree and $\mathring{V}(T)$ denote the set of internal vertices of T. For some $v \in \mathring{V}(T)$, and a set of cords $\mathcal{L} \subseteq \binom{X}{2}$, we associate a graph called $G(\mathcal{L}, v)$ defined as follows. The vertex set of the graph V_v is the set of all child edges of v. The edge set E_v is the set of all $\{e, e'\} \in \binom{V_v}{2}$ for which there exist leaves $a, b \in X$ and a cord $ab \in \mathcal{L}$ such that e and e' are edges on the path from a to b in T. We call the graph $G(\mathcal{L}, v)$ the *child-edge graph* (*of v with respect to T and $\mathcal{L}$*).

We have the following characterisation of topological lassos which originally appeared in [81]:

Theorem 16. *Suppose T is an X-tree and $\mathcal{L} \subseteq \binom{X}{2}$ is a covering of X. Then the following are equivalent:*

1. *$\mathcal{L}$ is a topological lasso for T,*

2. *for every vertex $v \in \mathring{V}(T)$, the graph $G(\mathcal{L}, v)$ is complete.*

Due to the characterisation of equidistant lassos also presented in [81], we also have that every topological lasso for T must also be an edge-weight lasso, and therefore a strong lasso, for T.

Chapter 5

Constructing Trees from Lassos

This chapter is largely based on an early version of the following paper:

G. Kettleborough, J. Dicks, I.N. Roberts, and K.T. Huber. Reconstructing (super)trees from data sets with missing distances: Not all is lost. *Molecular Biology and Evolution*, 2015. doi: 10.1093/molbev/msv027

~

I developed the Lasso algorithm and the simulation study for testing it. I was also responsible for processing the biological datasets and using them for testing. This involved implementation of the algorithm and methods described. I also implemented the methods for drawing the phylogenetic trees found in the chapter.

5.1 Introduction

5.1.1 Summary

In this chapter we introduce a new algorithm for reconstructing an equidistant X-tree from a partial distance on X. The algorithm returns both an X-tree and a subset of the distances given which correspond to an equidistant lasso for the X-tree. To understand the performance of Lasso, we assess it by means of artificial and real biological datasets, showing its effectiveness in the presence of missing data even in the presence of noise. We also show how it can be applied to the problem of combining datasets to build a supertree and compare our method with a well known supertree method.

5.1.2 Motivation

The ease and speed with which molecular sequence data can now be generated using modern Next Generation Sequencing (NGS) technologies has enabled evolutionary biologists to embark on exciting, albeit highly challenging, endeavours such as the Tree of Life project. NGS has perhaps been most influential at the sub-species level, and datasets encompassing numerous lines, strains or accessions are becoming commonplace. These new data, together with a wealth of legacy datasets, promise the interleaving of species and sub-species within a common evolutionary framework. However, to increase our chances of successfully constructing such a tree numerous obstacles have to be overcome, ranging from data collection via data storage and information extraction to tree building. The vastness of tree space one faces in overcoming the latter obstacle, that of tree building, coupled with the computational demands of Bayesian, likelihood and parsimony approaches implies that reconstruction methods based on these approaches cannot, at least at present, be directly applied to obtain such a tree. Given the large amount of legacy phylogenetic data, a natural solution is to try to find ways to merge what is already known and then to augment the result. Apart from having to deal with the problem of patchy taxonomic coverage, as some taxa have been studied more than others (see, for example, [140, 126, 153, 136] for more on this), constructing such a tree does not only entail finding ways to combine different types, qualities, and quantities of data but also addresses the problem of how to combine data sets that might only share very few taxa. The latter is a formidable problem in its own right and boils down to finding powerful ways of dealing with missing information.

Depending on the study within which a dataset was generated missing information may take the form of trees that have few taxa in common, or missing values in character state or distance matrices. To tackle the former, numerous supertree approaches have been introduced in the literature (see, for example, [15] and [18] and the references therein). A similar number of supermatrix approaches have also been proposed to deal with missing values in character state matrices (see, for example, [15]) while the number of approaches addressing missing values in distance matrices is comparatively small (see [32, 31, 108, 70, 71, 38, 66]).

The starting point for many supertree approaches is a collection of (potentially very small) phylogenetic trees and the goal is to find some kind of parental tree on all the taxa of the input trees that in some sense displays the evolutionary information contained within them. In contrast, the starting point for the remaining two approaches are character state matrices and distances on differing taxa sets, respectively. The aim here is to combine them into a supermatrix or distance, respectively, on all the taxa in the combined taxa set using some sort of imputing scheme (see e. g [15] and [129] for such schemes in the supermatrix context and [70, 108, 71] in the distance context). From the obtained matrices a phylogenetic tree is then constructed using one of the many tree reconstruction techniques.

All three types of approach have their pros and cons, with supermatrices being criticised on, for example, the dependence of the generated supermatrix upon the order in which the missing values are inferred, and the potentially heavy influence of even a small error in the estimation of a missing value on the tree topology, the latter due to a cascading effect such an error might have on other inferred missing values [101]. Criticisms of supertrees include not using primary information, combining trees that have potentially evolved under different evolutionary models into a supertree without properly accounting for this, and not properly taking branch-lengths associated with the input trees into account (see [165] for an exception to this and [99] for a recent comparison of supertree methods). Finally, criticisms of distances include losing valuable phylogenetic information between, for example, two DNA sequences by representing the observed difference by a single number. Nonetheless, distance-based tree reconstruction methods are known to provide quick but rough snapshots of the evolutionary relationships contained within a dataset making them particularly attractive for large datasets. In addition to providing evolutionary insights in their own right their attraction also lies in the provision of a potentially good starting/guide tree for more sophisticated, but computationally intensive, methods such as Maximum Likelihood and Bayesian Inference.

For popular distance methods such as Neighbor Joining [139], and BioNJ [65] (in the unrooted case) and UPGMA [149] (in the rooted case) to be applicable, however, the distance on the combined taxa set X must be complete. As we saw in Section 4.3.3, it is possible to reconstruct trees from partial distance information, but the existing algorithms deal only with the problem of existence of a tree that fits the partial distance, they say nothing about the uniqueness of the solution. In this chapter we focus on the problem of finding a unique equidistant X-tree for some subset of the given distances.

From a biological point of view, equidistant X-trees are commonly constructed when a molecular clock can be assumed for the evolution of the taxa of interest. Although molecular clocks have been criticised strongly over the years (see [7, 144]), widely-used software packages such as BEAST rely heavily on the concept [16]. Intriguingly, this popularity may be linked in part to the recent emergence of NGS datasets, including many consistent with a molecular clock, particularly those at the sub-species level. Indeed, examples of studies where the molecular clock assumption has been satisfied include population studies where they helped to understand the genetic diversity of germplasms [167], palaeontological studies where they helped to estimate divergence dates or shed light into effects that climate change and other global factors might have had on diversification

[162], and phylogeographic studies [28] (see [161] for more on these examples and [79] where so called symbolic ultrametric trees were used in orthology detection).

In the form of the Lasso approach, we propose a novel method for (super)tree reconstruction from partial distances, that is, some of the distance values between taxa are missing. Contrary to the methods alluded to above, Lasso is not imputing-based and is similar in spirit to the supermatrix approach introduced in [118] and the veto-supertree approach proposed in [145] in that not every taxon in the combined taxa set is guaranteed to be a leaf in the resulting tree. It essentially works by trying to detect a treelike signal on as many taxa as possible from the available distances and then reconstructs the *unique* equidistant tree on those taxa. Like UPGMA, Lasso is also an iterative process in the sense that it begins with a partial (or complete) distance D on some taxa set X with a graph G of $|X|$ isolated vertices, each of which is labelled by an element in X. In each repetition step, the distance on a smaller taxa set is recomputed and, in a bottom up manner, an equidistant tree on X is reconstructed. In contrast to UPGMA, Lasso looks to replace a certain type of edge-weighted clique in a canonical graph theoretical representation of the given distances by a composite vertex, rather than only an edge in that graph as UPGMA does. In addition, the distances between a newly created composite vertex and any other vertices is calculated by some kind of consensus rather than merely using average linkage as in the case of UPGMA. These two differences ensure that Lasso enjoys several desirable properties such as *consistency* whereby we mean that the equidistant tree T returned by Lasso is the unique tree which, for any two taxa x and y in the returned lasso, the given distance $d(x, y)$ is equal to the distance between them in T.

5.2 The Lasso algorithm

In this section, we present an outline of the Lasso algorithm. The algorithm is similar in spirit to UPGMA (as shown in Section 4.3.2). Like UPGMA, the Lasso approach is to construct the tree from the bottom up in a repetitive fashion where each repetition consists of a reduction step and a construction step. However, and contrary to UPGMA, Lasso takes as input a partial distance on X (which can of course also be complete). The complete algorithm outline is shown in Algorithm 9. In the following sections we outline the steps in detail.

5.2.1 Method outline

Given a partial distance D on some set of taxa X, let $\mathcal{L}_D$ be the set comprising all pairs of X for which the distance under D is known.

Lasso aims to find a subset $Y \subseteq X$ of taxa and subset $\mathcal{L}' \subseteq \mathcal{L}_D$, both as large as possible, so that the equidistant tree returned by it is uniquely determined by the available distances between pairs in $\mathcal{L}'$ with regards to topology and edge-weighting. In other words, Lasso finds an equidistant Y-tree (T, ω) such that the set $\mathcal{L}'$ is a strong lasso for T and $D_{(T,\omega)}(x, y) = D(x, y)$ holds for all cords $xy \in \mathcal{L}'$.

To do this, $\mathcal{L}_D$ is viewed as an edge-weighted graph Γ_D^ω, or Γ^ω for short. The (unweighted) underlying graph $\Gamma(\mathcal{L}_D)$ of Γ^ω has vertex set X and edge set $\mathcal{L}_D$. The edge-weighting of Γ^ω associates to every edge xy of Γ^ω the distance $D(x, y)$. Note that in the case where D is a complete distance on X, the graph $\Gamma(\mathcal{L}_D)$ is complete. To preserve the given taxa set X, we put $X^r = X$.

For ease of presentation, assume from now on that we have already carried out $q \geq 0$ repetitions and that C is the selected *connected component* of Γ^ω, that is, one of the connected graphs that make up Γ^ω. Note that for $q = 0$ we may assume that C is Γ^ω itself as connectedness of the graph $\Gamma(\mathcal{L}_D)$ and thus of Γ^ω is a necessary condition for a Z-tree to be topologically lassoed by a set of cords on some non-empty set Z [81]. However it should be noted that Γ^ω may become disconnected during successive repetitions. In that case, we exploit the fact that, as we will see below, in each repetition step an equidistant tree is grown from (hopefully all) the vertices of a connected component of the graph Γ^ω generated in the previous step. Put differently, we choose a connected component of Γ^ω such that, over all connected components of Γ^ω, the leaf set of the tree grown from it is as large as possible (where we break ties randomly). Other methods of component choice are conceivable though, such as identifying that which possesses the most cords of $\mathcal{L}_D$. Alternatively, it may be preferable to run Lasso for each connected component of Γ^ω separately, in which case prior biological knowledge might be used to join up the returned equidistant trees.

Note that Lasso terminates when the selected connected component consists of just one vertex. Also note that to help mitigate against a poor choice of a connected component which might yield an equidistant tree on a small number of taxa of X, Lasso returns the tree that connects the most taxa in X (and its associated strong lasso) found over p independent runs, where p is a user defined parameter that is currently set to ten.

To simplify the description of the remaining details of the q-th repetition step, let m denote the minimal edge weight over all edges of C. Then C is transformed into an unweighted graph C_m in which first all edges except those with weight m have been removed and then the weights of the remaining edges are ignored. Lasso now chooses a connected component S_m of C_m and a *suitable* clique of S_m (see below for details) and grows an equidistant tree using the vertex set of that clique. To make this more precise define for any equidistant tree (T', ω') with leaf set Z the *height* of T' to be $D_{(T',\omega')}(x, y)/2$ for any two elements x and y of Z for which the path joining them crosses the root of T'. Let G be a graph consisting of $|X|$ isolated vertices each of which is labelled by an element in X. For the purpose of growing a tree it will be useful to view each of them as an equidistant tree with height zero.

Now, let K_m denote a suitable clique of S_m found by Lasso (where we break ties randomly). Then to obtain a new distance D_m on a smaller set X_m which, for example, ensures that Lasso terminates, we first remove all vertices in S_m from X^r and then add a new vertex u_m to obtain X_m. Next, we define D_m to be the distance on X_m that assigns to any x and y contained in X_m the value $D(x, y)$ if x and y in $X_m - \{u_m\}$, zero if $x = y = u_m$ and the value $D^*(x, y)$ if either $x = u_m$ or $y = u_m$, where D^* is a distance such as the one described below in the section on recomputing the distance D_m.

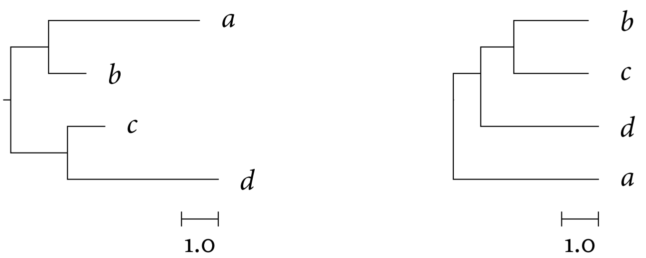
To find the equidistant tree (U_m, ω_m) that LASSO grows from X in this repetition step, let l denote the size of the vertex set of K_m and let $(T_1, \omega_1), \dots, (T_l, \omega_l)$ denote the equidistant trees with leaves in X^r found in the previous repetition steps such that the vertex set of K_m comprises of the roots ρ_i of $T_i, 1 \le i \le l$. Then to obtain U_m, we first add a new vertex to G labelled u_m and then join every root $\rho_i, 1 \le i \le l$, via an edge with u_m making U_m a tree with root u_m and leaves contained in X.

To obtain the equidistant edge-weighting ω_m for U_m, assume for all $i \in \{1, \dots, l\}$ that the height h_i of the tree (T_i, ω_i) was computed in one of the previous repetition steps. Note that, by definition, there must exist leaves u and v with distance $D(u, v) = m$ such that u_m lies on the path from u to v in U_m. Then we define ω_m to be the map that assigns to every edge e of U_m that is also contained in some tree $T_i, 1 \le i \le l$, its weight under ω_i and the weight $m/2 - h_i$ if e contains the root $\rho_i, 1 \le i \le l$. Since for all $i \in \{1, \dots, l\}$ the trees (T_i, ω_i) are equidistant, it is straightforward to see that ω_m is an equidistant edge-weighting for U_m and that the height of the tree (U_m, ω_m) is $m/2$.

To complete the repetition step, we replace X^r by X_m and D by D_m, and return to finding a connected component for the new graph Γ^w for D. Once the aforementioned termination criterion is satisfied, the found equidistant tree and its strong lasso is saved and the next run is started. LASSO stops once all p runs have been completed and returns the equidistant tree and its strong lasso, as described above.

5.2.2 Suitable cliques

Central to LASSO is finding a suitable clique in the graph Γ_D^ω where D was constructed in the previous step of the repetition (or of the input distance if the current step is the start of the repetition). Note that these cliques correspond to the complete graphs of theorem 16 and give us one internal vertex. In the case where the tree is binary these cliques are trivial (consisting of only one edge). To find a suitable clique, let m be the minimal edge-weight in the graph C and S_m a connected component with minimal edge-weight chosen as above. Exploiting again the fact that the new element u_m constructed in the current repetition



(i) Original tree.

(ii) Tree constructed by UPGMA.

Figure 5.1: UPGMA fails to reconstruct the correct tree if the inputted distance is not ultrametric. Here the true (non-equidistant) tree is shown in (i) and the tree constructed by UPGMA in (ii).

step can be thought of as the equidistant tree (U_m, ω_m) whose leaf set is contained in X, we say that a clique in K_m is *suitable* if, over all cliques of S_m, the number of taxa of X it contains is as high as possible.

Since the problem of finding such a clique further requires deciding whether a clique of a given size K or more exists in a graph, and this is a well-known NP-complete problem [62], we use a heuristic for this. More precisely, we start with a randomly chosen edge e in S_m. Note that e is clearly a clique. Denoting that clique by K_e, and its vertices by x and y, we check for all remaining vertices z of S_m if they are adjacent with every vertex of K_e or not. In the former case we update K_e by adding z to its vertex set and all edges of the form $\{c, z\}$ to its edge set where $c \in K_e$, and in the latter case we discard z. We continue in this fashion until we cannot enlarge K_e any further, in which case we stop and save the found clique. To again mitigate against a poor choice, we repeat this process k-times (ignoring edges that are chosen more than once) where k is a user-defined parameter that is currently set to ten. The clique that, over all found cliques, has the largest number of leaves is the clique that we take as the suitable clique.

5.2.3 Recomputing the distance D_m

There are several possible choices for recomputing the distance D_m. Continuing with the notation introduced above we need to define for all vertices a in C but not in S_m the distance $D^*(u_m, a)$ by somehow combining the distances $D(v, a)$ for all v in K_m. In the straightforward case where the inputted partial distance came from an ultrametric, the distances $D(v, a)$ for all $a \in C$ will be equal for all $v \in K_m$. In this case we would simply set $D^*(u_m, a)$ to $D(v, a)$ for all a. However, in practice the distances $D(v, a)$ for some a may not be equal. There are two reasons for this: either the partial distance does not come from an ultrametric at all, or the data from which we derived the partial distance information was subject to noise (as we would expect from biological data).

To illustrate this situation we consider how UPGMA behaves. Figure 5.1 (i) shows an edge-weighted X-tree (T, ω) with $X = \{a, \dots, d\}$ which is not equidistant, therefore the induced distance $D_{(T,\omega)}$ on X is not an ultrametric. If this distance is used as input to UPGMA we get the equidistant X-tree (T', ω') shown in Figure 5.1 (ii). The induced distance on the constructed tree $D_{(T',\omega')}$ is correct only between b and c. To see why recall that UPGMA agglomerates the nearest two clusters in each iteration, in the first iteration this is $\{b\}$ and $\{c\}$ which will come together to form a cherry with the correct height of $D_{(T,\omega)}(b, c)/2$. Next the algorithm must calculate the distance between the newly formed cluster and every other cluster using average-linkage. The distance between $\{b, c\}$ and $\{a\}$, for example is calculated from the distances between $\{b\}$ and $\{a\}$ and $\{c\}$ and $\{a\}$. But observe that these two distances are not equal. This is our indication that the inputted distance was not ultrametric.

Returning to Lasso, it is now clear that to provide the consistency property we cannot simply use average-linkage in the case where the distances $D(v, a)$ are not equal. Instead we must discard some of the input distances such that $D(v, a)$ is equal for all $v \in K_m$ and all $a \in C - S_m$. This reduces the size of our outputted lasso. To decide which of the distances to discard the obvious way is for each $a \in C - S_m$ to take the mode of the distances $D(v, a)$ for all $v \in K_m$ and discard those distances not equal to the mode. This ensures that the consistency property is satisfied while using as many distances as possible. The fact that the algorithm has not used every distance in the input can be used as a sign that the inputted distance was not ultrametric. In the worst case, the lasso returned by the algorithm will be only a minimal topological lasso even if many more distances were inputted. Indeed, this is the case if we try the example of Figure 5.1 using the above procedure to calculate D^*.

However, this does not take into account the fact that real data is noisy. In practice we can rarely expect to find equality, but this does not necessarily mean that the real underlying tree is not equidistant. To enable tolerance to noise we try to find for each $a \in C - S_m$ a cluster of distances in $D(v, a)$ over all $v \in K_m$. The distances in the cluster should be similar to within some noise threshold. We then discard distances not in the cluster and set $D^*(u_m, a)$ to the mean of the distances in the cluster.

To find a single cluster we can use a simple iterative approach. Let $S \subset \mathbb{R}_{\geq 0}$ be our set of distance values and $\bar{s}$ the mean of all values in S. We find an $s \in S$ such that $|s - \bar{s}|$ is maximised and remove this value if $\sigma = |s - \bar{s}|/\bar{s}$ is greater than some threshold ζ. We repeatedly remove values from S until σ is below the noise threshold. We are left with our cluster S. We have found that a good value for ζ is 0.1.

The overall algorithm is given in as Algorithm 9. The runtime complexity of the algorithm is $O(|\mathcal{L}_D|^2)$ since, in the worst case, we replace a clique consisting of only one edge in each iteration and must perform a linear search on the edges of Γ^ω to find the minimum edge-weight in each iteration.

5.2.4 An example

To illustrate the Lasso approach assume that $X = \{a, \ldots, e\}$ is a taxa set and that D is a partial distance on X given in terms of the edge-weights of the graph Γ_D^ω presented in Figure 5.2 (i).

Then in the first repetition step (with $q = 0$) the minimal edge-weight is 2 and the connected component C chosen by Lasso is Γ_D^ω itself as that graph is connected. The subgraph of C with vertex set $\{a, b, c, d\}$, edge set indicated in bold and edge-weights ignored, is C_2. Note that this graph coincides with the connected component S_2 chosen by Lasso as that graph is again connected. The suitable clique chosen by Lasso is the subgraph with vertex set $\{b, c, d\}$ and the three edges joining them in the form of a triangle in that

Algorithm 9 The Lasso algorithm

Input: Partial distance D on X

Output: A subset $\mathcal{L}'$ of cords of $\mathcal{L}_D$ and an equidistant Y-tree (T,ω) that is strongly lassoed by $\mathcal{L}'$ such that Y and $\mathcal{L}'$ are as large as possible, $Y = \bigcup_{xy\in\mathcal{L}'} xy$ holds, and $D_{(T,\omega)}(x,y) = D(x,y)$, for all $xy \in \mathcal{L}'$.

```
0. Compute $\Gamma^\omega = \Gamma^\omega_D$ and put $q := 0$ and $X^r = X$.
1. Choose a connected component $C$ of $\Gamma^\omega$ such that the leaf set of the equidistant tree grown from it is as large as possible. If $C$ has a single vertex, terminate and return that tree and the found set of cords that strongly lassos it.
2. Put $m := \min_{xy\in\mathcal{L}_D} D(x,y)$ and compute unweighted graph $C_m$.
3. Choose a connected component $S_m$ of $C_m$ and a suitable clique $K_m$ in $S_m$.
4. Using a new vertex $u_m$ and the definition of $D^*$, put $X_m := X^r - S_m \cup \{u_m\}$ and the new distance $D_m$ on $X_m$, respectively.
5. Join $u_m$ with the roots of the equidistant trees whose roots correspond to the vertices of $K_m$ to obtain the tree $U_m$ and define the equidistant edge-weight $\omega_m$ such that the height of $U_m$ is $m/2$.
6. Put $X^r := X_m$, $D := D_m$, $q := q + 1$ and return to step 1.
```

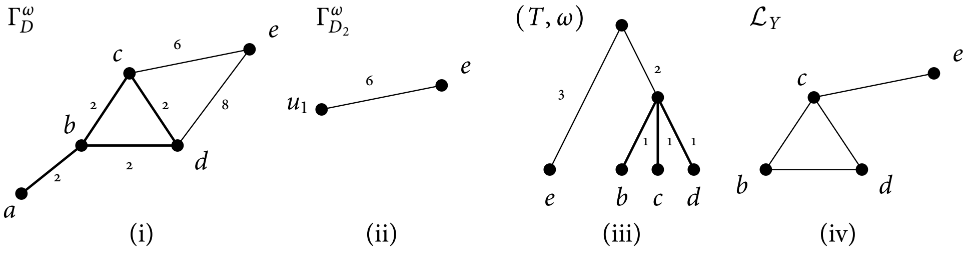


Figure 5.2: For the partial distance D on $X = \{a,\dots,e\}$ as indicated by the edge-weights of the graph Γ^ω_D depicted in (i) we depict in (iii) the equidistant tree (T,ω) returned by Lasso and in (iv) the strong lasso found by Lasso for T. In (ii) we depict updated distance D_2 in the first repetition step of Lasso in terms of the graph $\Gamma^\omega_{D_2}$.

figure. The equidistant tree (U_2, ω_2) grown by LASSO is the subtree of the equidistant tree depicted in Figure 5.2 (iii) with leaf set $\{b, c, d\}$ and edges in bold. Furthermore, the updated distance D_2 on $X_2 = \{u_2, e\}$ is represented in terms of the graph $\Gamma^{\omega}_{D_2}$ displayed in Figure 5.2 (ii) where the tie was broken by deleting the cord de and thus removing the distance $D(d, e)$. Putting $X^r = X_2$ and $D = D_2$ completes the first repetition step. Since Γ^{ω}_{D} is again connected and contains more than one vertex a second repetition step is carried out. Again C is the graph Γ^{ω}_{D} itself. The minimal weight m is six and C_6, S_6 and K_6 all equal Γ^{ω}_{D}. Then $X_6 = \{u_6\}$, $D_6(u_6, u_6) = 0$. The equidistant tree (U_6, ω_6) grown is depicted in Figure 5.2 (iii). To complete the second repetition step we now replace X^r by X_6 and D by D_6. Since the graph $\Gamma^{\omega}_{D_6}$ is connected LASSO picks $\Gamma^{\omega}_{D_6}$ as the connected component. Since $\Gamma^{\omega}_{D_6}$ contains only the vertex u_6, LASSO terminates. For $Y = \{b, c, d, e\}$, we picture in Figure 5.2 (iv) the strong lasso $\mathcal{L}_Y$ found by LASSO for T in terms of the graph $\Gamma(\mathcal{L}_Y)$. It is possible for LASSO to find smaller trees (for example we could have picked the clique with only a and b in the first iteration) but after multiple runs this tree will be found as the largest.

5.3 Results and Discussion

THE LASSO APPROACH enjoys several attractive theoretical features. These include that when given a partial distance D on some taxa set X, the set $\mathcal{L}'$ of cords returned by LASSO (together with the distances for the cords in that set) uniquely determines the topology as well as the edge-weighting of the equidistant X'-tree (T, ω) returned by LASSO, where X' is the vertex set of $\Gamma(\mathcal{L}')$. In particular, if D is such that $\mathcal{L}_D$ is a topological lasso for T then $X = X'$. That is, the leaf set of T is the whole of X. Moreover, if D is in fact a complete distance on X and ultrametric then the distance induced by (T, ω) on X is D.

Due to, for example, missing data it is in general too much to hope for that the available distances in a real biological study correspond to a topological lasso for some equidistant tree. To assess the performance of LASSO as a tree reconstruction approach with regards to this confounding factor, while controlling key aspects of the input data, we carried out a simulation study which is similar in spirit to the one presented in [31]. To gauge the performance of Lasso as a tree reconstruction approach on a real biological dataset, we applied it to a yeast dataset [163] recently developed from a whole genome resequencing study [104]. We further assessed the potential of Lasso as a supertree approach by combining two partially overlapping wheat datasets, developed in distinct studies [132, 142]. We start with outlining our missing data simulation scenario.

5.3.1 Missing data

To better understand how the topology of an equidistant tree affects our ability to reconstruct it from a partial distance, we first generated three binary X-trees, one of which was a balanced tree, a second a caterpillar tree, and the third we generated using the Yule-Harding model. For this, we took the size of X to be 128. For initial unweighted tree simulation, we implemented the approach described in [147, Section 2.5]. Next, we turned each of the resulting trees into an equidistant X-tree. In the balanced tree case we assigned weight one to all edges. In the caterpillar tree case we assigned the difference in height between two adjacent vertices to the weight of the joining edge, and in the Yule-Harding tree case we proceeded as follows. Starting with a Yule-Harding X-tree T, we first assign to every vertex v of T the number $h(v)$ of edges on a longest path from v to a leaf of T below v, where we put $h(v) = 0$ in case v is a leaf. For e an edge of T joining two vertices u and v, we then assign $|h(u) - h(v)|$ as weight to e.

For each of these three equidistant X-trees, we then generated an incomplete distance matrix from the induced (complete) distance matrix by randomly removing a percentage P_{miss} of entries, ensuring that with $\mathcal{L}$ denoting the associated set of cords the $\Gamma(\mathcal{L})$ graph remained connected. More precisely, we generated 500 incomplete distance matrices for each of the three equidistant X-tree types using the values 1%, 5%, 10%, 20% and 30% for P_{miss}. We then used the resulting $3 \times 5 \times 500$ incomplete distance matrices as input to Lasso. Each equidistant X-tree found by Lasso was then compared with the

P_{miss}	mean	min	max	meanc	minc	maxc
0.0	0.000	0.000	0.000	2016.000	2016	2016
1.0	0.000	0.000	0.000	2015.770	2014	2016
5.0	0.011	0.000	0.095	2003.480	1945	2015
10.0	0.029	0.000	0.190	1976.640	1814	2005
20.0	0.099	0.000	0.587	1849.600	475	1944
30.0	0.261	0.000	0.762	1445.220	255	1842

Table 5.1: Normalised Robinson-Foulds distances for the balanced trees and the sizes of the supporting strong lassos.

P_{miss}	mean	min	max	meanc	minc	maxc
0.0	0.000	0.000	0.000	2016.000	2016	2016
1.0	0.000	0.000	0.000	2015.760	2013	2016
5.0	0.005	0.000	0.143	2007.850	1949	2016
10.0	0.024	0.000	0.270	1982.360	1869	2003
20.0	0.110	0.000	0.492	1842.460	672	1954
30.0	0.187	0.000	0.635	1670.370	317	1859

Table 5.2: Normalised Robinson-Foulds distances for the Yule trees and the sizes of the supporting strong lassos.

P_{miss}	mean	min	max	meanc	minc	maxc
0.0	0.000	0.000	0.000	2016.000	2016	2016
1.0	0.000	0.000	0.000	2015.780	2014	2016
5.0	0.000	0.000	0.000	2011.090	2003	2016
10.0	0.020	0.000	1.000	1994.720	1933	2004
20.0	0.040	0.000	1.000	1931.610	1864	1954
30.0	0.110	0.000	1.000	1829.440	1769	1861

Table 5.3: Normalised Robinson-Foulds distances for the caterpillar trees and the sizes of the supporting strong lassos.

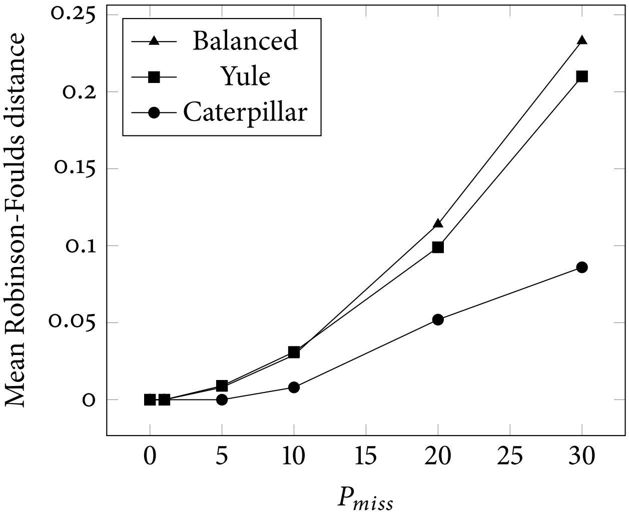


Figure 5.3: For all three equidistant X-tree types, we plot the normalised Robinson-Foulds distance between T and $T'|_Y$.

respective equidistant X-tree (T', ω') used to generate the underlying input matrix. More precisely, for Y denoting the leaf set of a tree (T, ω) returned by LASSO, we computed the Robinson-Foulds distance [135] $D_{RF}(T, T'|_Y)$ between T and the restriction $T'|_Y$ of T' to Y, that is, we counted the number of clusters induced by $T'|_Y$ but not by T and vice versa. For each equidistant X-tree type and percentage P_{miss}, we then averaged the obtained 500 distances using the mean. We summarise our results in Figure 5.3 in terms of the *normalised Robinson-Foulds distance* $D_{RF}(T, T'|_Y)/2(|X|-1)$ between T and $T'|_Y$, that is, we divided $D_{RF}(T, T'|_Y)$ by the maximal Robinson-Foulds distance between two X-trees, which is $2(|X|-1)$. Tables 5.1, 5.2 and 5.3 show some simple statistical measures on the supporting strong lassos. Mean, min and max refer to the Robinson-Foulds distance and meanc, minc and maxc refer to the size of the lasso.

We observed that, independent of the type of equidistant tree considered, the Robinson-Foulds distance between $T'|_Y$ and T increases as the number of distance values decreases, which is as expected. Having said that, out of all three X-tree types equidistant caterpillar trees were reconstructed most accurately with a mean normalised Robinson-Foulds distance below 0.1 even when 30% of the distance values were missing. A potential reason for

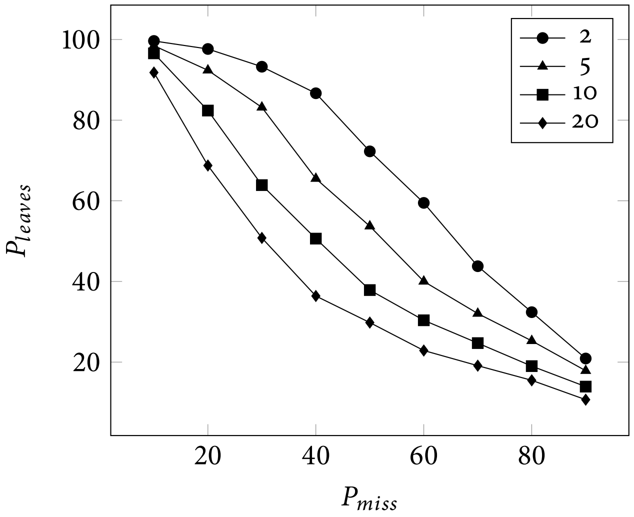


Figure 5.4: For T' an X-tree with $|X| = 100$ and maximum out-degree $k = 2, 5, 10$ and 20 we depict the proportion of X which forms the leaf set of T. – see text for details.

this might be that to correctly reconstruct a internal vertex v of T' the child-edge graph associated to v must be a complete graph (see Section 4.4.2). In a caterpillar tree there is only one cherry, all other internal vertices have a child that is a tree. So the likelihood that the child-edge graph associated to an internal vertex is a complete graph is high as an edge in that vertex's child edge graph tends to be supported by many leaf pairs of T implying that Lasso correctly reconstructs v. On the other hand, a balanced tree has the highest number of cherries. Thus, the likelihood that the child-edge graph associated to such vertices is not a complete graph increases as the number of missing distance values grows, implying that T may be very different from $T'|_Y$. Given that it is well-known that trees generated under the Yule-Harding model tend to be highly balanced (see e. g. [147, Section 2.5]) it is unsurprising to see that equidistant Yule-Harding trees and equidistant balanced trees exhibit a similar behaviour. Interestingly, equidistant Yule-Harding trees were reconstructed slightly more accurately than equidistant balanced trees overall, presumably due to small departures from a purely balanced state.

Figure 5.3 suggests that for low quantities of missing distances, Lasso is very good at exploiting redundancy in a given distance matrix to correctly reconstruct the underlying

Algorithm 10 Random tree generation

Input: A set X and an integer k.
Output: An equidistant X-tree with maximum out-degree k.

```
1. Choose an integer $p$ in $\{2, \dots, \min(|X|, k)\}$ and some subset $C$ of $X$ of size $p$.
2. Construct a tree $T$ with root $\rho_T$ by attaching each element $c$ in $C$ via an edge to $\rho_T$
   and setting the weight of that edge to $1 + \max_{x \in X} height(x) - height(c)$.
3. Put $X := (X - C) \cup \{\rho_T\}$.
4. If $|X| > 1$ go to step 1, otherwise return $T$ and its edge-weighting.
```

equidistant X-tree, independent of the tree type. To better understand how much this observation is dependent on the fact that the starting X-trees were all binary, we also investigated the influence of the maximal vertex degree k of such a tree on Lasso's performance. To this end, we generated 500 random equidistant X-trees (T', ω') with maximum vertex out-degree k as described in Algorithm 10. The values for k that we used in this study were $k = 2, 5, 10, 20$ and the size of X was always 100. We summarise our results in Figure 5.4 in terms of the average percentage P_{leaves} of the elements in X that are also present in the leaf set of the equidistant tree (T, ω) returned by Lasso.

As expected, P_{leaves} is very high (above 80%) for all values of k if the number of missing distances is not too large ($P_{miss} \leq 10\%$) which is encouraging from a supertree perspective as the out-degree of a vertex can be comparatively high in such trees. However with an increasing number of missing distances, equidistant X-trees with a lower maximal degree seem to fare better overall. More precisely, in the case $k = 2$ the equidistant tree returned by Lasso still contains 80% of the leaves of T' even if 40% of the distance values are missing. To obtain a similar result for $k = 20$ only around 10% of the distance values are allowed to be missing from a distance matrix. A potential reason for this discrepancy might be that the more distances are missing from a distance matrix the more likely it is for the child-edge graph of a high out-degree vertex not to be a complete graph and thus to not be correctly reconstructed by Lasso.

5.3.2 A yeast dataset

To test Lasso on a real biological dataset, again as a tree reconstruction approach in the face of missing data, we applied it to a distance matrix generated for the analysis of several intra-specific strains of yeast. In [163] the authors identified both fully and partially resolved single nucleotide polymorphisms (SNPs and pSNPS) within the ribosomal DNA (rDNA) tandem arrays of 26 strains of the wild yeast *Saccharomyces paradoxus*. A distance matrix was constructed from the resulting allele frequency dataset using the Cavalli-Sforza and Edwards Chord distance measure [25]. A phylogenetic tree was estimated using Neighbor Joining. The tree was rooted by analysing rDNA variation in S288c, the type strain of the closely related baker's yeast *Saccharomyces cerevisiae*. We note that the rooted tree built with either UPGMA or Lasso is very similar to the authors' tree suggesting that the tree underlying the dataset is indeed equidistant. From the distance D on the strains induced by the Lasso-tree we then randomly removed 10% of the distance values ensuring that (i) whenever we removed for two strains x and y the distance $D(x, y)$ we also removed the distance $D(y, x)$, that (ii) values of the form $D(x, x)$ where x denotes a strain did not count towards the removed 10%, and that (iii) the graph $\Gamma(\mathcal{L}_D)$ remained connected. We present the equidistant tree returned by Lasso in Figure 5.5. Note that this tree contains all 26 input taxa. Furthermore, its topology is highly similar to that produced, on the full distance matrix, by [163]. Most importantly, the groupings of the American, Far Eastern and European strains are preserved, as is the separation of the UK and non-UK derived strains within the European group. Furthermore, the putative European/Far Eastern hybrid strains N_17 and N_45 are located within the tree at positions consistent with such an evolutionary history. While some minor changes in topology are seen within the European and American groups, the relationships within the Far Eastern group are wholly preserved once 10% of distances have been removed.

Since Lasso's ability to reconstruct an equidistant tree from a partial distance depends on both which distances are missing and the random decisions made by the algorithm

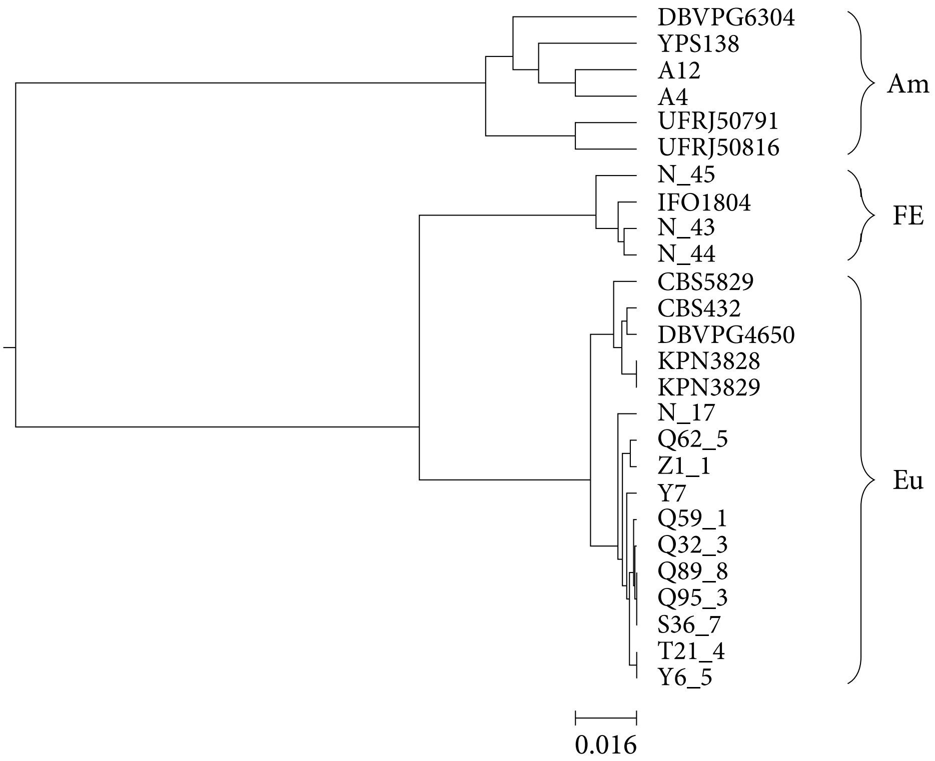


Figure 5.5: An equidistant tree returned by Lasso from the yeast dataset with 10% of the distances randomly removed. The sixteen European strains are denoted by the label "Eu", the four Far Eastern strains by "FE" and the six American strains by "Am". The uppermost five European strains (CBS5829 to KPN3829), together with N_17, derive from outside the UK, with the remaining ten European strains having been isolated within the UK.

to break ties, we also constructed a consensus tree from the equidistant trees returned by Lasso (see Figure 5.6). For this, we used an approach similar to bootstrapping and indicate in terms of numbers assigned to each internal vertex, except the root, how often each cluster induced by such a vertex is displayed by an equidistant tree returned by Lasso. More precisely we generated 100 partial distances with 10% of the distances removed at random as described above. We then ran Lasso on each partial distance resulting in a total of 100 equidistant trees each supported, on average, by a strong lasso with 205 cords (out of the possible 293). The resulting trees were then used as input to the Consense program [56] with default settings to build a consensus tree using the "majority rule (extended)" option. This tree is again highly consistent with the full distance matrix tree, differing only

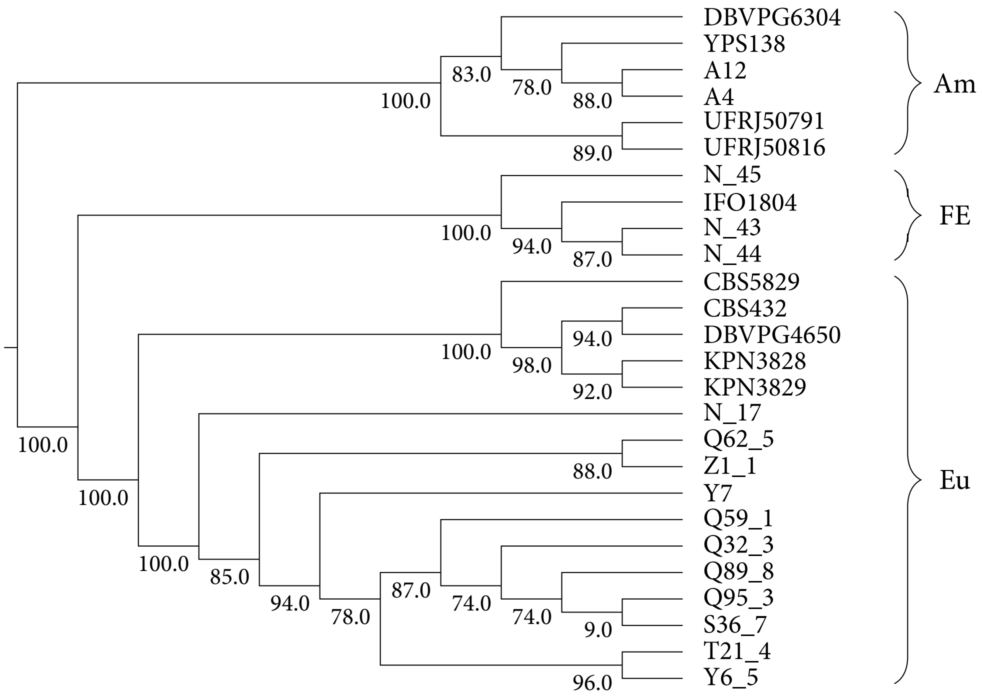


Figure 5.6: Consensus tree built from 100 runs of Lasso on matrices with 10% of the distances randomly removed. The number next to a vertex shows the number of times the cluster induced by that vertex appeared in the input of Consense. The length of an edge is of no relevance.

from Figure 5.5 in the relationship between the three European strains Q89_8, Q95_3 and S36_7. Noticeably, the support for the bifurcation of the latter two strains is the only one in Figure 5.6 less than 74.

5.3.3 A wheat dataset

Next, we analysed two partially overlapping wheat genetic marker datasets in order to evaluate the potential of Lasso as a supertree approach. The first dataset (which we will refer to as dataset *A*) consisted of 57 NBS (Nucleotide Binding Site) markers scored over 411 accessions, a subset of a dataset developed within the GEDIFLUX EU Framework V project [132] to assess genetic diversity over time across four major crops, including wheat. The second dataset (which we will refer to as dataset *B*) consisted of 71 NBS markers scored over 118 accessions, a subset of a study comparing genetic diversity within and between

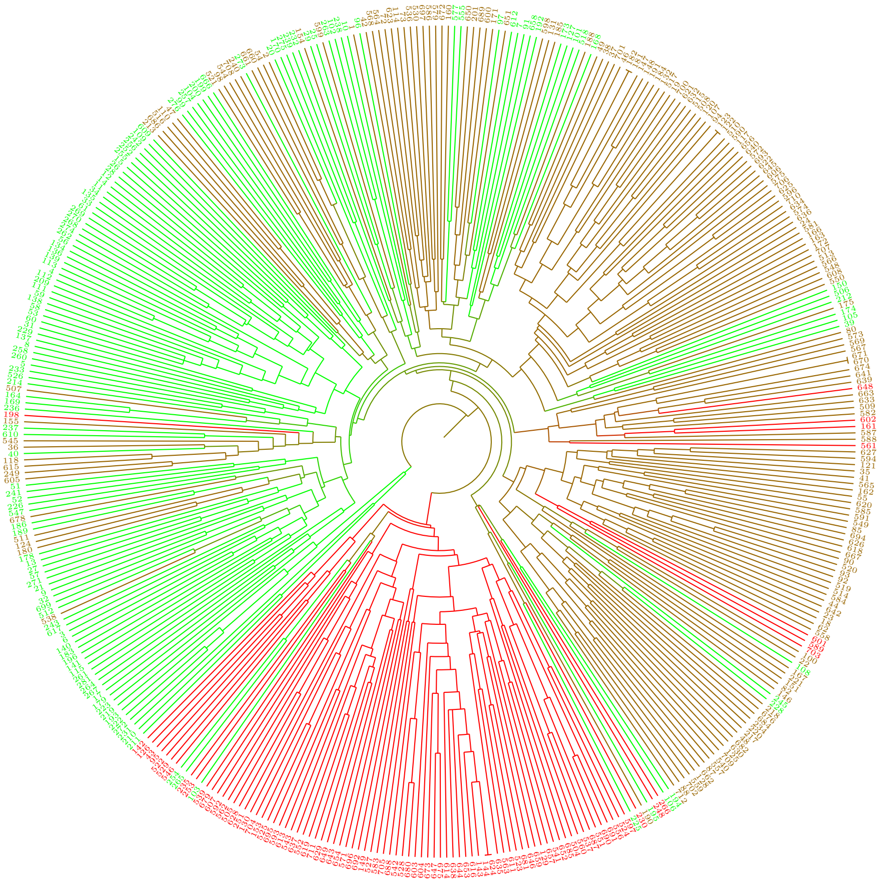


Figure 5.7: The Lasso tree for wheat dataset A coloured according to the groupings found by ADMIXTURE.

winter wheat accessions from Turkey, Kazakhstan and Europe [142], the latter comprising a small group of the GEDIFLUX wheat accessions. Consequently, the two datasets share a common set of 26 European winter wheat accessions, comprising 6.3% and 22.0% of their accessions respectively. We will refer to this shared dataset as dataset C.

Equidistant trees were estimated for datasets A and B separately, using the Modified Rogers measure [133] to calculate a distance matrix, followed by tree construction with Lasso. For convenience, we denote the two distance matrices as d_A and d_B, where the index indicates the dataset to which they refer. The resulting equidistant trees were found to be supported by 77,577 (out of 84,255) and 6,844 (out of 6,903) distances from d_A and d_B respectively, and are shown in Figures 5.7 and 5.8. We assessed the individual

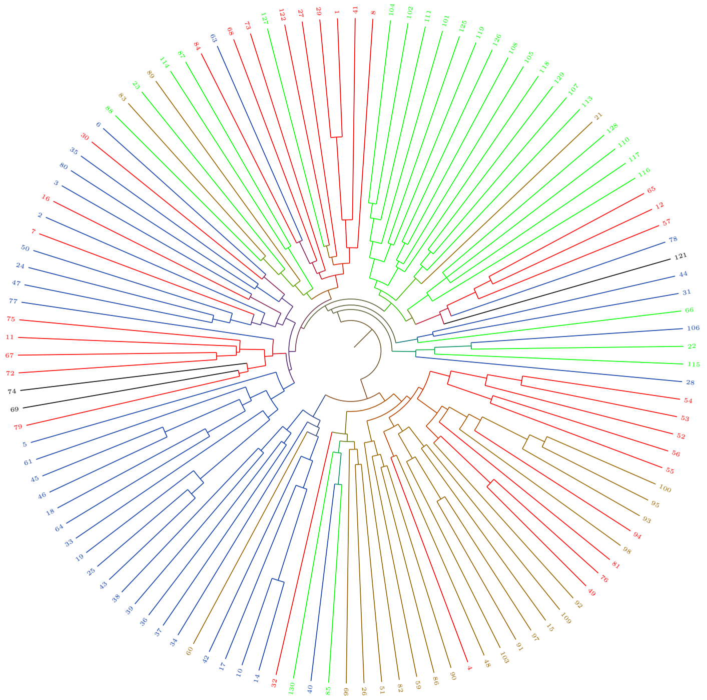


Figure 5.8: The LASSO tree for wheat dataset B coloured according to the groupings found by ADMIXTURE.

LASSO trees according to population group data for the two datasets. In the original analysis of dataset B [142], the model-based clustering method STRUCTURE [128] was carried out to estimate the number of founder populations underlying the dataset and the genetic contribution of each population to each accession. We coloured branches of the LASSO tree in Figure 5.8 such that the colour of each branch corresponded to the main population group to which the relevant accession belonged. For dataset A, we conducted our own population structure analysis, here using the ADMIXTURE method [2] with default parameter values. ADMIXTURE uses an identical genetic model to STRUCTURE, but a different computational approach to optimise population parameters, rendering it considerably faster to run. The LASSO tree for dataset A, coloured according to these groups, is shown in Figure 5.7. For both datasets, we see that accessions belonging to the same population group are largely clustered within the equidistant trees. The large sizes of the two Lassos (encompassing 92.1% and 99.1% of distances within d_A and d_B respectively), together with the consistency of population grouping across the equidistant trees, strongly suggest the suitability of the Lasso approach in determining the genetic relationships between accessions within both of these datasets. Furthermore, we note the previous use of the UPGMA method to analyse NBS markers for wheat accessions [109].

To obtain a (partial) distance D on the combined dataset of $411 + 118 - 26 = 503$ accessions we proceeded as follows. If x and y are accessions such that one of them is contained in dataset $A \backslash C$ and the other in A then we put $D(x, y) = d_A(x, y)$. Similarly, if one of them is contained in dataset $B \backslash C$ and the other in B then we put $D(x, y) = d_B(x, y)$. For the remaining case that both accessions are contained in the overlap we took the mean, that is, we put $D(x, y) = d_A(x, y) + d_B(x, y))/2$ where x and y are the accessions under consideration. To mitigate against the fact that for some accessions in C the distance values $d_A(x, y)$ and $d_B(x, y)$ correlate more strongly than for others we used the ratio $d_A(x, y)/d_B(x, y)$ to identify outliers, which we subsequently removed from the analysis. For this, we employed the distribution of these ratios and defined a distance value to be an outlier if it is more than one interquartile range above the upper quartile or one interquartile range below the lower quartile.

In total, the distance matrix representing D contained 90,814 entries (where we exclude entries of the form $D(x, x)$ and only count entries of the form $D(x, y)$ and $D(y, x)$ once) which equates to 28.1% of the $126, 253$ potential distance values of a distance matrix on 503 taxa missing. We then used this distance as input for Lasso to obtain a supertree T on the combined dataset. We depict that supertree in (Figure 5.9) and remark in passing that the tree contains all 503 input taxa and that the size of the strong lasso returned by Lasso supporting T is 89,642. Put differently, T is the unique equidistant tree that displays correctly 98.7% of the 90,814 distance values for D.

To better understand the relevance of the obtained supertree T we tested it with regards to consistency with the original distances on the two datasets. In addition, we compared its topology against the supertree generated by the modified MinCutSupertree approach [123]. For the former, we performed Mantel tests between the distance matrices derived

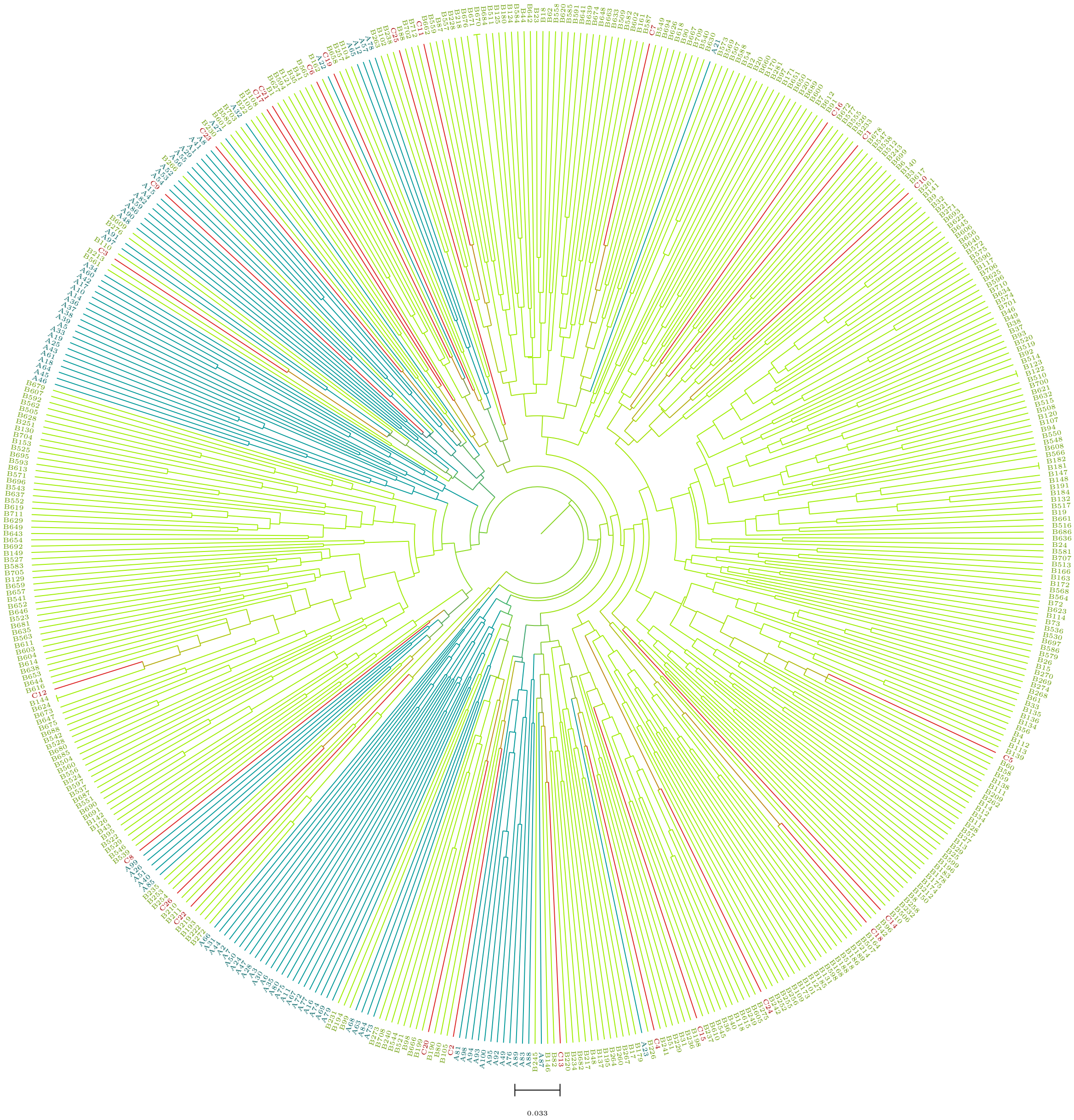


Figure 5.9: The equidistant supertree built by Lasso for the two wheat datasets. Accessions from the GEDIFLUX dataset (*A*) are indicated by green branches, those from the Turkish dataset (*B*) by blue branches, with the 26 accessions found in both datasets (*C*) indicated by red branches. Note that the shared accessions are spread across the supertree and that the tree contains all 503 input taxa.

from d_A and d_B and the distances displayed by T. The tests showed a positive correlation of 0.57 and 0.47, respectively, with p-values for both being 0.0009990. These results indicate that T displayed relationships between accessions within the two datasets appropriately, including the overlapping accessions.

For the latter, we employed again the Robinson-Foulds distance. More precisely, we used the equidistant trees T_A and T_B generated by LASSO from the datasets A and B, respectively, to generate a supertree S using the modified MINCUTSUPERTREE approach. This tree we then restricted to each of the data sets A and B resulting in the trees $S|_A$ and $S|_B$, respectively. Next, we restricted the topology t of T to the accessions in A and in B, respectively, resulting in the trees $t|_A$ and $t|_B$ (where the index indicates the data set). For these four trees we computed the Robinson-Foulds distances between them and found that $D_{RF}(T_A, S|_A) = 96 > 79 = D_{RF}(T_A, t|_A)$ and that $D_{RF}(T_B, S|_B) = 384 > 322 = D_{RF}(T_B, t|_B)$. This suggests that the topology of the supertree returned by LASSO is closer to that of the original trees on the two datasets than the supertree constructed by the modified MINCUTSUPERTREE approach, thus highlighting LASSOS's potential as a supertree method.

5.4 Conclusion

IN THIS CHAPTER, we have proposed the novel LASSO approach both for distance-based phylogenetic tree reconstruction and supertree construction. LASSO is similar in spirit to UPGMA but takes as input partial distances and aims to reconstruct a unique (in a well-defined sense), equidistant tree by exploiting redundancy in a given distance matrix rather than trying to estimate missing distance values as do approaches such as [31]. Given that equidistant trees can be exploited as starting trees within a search of tree space, for sophisticated phylogenetic methods such as Maximum Likelihood and Bayesian Inference [23, 16], LASSO might provide a quick and promising way to extend these methods to data sets for which only partial distance information is available. Such datasets might arise when, for example, confidence in a distance value is low resulting in an exclusion of that distance (and thus the taxa involved) from an analysis, or when combining disparate but overlapping datasets in a supertree context.

We assessed the performance of LASSO by means of artificial and real biological data. For the former we considered three different types of binary equidistant trees and found that, independent of the tree type, LASSO showed great promise when up to approximately 10% of distance values were missing. For higher percentages of missing distance values we found that the type of equidistant tree started to affect our ability to accurately reconstruct it. More precisely, the equidistant caterpillar tree was recovered most accurately under our simulation scenario and the equidistant balanced tree the least accurately, with the equidistant Yule-Harding tree faring slightly better than the balanced tree. We also found that LASSO showed great promise when the (unknown) equidistant tree underlying the given partial distance did not possess vertices of too high a degree if a high proportion of distance values were missing. Put another way, even with 10% of distance values missing for an underlying equidistant tree with vertices of maximal out-degree 10, LASSO was still able to return a tree on more than 80% of the original taxa.

We also assessed the performance of LASSO on two real biological datasets. In the first of these studies, a yeast dataset originally developed and analysed in [163] was successfully reconstructed by LASSO, even with 10% of distance values removed at random. In the second study, LASSO was used to construct a supertree of two partially overlapping wheat NBS marker datasets [132, 142]. Subsequent statistical tests showed both that the LASSO supertree successfully displayed relationships found with the two original datasets and that it was more consistent with the two underlying trees than a supertree constructed using the rival modified MINCUTSUPERTREE approach. Collectively, these studies suggest that LASSO is potentially a highly useful method for both tree reconstruction and supertree construction on real datasets.

It is interesting to speculate that the strong performance of LASSO in a supertree context owes part of its success to its ability to reject shared distances that are not highly correlated. Indeed, when we repeated our Mantel tests comparing the equidistant supertree derived from the full combined dataset (that is, not rejecting any shared distance values) to the two separate distance matrices (A and B), the correlations were found to be 0.47 (p = 0.0009990) for both datasets. Although this performance is almost identical for dataset B, we see that removing certain shared distances leads to a highly improved result for dataset A, which we earlier noted possessed a lower proportion of distance values within the Lasso. In future, an investigation of methods to combine datasets for supertree construction would be highly valuable, to see whether a further improvement could be obtained.

Additional future work might include updating the distance in each repetition step in a different way. An alternative might be to simply throw out outliers and remain much more tolerant to non-ultrametric input. Although as we become more tolerant to this we are less able to claim that we are satisfying the consistency property. We decided to only remove distances to avoid the problems that come with introducing distances that exist with other methods (as in Section 4.3.3), but it might also be worth considering introducing distances under certain circumstances. For example, we may consider adding one distance if it would allow us to grow our suitable clique further. Also it might be interesting to investigate the Lasso approach in a *relaxed* molecular clock framework [48].

In summary, we propose the Lasso approach to become a new method within the molecular phylogenetics toolkit. Given its demonstrated potential both in tree reconstruction in the face of missing data, and in supertree construction, we believe it can play an important role within a key project of our generation, uncovering the Tree of Life.

5.5 Acknowledgements

For their assistance in the preparation of the paper underlining this chapter we gratefully acknowledge Lesley Boyd and Muge Sayar-Turet for providing a wheat dataset and Andrei-Alin Popescu for help with the ADMIXTURE analysis and testing of software.

Chapter 6

Distinguished Minimal Topological Lassos

This chapter is based on the following paper of which I was a minor author:

K.T. Huber and G. Kettleborough. Distinguished minimal topological lassos. Submitted, 2014

∼

I was involved in developing the idea of a special type of minimal topological lasso whose $\Gamma(\mathcal{L})$ graph is a block graph, the development of much of the terminology, lemma 8, propositions 1 and 2, theorem 17 and all of the examples used throughout. This theory was intended to be used by a new algorithm, however the development of the algorithm was unsuccessful.

6.1 Introduction

6.1.1 Summary

In this chapter we focus on a type of lasso called a minimal topological lasso, that is a topological lasso from which no cord can be removed such that the set of cords remains a topological lasso. We show that any set-inclusion minimal topological lasso for such a tree T can be transformed into a "distinguished" minimal topological lasso $\mathcal{L}$ for T, that is, the graph $(X, \mathcal{L})$ is a claw-free block graph. Furthermore, we characterise such lassos in terms of the novel concept of a cluster marker map for T and present results concerning the heritability of such lassos in the context of the subtree and supertree problems.

6.1.2 Minimal topological lassos and the graph $\Gamma(\mathcal{L})$

In this section, we introduce the extra terminology required for this chapter and establish some initial results. Assume throughout that X is a finite set with at least 3 elements and that, unless stated otherwise, all sets $\mathcal{L}$ of cords of X considered satisfy the property that $X = \bigcup \mathcal{L}$. We say that $\mathcal{L}$ is a *(set-inclusion) minimal topological lasso for* T if $\mathcal{L}$ is a topological lasso for T but no cord $c \in \mathcal{L}$ can be removed from $\mathcal{L}$ such that $\mathcal{L} - \{c\}$ is still a topological lasso for T.

We denote the set of leaves of T that are also descendants of v by $L_T(v)$. If v is a leaf of T then we put $L_T(v) := \{v\}$. If there is no ambiguity as to which X-tree T we are referring to then, for all $v \in V(T)$, we will write $L(v)$ rather than $L_T(v)$ and $ch(v)$ rather than $ch_T(v)$.

To facilitate the discussion of lassos we will very often refer to a graph called $\Gamma(\mathcal{L})$. For a set of cords $\mathcal{L}$ of X the graph $\Gamma(\mathcal{L})$ has vertex set X and an edge between distinct elements x and y in X whenever $xy \in \mathcal{L}$. If there is no danger of confusion, we denote an edge $\{a, b\}$ of $\Gamma(\mathcal{L})$ by ab rather than $\{a, b\}$.

To illustrate these definitions, let $X = \{a, \cdots, f\}$ and let $\mathcal{L}$ be the set of cords such that $\Gamma(\mathcal{L})$ is the graph depicted in Figure 6.1 (i). It is easy to see that the X-trees depicted in

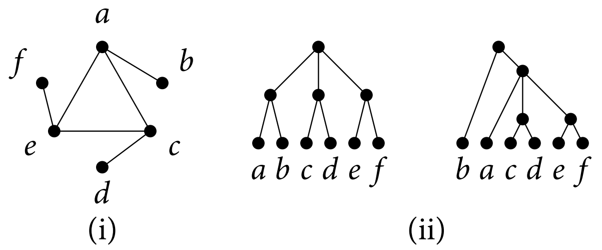


Figure 6.1: (i) The graph $\Gamma(\mathcal{L})$ with vertex set $X = \{a, b, \ldots, f\}$ for the set $\mathcal{L} = \{ab, cd, ef, ac, ce, ea\}$. (ii) Two non-equivalent X-trees T and T' that are both topologically lassoed by $\mathcal{L}$. In fact, $\mathcal{L}$ is a minimal topological lasso for either one of them.

Figure 6.1 (ii) are topologically lassoed by $\mathcal{L}$. In fact, $\mathcal{L}$ is a minimal topological lasso for both of them.

Denoting for an X-tree T, a topological lasso $\mathcal{L}$ for T, and an interior vertex $v \in \mathring{V}(T)$ the set of all cords $ab \in \mathcal{L}$ for which $v = lca_T(a, b)$ holds by $\mathcal{A}(v)$, Theorem 16 readily implies $|\mathcal{A}(v)| \geq \binom{|ch(v)|}{2}$. The next observation is almost trivial yet central to this chapter and concerns the special case that $\mathcal{L}$ is a minimal topological lasso for T. To able to state it, we denote for an interior vertex $v \in \mathring{V}(T)$ and a child edge $e \in E(T)$ of v the child of v incident with e by v_e.

Lemma 8. *Suppose T is an X-tree and $\mathcal{L}$ is a minimal topological lasso for T. Then, for all $v \in \mathring{V}(T)$, we have $|\mathcal{A}(v)| = \binom{|ch(v)|}{2}$. In particular, for any two distinct child edges e_1 and e_2 of v there exists precisely one pair $(a_1, a_2) \in L(v_{e_1}) \times L(v_{e_2})$ such that $a_1a_2 \in \mathcal{L}$.*

Note that Lemma 8 immediately implies that any two minimal topological lassos for the same X-tree must be of equal size.

To be able to establish Proposition 1, we require a further definition. Suppose T is an X-tree and $\mathcal{L}$ is a topological lasso for T. Then for all $v \in V(T)$, we denote by $\Gamma_v(\mathcal{L})$ the subgraph of $\Gamma(\mathcal{L})$ induced by $L(v)$. Note that in case v is a leaf of T and thus an element in X the only vertex in $\Gamma_v(\mathcal{L})$ is v (and $E(\Gamma_v(\mathcal{L})) = \emptyset$).

Proposition 1. *Suppose T is an X-tree and $\mathcal{L}$ is a topological lasso for T. Then, for all $v \in V(T)$, the graph $\Gamma_v(\mathcal{L})$ is connected. In particular, $\Gamma(\mathcal{L})$ is connected.*

Proof. Assume for contradiction that there exists some vertex $v \in V(T)$ such that $\Gamma_v(\mathcal{L})$ is not connected. Then v cannot be a leaf of T and so $v \in \mathring{V}(T)$ must hold. Without loss of generality we may assume that v is such that for all descendants $w \in V(T)$ of v the induced graph $\Gamma_w(\mathcal{L})$ is connected. Since $\mathcal{L}$ is a topological lasso for T and so $G(\mathcal{L}, v)$ is a clique, it follows for any two distinct children $v_1, v_2 \in ch(v)$ that there exists a pair $(x_1, x_2) \in L(v_1) \times L(v_2)$ such that $x_1x_2 \in \mathcal{L}$. Since the assumption on v implies that the graphs $\Gamma_w(\mathcal{L})$ are connected for all children $w \in ch(v)$, it follows that $\Gamma_v(\mathcal{L})$ is connected which is impossible. Thus, $\Gamma_v(\mathcal{L})$ is connected, for all $v \in V(T)$. That $\Gamma(\mathcal{L})$ is connected is a trivial consequence. □

6.2 The case that $\Gamma(\mathcal{L})$ is a block graph

To establish a further property of $\Gamma(\mathcal{L})$ which we will do in Proposition 2, we require some terminology related to block graphs (see e. g. [42]). Suppose G is a graph. Then a vertex of G is called a *cut vertex* if its deletion (plus its incident edges) disconnects G. A graph is called a *block* if it has at least one vertex, is connected, and does not contain a cut vertex. A *block of a graph G* is a maximal connected subgraph of G that is a block and a graph is called a *block graph* if all of its blocks are cliques. For convenience, we refer to a block graph with vertex set X as a *block graph on X*.

As the example of the two minimal topological lassos $\{ab, cd, ef, ac, ce, ea\}$ and $\{ab, bc, cd, de, ef, fa\}$ for the $\{a, \dots, f\}$-tree depicted in Figure 6.1 (ii) indicates, the graph $\Gamma(\mathcal{L})$ associated to a minimal topological lasso $\mathcal{L}$ may be but need not be a block graph. However if it is then Lemma 8 can be strengthened to the following central result where for all positive integers n we put $\langle n \rangle := \{1, \dots, n\}$ and set $\langle 0 \rangle := \emptyset$.

Proposition 2. *Suppose T is an X-tree and $\mathcal{L}$ is a minimal topological lasso for T such that $\Gamma(\mathcal{L})$ is a block graph. Let $v \in \mathring{V}(T)$ and let $v_1, \dots, v_l \in V(T)$ denote the children of v where $l = |ch(v)|$. Then, for all $i \in \langle l \rangle$, there exists a unique leaf $x_i \in L(v_i)$ such that $x_sx_t \in \mathcal{L}$ holds for all $s, t \in \langle l \rangle$ distinct.*

Proof. For all $v \in \mathring{V}(T)$ and all $w \in ch(v)$, put

$$L_w^v := \{x \in L(w) : \text{ there exist } w' \in ch(v) - \{w\} \text{ and } y \in L(w') \text{ such that } xy \in \mathcal{L}\}.$$

We need to show that $|L_w^v| = 1$ holds for all $v \in \mathring{V}(T)$ and all $w \in ch(v)$. To see this, note first that since $G(\mathcal{L}, v)$ is a clique for all $v \in \mathring{V}(T)$, we have, for all $w \in ch(v)$ with $v \in \mathring{V}(T)$, that $L_w^v \neq \emptyset$. Thus, $|L_w^v| \geq 1$ holds for all such v and w.

To establish equality, suppose there exists some interior vertex $v \in \mathring{V}(T)$ and some child $v_1 \in ch(v)$ such that $|L_{v_1}^v| \geq 2$. Choose two distinct leaves x_1 and y_1 of T contained in $L_{v_1}^v$ and denote the parent edge of v_1 by e_1. Note that $v_1 = v_{e_1}$. Since $y_1 \in L_{v_1}^v$, there exists a child edge e_2 of v distinct from e_1 and some $x_2 \in L(v_{e_2})$ such that $y_1x_2 \in \mathcal{L}$. In view of $x_1 \in L_{v_1}^v$, we distinguish between the cases that (i) $x_1z \notin \mathcal{L}$ holds for all $z \in L(v_{e_2})$ and (ii) there exists some $z \in L(v_{e_2})$ such that $x_1z \in \mathcal{L}$.

Assume first that Case (i) holds. Then since $x_1 \in L_{v_1}^v$ there exists a further child edge e_3 of v and some $y_3 \in L(v_{e_3})$ such that $x_1y_3 \in \mathcal{L}$. Since, by Theorem 16, $G(\mathcal{L}, v)$ is a clique and so $\{e_2, e_3\}$ is an edge in $G(\mathcal{L}, v)$, there must exist leaves $y_2 \in L(v_{e_2})$ and $x_3 \in L(v_{e_3})$ such that $y_2x_3 \in \mathcal{L}$. By Proposition 1, the graphs $\Gamma_{v_{e_i}}(\mathcal{L})$, $i = 2, 3$, are connected and, by definition, clearly do not share a vertex. Hence, there must exist a cycle in $\Gamma(\mathcal{L})$ whose vertex set contains $\bigcup_{j \in \langle 3 \rangle} \{x_j, y_j\}$. But then $x_1x_2 \in \mathcal{L}$ must hold since $\Gamma(\mathcal{L})$ is a block graph and so every block in $\Gamma(\mathcal{L})$ is a clique. By Lemma 8 applied to e_1 and e_2, it follows that $x_1 = y_1$ as $x_1, y_1 \in L(v_1)$ and $y_1x_2 \in \mathcal{L}$ which is impossible.

Now assume that Case (ii) holds, that is, there exists some $z \in L(v_{e_2})$ such that $x_1z \in \mathcal{L}$. Then Lemma 8 applied to e_1 and e_2 implies $x_1 = y_1$ as $y_1x_2 \in \mathcal{L}$ also holds which is impossible. □

To illustrate Proposition 2, let T be the X-tree depicted in Figure 6.1 (ii) and let $\mathcal{L}$ be the set of cords of X whose $\Gamma(\mathcal{L})$ graph is pictured in Figure 6.1 (i). Using the notation from Proposition 2 and labelling the children of the root of T from left to right by v_1, v_2 and v_3 it is easy to see that Proposition 2 holds for $x_1 = a$, $x_2 = c$ and $x_3 = e$.

The next result is the main result of this section and lies at the heart of Corollary 3 which provides for an X-tree T and a minimal topological lasso $\mathcal{L}$ for T such that $\Gamma(\mathcal{L})$ is a block graph a close link between the blocks of $\Gamma(\mathcal{L})$, the interior vertices of T and, for all $v \in \mathring{V}(T)$, the child-edge graphs $G(\mathcal{L}, v)$. To establish it, we denote for all $v \in V(T) - \{\rho_T\}$ the parent edge of v by e_v and the set of blocks of a graph G by $Block(G)$.

Theorem 17. *Suppose T is an X-tree and $\mathcal{L}$ is a minimal topological lasso for T such that $\Gamma(\mathcal{L})$ is a block graph. Then, for all $v \in \mathring{V}(T)$, there exists a unique block $B \in Block(\Gamma(\mathcal{L}))$ such that $v = lca_T(V(B))$.*

Proof. We first show existence. Suppose $v \in \mathring{V}(T)$. Let $v_1, \ldots, v_l \in V(T)$ denote the children of v where $l = |ch(v)|$. By Proposition 2, there exists, for all $i \in \langle l \rangle$, a unique leaf $x_i \in L(v_i)$ such that, for all $s, t \in \langle l \rangle$ distinct, we have $x_s x_t \in \mathcal{L}$. Put $A = \{x_1, \ldots, x_l\}$. Clearly, $v = lca_T(A)$ and the graph $G(v)$ with vertex set A and edge set $E = \{\{x, y\} \in \binom{A}{2} : xy \in \mathcal{L}\}$ is a clique. Then since $\Gamma(\mathcal{L})$ is a block graph there must exist a block $B \in Block(\Gamma(\mathcal{L}))$ that contains $G(v)$ as an induced subgraph.

We claim that the graphs $G(v)$ and B are equal. In view of the facts that $A \subseteq V(B)$, the blocks in a block graph are cliques, and $G(v)$ is a clique it suffices to show that $V(B) \subseteq A$. Suppose for contradiction that there exists some $y \in V(B) - A$. Note first that $yx \in \mathcal{L}$ must hold for all $x \in A$. Next note that y cannot be a descendant of v since otherwise there would exist some $i \in \langle l \rangle$ such that $y \in L(v_i)$. Choose some $j \in \langle l \rangle - \{i\}$. Then Lemma 8 applied to e_{v_i} and e_{v_j} implies $x_i = y$ as $yx_j, x_i x_j \in \mathcal{L}$ which is impossible.

Choose some $x \in A$ and put $w = lca_T(x, y)$. Then v is a descendant of w and $w = lca_T(x, y)$ holds for all $x \in A$. Let $w_1 \in V(T)$ and $w_2 \in \mathring{V}(T)$ denote two distinct children of w such that $y \in L(w_1)$ and $x \in L(w_2)$. Then Lemma 8 applied to e_{w_1} and e_{w_2} implies $x_i = x_j$ for all $i, j \in \langle l \rangle$ distinct since $yx \in \mathcal{L}$ holds for all $x \in A$ which is impossible. Thus, $V(B) \subseteq A$, as required. This concludes the proof of the existence part of the theorem.

We next show uniqueness. Suppose for contradiction that there exists some $v \in \mathring{V}(T)$ and distinct blocks $B, B' \in Block(\Gamma(\mathcal{L}))$ such that $lca_T(B) = v = lca_T(B')$. Since every block of $\Gamma(\mathcal{L})$ contains at least two vertices as $\Gamma(\mathcal{L})$ is connected and $|X| \geq 3$, we may choose distinct vertices $b_1, b_2 \in V(B)$ and $b'_1, b'_2 \in V(B')$ such that $lca_T(b_1, b_2) = lca_T(B) = v = lca_T(B') = lca_T(b'_1, b'_2)$. Note that b_1b_2 and $b'_1b'_2$ must be cords in $\mathcal{L}$ as B and B' are cliques of $\Gamma(\mathcal{L})$. We distinguish between the cases that (i) $\{b_1, b_2\} \cap \{b'_1, b'_2\} = \emptyset$ and (ii) $\{b_1, b_2\} \cap \{b'_1, b'_2\} \neq \emptyset$.

We first show that Case (i) cannot hold. Assume for contradiction that Case (i) holds, that is, $\{b_1, b_2\} \cap \{b'_1, b'_2\} = \emptyset$. We claim that $lca_T(b_1, b'_1) = v$. Assume for contradiction that $w := lca_T(b_1, b'_1) \neq v$. Let $v_1 \in ch(v)$ such that v_1 lies on the path from v to w. If $v \neq lca_T(b_2, b'_2)$ then there exists a descendant $w' \in V(T)$ of v such that $lca_T(b_2, b'_2) = w'$. Let $v_2 \in ch(v)$ such that v_2 that lies on the path from v to w'. Then Lemma 8 applied to e_{v_1} and e_{v_2} implies $b_1 = b'_1$ and $b_2 = b'_2$ as $b_1b_2, b'_1b'_2 \in \mathcal{L}$ which is impossible. Thus, $lca_T(b_2, b'_2) = v$ must hold. Let $v_2, v'_2 \in ch(v)$ such that $b_2 \in L(v_2)$ and $b'_2 \in L(v'_2)$. Then since $b_1, b'_1 \in L(v_1)$ and $b_1b_2, b'_1b'_2 \in \mathcal{L}$, Proposition 2 implies $b'_1 = b_1$. Consequently, $\{b_1, b_2\} \cap \{b'_1, b'_2\} \neq \emptyset$ which is impossible.

Thus, $lca_T(b_2, b'_2) = v$ cannot hold and so

$$lca_T(b_1, b'_1) = v,$$

as claimed. Swapping the roles of b_1, b'_1 and b_2, b'_2 in the previous claim implies that $v = lca_T(b_2, b'_2)$ must hold, too. For $i = 1, 2$ let $v_i, v'_i \in ch(v)$ such that $b_i \in L(v_i)$ and $b'_i \in L(v'_i)$. Then, by Lemma 8, there exist pairs $(c, c') \in L(v_1) \times L(v'_1)$ and $(d, d') \in L(v_2) \times L(v'_2)$ such that $cc', dd' \in \mathcal{L}$. Since $(b_1, b_2) \in L(v_1) \times L(v_2)$ and $(b'_1, b'_2) \in L(v'_1) \times L(v'_2)$ and $b_1b_2, b'_1b'_2 \in \mathcal{L}$, Proposition 2 implies that $c = b_1$, $b_2 = d$, $d' = b'_2$ and $c' = b'_1$. But then C: $c' = b'_1, b'_2 = d', d = b_2, b_1 = c, c'$ is a cycle in $\Gamma(\mathcal{L})$. Since $\Gamma(\mathcal{L})$ is a block graph it follows that there must exist a block B^C in $\Gamma(\mathcal{L})$ that contains C. Since $\{b_1, b_2\} \subseteq V(B^C) \cap V(B)$ and two distinct blocks of a block graph can share at most one vertex it follows that B^C and B must coincide. Since $\{b'_1, b'_2\} \subseteq V(B^C) \cap V(B')$ holds too, similar arguments imply that B^C must also coincide with B'. Thus, B and B' must be equal which is impossible.

Hence Case (i) cannot hold, as required.

Thus, Case (ii) must hold, that is, $\{b_1, b_2\} \cap \{b'_1, b'_2\} \neq \emptyset$. Since any two distinct blocks in a block graph can share at most one vertex it follows that $|\{b_1, b_2\} \cap \{b'_1, b'_2\}| = 1$. Without loss of generality we may assume that $b_1 = b'_1$. We first claim that

$$lca_T(b_2, b'_2) = v.$$

Assume to the contrary that $lca_T(b_2, b'_2) \neq v$. Then there exist distinct children $v_1, v_2 \in ch(v)$ such that $b_1 \in L(v_1)$ and $b_2, b'_2 \in L(v_2)$ hold. Since both b_1b_2 and $b'_1b'_2 = b_1b'_2$ are cords in $\mathcal{L}$, Lemma 8 applied to e_{v_1} and e_{v_2} implies $b'_2 = b_2$. Hence, $|\{b_1, b_2\} \cap \{b'_1, b'_2\}| = 2$ which is impossible. Thus, $lca_T(b_2, b'_2) = v$, as claimed.

Let $v_1, v_2, v'_2 \in ch(v)$ such that $b_1 \in L(v_1)$, $b_2 \in L(v_2)$, and $b'_2 \in L(v'_2)$. By Lemma 8, there exist some $(c, c') \in L(v_2) \times L(v'_2)$ such that $cc' \in \mathcal{L}$. Since we also have $(b_1, b_2) \in L(v_1) \times L(v_2)$ with $b_1b_2 \in \mathcal{L}$ holding and $(b_1, b'_2) \in L(v_1) \times L(v'_2)$ with $b'_2b_1 = b'_2b'_1 \in \mathcal{L}$ holding, Proposition 2 implies that $b_2 = c$ and $b'_2 = c'$. Hence, C: $b_1 = b'_1, b'_2 = c', c = b_2, b_1$ is a cycle in $\Gamma(\mathcal{L})$ and so similar arguments as in the corresponding subcase for Case (i) imply that

B and B' must coincide which is impossible. Thus, $lca_T(b_2, b'_2) = v$ cannot hold which concludes the discussion of Case (ii) and thus the proof of the uniqueness part of the theorem. □

In view of Theorem 17, we denote for T an X-tree, a minimal topological lasso $\mathcal{L}$ for T such that $\Gamma(\mathcal{L})$ is a block graph, and a vertex $v \in \mathring{V}(T)$ the unique block B in $\Gamma(\mathcal{L})$ for which $v = lca_T(V(B))$ holds by $B_v^{\mathcal{L}}$, or simply by B_v if the set $\mathcal{L}$ of cords is clear from the context. Moreover, we denote for all $x \in L(v)$ the child of v on the path from v to x by v_x.

Corollary 3. *Suppose T is an X-tree and $\mathcal{L}$ is a minimal topological lasso for T such that $\Gamma(\mathcal{L})$ is a block graph. Then the map*

$$\psi : \mathring{V}(T) \to Block(\Gamma(\mathcal{L})) : v \mapsto B_v$$

is a bijection with inverse map $\psi^{-1} : Block(\Gamma(\mathcal{L})) \to \mathring{V}(T)$*:* $B \mapsto lca_T(V(B))$*. Moreover, the map*

$$\chi : Block(\Gamma(\mathcal{L})) \to \{G(\mathcal{L}, v) \,:\, v \in \mathring{V}(T)\} : B \mapsto G(\mathcal{L}, \psi^{-1}(B))$$

is bijective and, for all $B \in Block(\Gamma(\mathcal{L}))$*, the map*

$$\xi_B : V(B) \to V_{\psi^{-1}(B)} : x \mapsto e_{\psi^{-1}(B)_x}$$

induces a graph isomorphism between B *and the child-edge graph* $G(\mathcal{L}, \psi^{-1}(B))$*.*

Proof. In view of Theorem 17, the map ψ is clearly well-defined and injective. To see that ψ is surjective let $B \in Block(\Gamma(\mathcal{L}))$ and put $v_B = lca_T(V(B))$. Clearly, $v_B \in \mathring{V}(T)$. Since $B_{v_B} = \psi(v_B)$ is a block in $\Gamma(\mathcal{L})$ for which also $v_B = lca_T(V(B_{v_B}))$ holds, Theorem 17 implies that $\psi(v_B)$ and B must coincide. Consequently, ψ must also be surjective and thus bijective. That the map ψ^{-1} is as stated is trivial. Combined with Theorem 16, the bijectivity of the map ψ implies in particular that, for all $B \in Block(\Gamma(\mathcal{L}))$, the map $\xi_B : V(B) \to V_{\psi^{-1}(B)}$ from $V(B)$ to the vertex set $V_{\psi^{-1}(B)}$ of the child-edge graph $G(\mathcal{L}, \psi^{-1}(B))$ induces a graph isomorphism between B and $G(\mathcal{L}, \psi^{-1}(B))$.

To see that the map χ is bijective note first that χ is well-defined since $\psi^{-1}(B) \in \mathring{V}(T)$ holds for all blocks $B \in Block(\Gamma(\mathcal{L}))$. To see that χ is injective assume that there exist blocks $B_1, B_2 \in Block(\Gamma(\mathcal{L}))$ such that $\chi(B_1) = \chi(B_2)$ but B_1 and B_2 are distinct. Then $\psi^{-1}(B_1) \neq \psi^{-1}(B_2)$ as ψ is a bijection from $\mathring{V}(T)$ to $Block(\Gamma(\mathcal{L}))$. Combined with the fact that, for all $B \in Block(\Gamma(\mathcal{L}))$, the map ξ_B induces a graph isomorphism between B and $G(\mathcal{L}, \psi^{-1}(B))$ it follows that $\chi(B_1) = G(\mathcal{L}, \psi^{-1}(B_1)) \neq G(\mathcal{L}, \psi^{-1}(B_2)) = \chi(B_2)$ which is impossible. Thus, χ must be injective. Combined with the fact that $|Blocks(\Gamma(\mathcal{L}))| = |\mathring{V}(T)| = |\{G(\mathcal{L}, v) \,:\, v \in \mathring{V}(T)\}|$ it follows that χ must also be surjective and thus bijective. □

6.3 A special type of minimal topological lasso

Returning to the example depicted in Figure 6.1, it should be noted that, in addition to being a block graph, $\Gamma(\mathcal{L})$ enjoys a very special property where $\mathcal{L}$ is the minimal topological lasso considered in that example. More precisely, every vertex of $\Gamma(\mathcal{L})$ is contained in at most two blocks. Put differently, $\Gamma(\mathcal{L})$ is a claw-free graph. Motivated by this, we call a minimal topological lasso $\mathcal{L}$ *distinguished* if $\Gamma(\mathcal{L})$ is a claw-free block graph. Note that such block graphs are precisely the *line graphs of (unrooted) trees* where for any graph G the associated line graph has vertex set $E(G)$ and two vertices $a, b \in E(G)$ are joined by an edge if $a \cap b \neq \emptyset$ [76].

In this section, we show in Theorem 18 that distinguished minimal topological lassos are a very special type of lasso in that for every X-tree T any minimal topological lasso $\mathcal{L}$ for T can be transformed into a distinguished minimal topological lasso $\mathcal{L}^*$ for T via a *repeated application* (that is, $l \geq 0$ applications) of the rule:

(R) If $xy, yz \in \mathcal{L}$ and $lca_T(y,z)$ is a descendant of $lca_T(x,y)$ in T then delete xy from the edge set of $\Gamma(\mathcal{L})$ and add the edge xz to it.

Before we make this more precise which we will do next, we remark that since a topological lasso for a star tree is in particular a distinguished minimal topological lasso for it, we will for this and the next two sections restrict our attention to *non-degenerate* X-trees, that is, X-trees that are not star trees on X.

Suppose T is a non-degenerate X-tree and $\mathcal{L}$ is a set of cords of X. Let $\mathring{V}(T)$ denote a set of colours and let

$$\gamma_{(\mathcal{L},T)} : \mathcal{L} \to \mathring{V}(T) : ab \mapsto lca_T(a,b)$$

denote an edge colouring of $\Gamma(\mathcal{L})$ in terms of the interior vertices of T. Note that if $\mathcal{L}$ is a topological lasso for T then Theorem 16 implies that $\gamma_{(\mathcal{L},T)}$ is surjective. Returning to Rule (R), note that a repeated application of that rule to such a set $\mathcal{L}$ of cords results in a set $\mathcal{L}'$ of cords that is also a topological lasso for T. Furthermore, note that if $\mathcal{L}$ is a minimal topological lasso for T then $\mathcal{L}'$ is necessarily also a minimal topological lasso for T. Finally note for all $v \in \mathring{V}(T)$ that $|\gamma^{-1}_{(\mathcal{L},T)}(v)| = 1$ or $|\gamma^{-1}_{(\mathcal{L},T)}(v)| \geq 3$ must hold in this case.

Lemma 9. *Suppose T is a non-degenerate X-tree and $\mathcal{L}$ is a minimal topological lasso for T. Put $\gamma = \gamma_{(\mathcal{L},T)}$ and assume that $v \in \mathring{V}(T)$ such that $|\gamma^{-1}(v)| \geq 3$. Then for any three pairwise distinct cords $c_1, c_2, c_3 \in \gamma^{-1}(v)$, there exists a cycle C_v in $\Gamma(\mathcal{L})$ such that $c_1, c_2, c_3 \in E(C_v)$ and, for all $c \in E(C_v)$, $\gamma(c)$ either equals v or is a descendant of v.*

Proof. Let $v \in \mathring{V}(T)$ and let $c_1 = x_1y_1$, $c_2 = x_2y_2$ and $c_3 = x_3y_3$ denote three pairwise distinct cords in $\gamma^{-1}(v)$. For all $i \in \langle 3 \rangle$, let $v_i \in ch(v)$ such that v_i lies on the path from v to x_i in T and let $w_i \in ch(v)$ such that w_i lies on the path from v to y_i in T. Then, by Lemma 8, there exists unique pairs $(s_1, t_1) \in L(v_1) \times L(v_2)$, $(s_2, t_2) \in L(w_2) \times L(w_3)$, and $(s_3, t_3) \in L(w_1) \times L(v_3)$ such that, for all $i \in \langle 3 \rangle$, we have $s_it_i \in \mathcal{L}$. Since for all such i, we also have that $x_i \in L(v_i)$ and $y_i \in L(w_i)$ and, by Proposition 1, the graphs $\Gamma_{v_i}(\mathcal{L})$ and $\Gamma_{w_i}(\mathcal{L})$ are connected, it follows that there exists a cycle C_v in $\Gamma(\mathcal{L})$ that contains, for all $i \in \langle 3 \rangle$, the cords c_i and s_it_i in its edge set.

It remains to show that for every edge $c \in E(C_v)$, we have that $\gamma(c)$ either equals v or is a descendant of v. Suppose $c \in E(C_v)$. If there exists some $i \in \langle 3 \rangle$ such that $c \in \{c_i, s_it_i\}$ then $\gamma(c) = v$ clearly holds. So assume that this is not the case. Without loss of generality, we may assume that c lies on the path P from x_1 to s_1 in C_v that does not cross y_1. Since P is a subgraph of $\Gamma_{v_1}(\mathcal{L})$ and, implied by Proposition 1, every edge in $\Gamma_{v_1}(\mathcal{L})$ is coloured via γ with a descendant of v_1, it follows that $\gamma(c)$ is a descendant of v. □

To establish Theorem 18, we require further terminology. Let T be a non-degenerate X-tree, $\mathcal{L}$ a minimal topological lasso for T, and $v \in \mathring{V}(T)$. Then we denote by $H_{\mathcal{L}}(v)$ the induced subgraph of $\Gamma(\mathcal{L})$ whose vertex set is the set of all $x \in X$ that are incident with some cord $c \in \mathcal{L}$ for which $\gamma_{(\mathcal{L},T)}(c) = v$ holds. Moreover, we denote the set of cut vertices of a connected block graph G by $Cut(G)$ and note that in every connected block graph G

there must exist a vertex that is contained in at most one block of G. This last observation is central to the proof of Theorem 18 (ii).

Theorem 18. *Suppose T is a non-degenerate X-tree and $\mathcal{L}$ is a minimal topological lasso for T. Then there exists an ordering $\sigma : v_0, v_1, \ldots, v_k = \rho_T$, $k = |\mathring{V}(T)|$, of $\mathring{V}(T)$ such that the following holds:*

(i) There exists a sequence $\mathcal{L}_{v_0} = \mathcal{L}, \mathcal{L}_{v_1}, \ldots, \mathcal{L}^\dagger = \mathcal{L}_{v_k}$ of minimal topological lassos $\mathcal{L}_{v_i}$ for T, $i \in \langle k \rangle$, such that for all such i, we have:

(L1) $\mathcal{L}_{v_i}$ is obtained from $\mathcal{L}_{v_{i-1}}$ via a repeated application of Rule (R) and $H_{\mathcal{L}_{v_i}}(v_i)$ is a maximal clique in $\Gamma(\mathcal{L}_{v_i})$.

(L2) For all $j \in \langle i-1 \rangle$, $H_{\mathcal{L}_{v_i}}(v_j)$ is a maximal clique in $\Gamma(\mathcal{L}_{v_i})$.

In particular, $\Gamma(\mathcal{L}^\dagger)$ is a block graph.

(ii) If $\Gamma(\mathcal{L})$ is a block graph then there exists a sequence $\mathcal{L}_{v_0} = \mathcal{L}, \mathcal{L}_{v_1}, \ldots, \mathcal{L}^* = \mathcal{L}_{v_k}$ of minimal topological lassos $\mathcal{L}_{v_i}$ for T, $i \in \langle k \rangle$, such that for all such i, we have:

(L1') $\mathcal{L}_{v_i}$ is obtained from $\mathcal{L}_{v_{i-1}}$ via a repeated application of Rule (R) and $\Gamma(\mathcal{L}_{v_i})$ is a block graph.

(L2') $\Gamma_{v_i}(\mathcal{L}_{v_i})$ is a claw-free block graph.

In particular, $\mathcal{L}^*$ is a distinguished minimal topological lasso for T.

Proof. For all $i \in \langle k \rangle$, put $\mathcal{L}_i = \mathcal{L}_{v_i}$ and $\gamma_i = \gamma_{(\mathcal{L}_i,T)}$. Clearly, if $\mathcal{L}$ is distinguished then the sequences as described in (i) and (ii) exist. So assume that $\mathcal{L}$ is not distinguished. For all $v \in \mathring{V}(T)$, let $l(v)$ denote the length of the path from the root ρ_T of T to v and put $h = \max_{v \in \mathring{V}(T)}\{l(v)\}$. Note that $h \geq 1$ as T is non-degenerate. For all $i \in \langle h \rangle$, let $V(i) \subseteq \mathring{V}(T)$ denote the set of all interior vertices v of T such that $l(v) = i$. Let σ denote an ordering of the vertices in $\mathring{V}(T)$ such that the vertices in $V(h)$ come first (in any order), then (again in any order) the vertices in $V(h-1)$ and so on with the last vertex in that ordering being ρ_T.

(i) Suppose $v \in \mathring{V}(T)$. If $v \in V(h)$ then we may assume without loss of generality that $v = v_1$. Then v_1 is the parent of a pseudo-cherry of T and so Theorem 16 implies that $H_{\mathcal{L}}(v_1)$ is a maximal clique in $\Gamma(\mathcal{L})$. Thus, $\mathcal{L}_1 := \mathcal{L}$ is a minimal topological lasso for T that satisfies Properties (L1) and (L2).

So assume that $v \notin V(h)$. Then there exists some $|V(h)| < i \leq k$ such that $v = v_i$. Without loss of generality, we may assume that v_i is such that, for all $j \in \langle i-1 \rangle$, $\mathcal{L}_j$ is a minimal topological lasso for T that satisfies Properties (L1) and (L2). If v_i is the parent of a pseudo-cherry of T then similar arguments as before imply that $\mathcal{L}_i := \mathcal{L}_{i-1}$ is a minimal topological lasso for T that satisfies Properties (L1) and (L2). So assume that v_i is not the parent of a pseudo-cherry of T. We distinguish between the cases that $H_{\mathcal{L}_{i-1}}(v)$ is a maximal clique in $\mathcal{L}_{i-1}$ and that it is not.

Assume first that $H_{\mathcal{L}_{i-1}}(v)$ is a maximal clique in $\mathcal{L}_{i-1}$. Then since $\mathcal{L}_{i-1}$ is a minimal topological lasso for T that satisfies Properties (L1) and (L2), it is easy to see that $\mathcal{L}_i := \mathcal{L}_{i-1}$ is also a minimal topological lasso for T that satisfies Properties (L1) and (L2). So assume that $H_{\mathcal{L}_{i-1}}(v)$ is not a maximal clique in $\mathcal{L}_{i-1}$. Then $H_{\mathcal{L}_{i-1}}(v)$ must contain three pairwise distinct edges, $e_1 = x_1y_1$, $e_2 = x_2y_2$, and $e_3 = x_3y_3$ say, such that $\{e_1, e_2, e_3\}$ is not the edge set of a 3-clique in $H_{\mathcal{L}_{i-1}}(v)$. For all $i \in \langle 3 \rangle$, put $z_i = lca_T(x_i, y_i)$. Then Lemma 9 combined with a repeated application of Rule (R) to $\mathcal{L}_{i-1}$ implies that, for all $i \in \langle 3 \rangle$, we can find elements $x_i' \in L(z_i)$ such that

$$\mathcal{L}_{i-1}' = \mathcal{L}_{i-1} - \{x_1y_1, x_2y_2, x_3y_3\} \cup \{x_1'x_2', x_2'x_3', x_3'x_1'\}$$

is a minimal topological lasso for T and the cords $x_1'x_2'$, $x_2'x_3'$, and $x_3'x_1'$ form a 3-clique in $H_{\mathcal{L}_{i-1}'}(v)$. Transforming $\mathcal{L}_{i-1}'$ further by processing any three pairwise distinct edges in $H_{\mathcal{L}_{i-1}'}(v)$ that do not already form a 3-clique in the same way and so on eventually yields a minimal topological lasso $\mathcal{L}_i$ for T such that any three pairwise distinct edges in $H_{\mathcal{L}_i}(v)$ form a 3-clique. But this implies that $H_{\mathcal{L}_i}(v)$ is a maximal clique in $\Gamma(\mathcal{L}_i)$ and so Property (L1) is satisfied by $\mathcal{L}_i$. Since only edges e of $\Gamma(\mathcal{L}_{i-1})$ have been modified by the above transformation for which $\gamma_{i-1}(e) = v$ holds and, by assumption, $\mathcal{L}_{i-1}$ satisfies Property (L2) it follows that $\mathcal{L}_i$ also satisfies that property.

Processing the successor of v_i in σ in the same way and so on yields a minimal topological lasso $\mathcal{L}^\dagger$ for T for which $\Gamma(\mathcal{L}^\dagger)$ is a block graph. This completes the proof of (i).

(ii) For all $i \in \langle k \rangle$ and all vertices $w \in \mathring{V}(T)$ put $B_w^i = B_w^{\mathcal{L}_i}$. Suppose that $v \in \mathring{V}(T)$. If $v \in V(h)$ then we may assume without loss of generality that $v = v_1$. Then v is the parent of a pseudo-cherry of T and so $\mathcal{L}_1 := \mathcal{L}$ clearly satisfies Properties (L1') and (L2').

So assume that $v \notin V(h)$. Then there exists some $|V(h)| < i \leq k$ such that $v = v_i$. Without loss of generality, we may assume that v_i is minimal, that is, for all $j \in \langle i-1 \rangle$, we have that $\mathcal{L}_j$ is a minimal topological lasso for T that satisfies Properties (L1') and (L2'). If v is the parent of a pseudo-cherry of T then similar arguments as before imply that $\mathcal{L}_i := \mathcal{L}_{i-1}$ satisfies Properties (L1') and (L2'). So assume that v is not the parent of a pseudo-cherry of T. If $\Gamma_v(\mathcal{L}_{i-1})$ is a claw-free block graph then setting $\mathcal{L}_i := \mathcal{L}_{i-1}$ implies that $\mathcal{L}_i$ satisfies Properties (L1') and (L2').

So assume that this is not the case, that is, there exists a vertex $x \in L(v)$ that, in addition to being a vertex in the block B_v^{i-1} of $\Gamma(\mathcal{L}_{i-1})$ and thus of $\Gamma_v(\mathcal{L}_{i-1})$, is also a vertex in $l \geq 2$ further blocks $B_1, \ldots, B_l$ of $\Gamma_v(\mathcal{L}_{i-1})$ which are also blocks in $\Gamma(\mathcal{L})$. Then there exists a path P from v to x in T that contains, for all $l \geq 2$, the vertices $\psi^{-1}(B_1), \ldots, \psi^{-1}(B_l)$ in its vertex set where $\psi : \mathring{V}(T) \to \mathrm{Block}(\Gamma(\mathcal{L}))$ is the map from Corollary 3. Let $w \in ch(v)$ denote the child of v that lies on P. Note that since $l \geq 2$, we have $w \in \mathring{V}(T)$. Without loss of generality, we may assume that $w = v_{i-1}$. The fact that $\Gamma(\mathcal{L}_{i-1})$ is a block graph and so $\Gamma_{v_{i-1}}(\mathcal{L}_{i-1})$ is a block graph combined with the fact that $\Gamma_{v_{i-1}}(\mathcal{L}_{i-1})$ is connected implies, in view of the observation preceding Theorem 18, that we may choose some $y \in L(v_{i-1}) - Cut(\Gamma_{v_{i-1}}(\mathcal{L}_{i-1}))$. Then y is a vertex in precisely one block of $\Gamma_{v_{i-1}}(\mathcal{L}_{i-1})$ and thus can be a vertex in at most two blocks of $\Gamma_v(\mathcal{L}_{i-1})$. Consequently, $y \neq x$. Applying Rule (R) repeatedly to $\mathcal{L}_{i-1}$, let $\mathcal{L}_i$ denote the set of cords obtained from $\mathcal{L}_{i-1}$ by replacing, for all $i \leq l \leq k$, every cord of $\mathcal{L}_{i-1}$ of the form xa with $a \in V(B_{v_l}^{i-1})$ by the cord ya. Then,

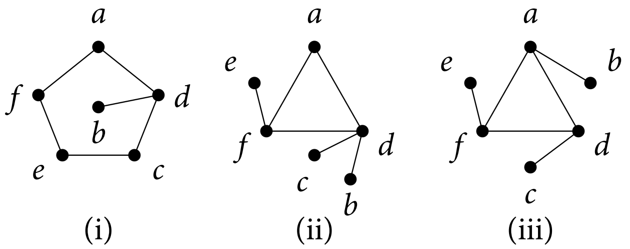


Figure 6.2: For $X = \{a, \dots, f\}$ and the X-tree T' pictured in Figure 6.1 (iii), we depict in (i) the minimal topological lasso $\mathcal{L} = \{ad, ec, fa, fe, cd, bd\}$ for T' in the form of $\Gamma(\mathcal{L})$. In the same way as in (i), we depict in (ii) the transformed minimal topological lasso $\mathcal{L}^\dagger$ for T' such that $\Gamma(\mathcal{L}^\dagger)$ is a block graph and in (iii) the distinguished minimal topological lasso $\mathcal{L}^*$ for T' obtained from $\mathcal{L}^\dagger$ – see text for details.

by construction, $\mathcal{L}_i$ is a minimal topological lasso for T and $\Gamma(\mathcal{L}_i)$ is a block graph. Hence, $\mathcal{L}_i$ satisfies Property (L1'). Moreover, since $\Gamma_{v_{i-1}}(\mathcal{L}_{i-1})$ is claw-free it follows that $\Gamma_{v_i}(\mathcal{L}_i)$ is claw-free and so $\mathcal{L}_i$ satisfies Property (L2'), too.

Applying the above arguments to the successor of v_i in σ and so on eventually yields a minimal topological lasso $\mathcal{L}_k$ for T that satisfies Properties (L1') and (L2'). Thus, $\Gamma_{v_k}(\mathcal{L}_k)$ is a claw-free block graph and, so, $\mathcal{L}^*$ is a distinguished minimal topological lasso for T. □

To illustrate Theorem 18, let $X = \{a, \dots, f\}$ and consider the X-tree T' depicted in Figure 6.1 (iii) along with the set $\mathcal{L} = \{ad, ec, fa, ef, cd, bd\}$ of cords of X which we depict in Figure 6.2 (i) in the form of $\Gamma(\mathcal{L})$.

Using for example Theorem 16, it is straight-forward to check that $\mathcal{L}$ is a minimal topological lasso for T' but $\Gamma(\mathcal{L})$ is clearly not a block graph and so $\mathcal{L}$ is also not distinguished. To transform $\mathcal{L}$ into a distinguished minimal topological lasso $\mathcal{L}^*$ for T' as described in Theorem 18, consider the ordering $v_1 = lca_{T'}(e, f)$, $v_2 = lca_{T'}(c, d)$, $v_3 = lca_{T'}(a, d)$, $v_4 = \rho_{T'}$ of the interior vertices of T'. For all $i \in \langle 4 \rangle$, put $\mathcal{L}_i = \mathcal{L}_{v_i}$. Then we first transform $\mathcal{L}$ into a minimal topological lasso $\mathcal{L}^\dagger$ for T' as described in Theorem 18 (i). For this we have $\mathcal{L} = \mathcal{L}_0 = \mathcal{L}_1 = \mathcal{L}_2$ and $\mathcal{L}_3$ is obtained from $\mathcal{L}_2$ by first applying Rule (R) to the cords $ec, cd \in \mathcal{L}_2$ resulting in the deletion of the cord ce from $\mathcal{L}_2$ and the addition of the cord ed

to $\mathcal{L}_2$ and then to the cords $fe, ed \in \mathcal{L}_2$ resulting in the deletion of the cord ed from $\mathcal{L}_2$ and the addition of the cord fd to it. The graph $\Gamma(\mathcal{L}_3)$ is depicted in Figure 6.2 (ii). Note that $\mathcal{L}_3 = \mathcal{L}^\dagger$ and that although $\Gamma(\mathcal{L}^\dagger)$ is clearly a block graph $\mathcal{L}^\dagger$ is not distinguished.

To transform $\mathcal{L}^\dagger$ into a distinguished minimal topological lasso $\mathcal{L}^*$ for T', we next apply Theorem 18 (ii). For this, we need only consider the vertex d of $\Gamma(\mathcal{L}^\dagger)$ that is, we have $\mathcal{L}^\dagger = \mathcal{L}_0 = \mathcal{L}_1 = \mathcal{L}_2 = \mathcal{L}_3$. Since the child of v_4 on the path from v_4 to d is v_3, we may choose a as the element y in $L(v_3) - Cut(\Gamma_{v_3}(\mathcal{L}_3))$. Then applying Rule (R) to the cords $bd, da \in \mathcal{L}_3$ implies the deletion of bd from $\mathcal{L}_3$ and the addition of the cord ab to it. The resulting minimal topological lasso for T' is $\mathcal{L}^*$ which we depict in Figure 6.2 (iii) in the form of $\Gamma(\mathcal{L}^*)$.

We conclude this section by remarking in passing that combined with Theorem 16 which implies that any minimum sized topological lasso for an X-tree T must have $\sum_{v \in \mathring{V}(T)} \binom{|ch(v)|}{2}$ cords, Theorem 18 and Corollary 3 imply that the minimum sized topological lassos of an X-tree T are precisely the minimal topological lassos of T.

6.4 A sufficient condition for being distinguished

In this section, we turn our attention towards presenting a sufficient condition for a minimal topological lasso for some X-tree T to be a distinguished minimal topological lasso for T. In the next section, we will show that this condition is also necessary.

We start our discussion with introducing some more terminology. Suppose T is a non-degenerate X-tree. Put $cl(T) = \{L(v) : v \in \mathring{V}(T) - \{\rho_T\}\}$ and note that $cl(T) \neq \emptyset$. For all $A \in cl(T)$, put $cl_A(T) := \{B \in cl(T) : B \subsetneq A\}$ and note that a vertex $v \in \mathring{V}(T) - \{\rho_T\}$ is the parent of a pseudo-cherry of T if and only if $cl_{L(v)}(T) = \emptyset$. For σ a total ordering of X and $\min_\sigma(C)$ denoting the minimal element of a non-empty subset C of X, we call a map of the form

$$f : cl(T) \to X : A \mapsto \begin{cases} \min_\sigma(A - \{f(B) : B \in cl_A(T)\}) & \text{if } cl_A(T) \neq \emptyset, \\ \min_\sigma(A) & \text{else.} \end{cases}$$

a *cluster marker map (for T and σ)*. Note that since $|\mathring{V}(T')| \leq |X| - 1$ holds for all X-trees T' and so $A - \{f(B) : B \in cl_A(T)\} \neq \emptyset$ must hold for all $A \in cl(T)$ with $cl_A(T) \neq \emptyset$, it follows that f is well-defined. Also note that if $v \in \mathring{V}(T)$ is the parent of a pseudo-cherry C of T then $f(L(v)) = f(C) = \min_\sigma(C)$ as $cl_C(T) = \emptyset$ in this case. Finally, note that it is easy to see that a cluster marker map must be injective but need not be surjective.

We are now ready to present a construction of a distinguished minimal topological lasso which underpins the aforementioned sufficient condition that a minimal topological lasso must satisfy to be distinguished. Suppose that T is a non-degenerate X-tree, that σ is a total ordering of X, and that $f : cl(T) \to X$ is a cluster marker map for T and σ. We first associate to every interior vertex $v \in \mathring{V}(T)$ a set $\mathcal{L}_{(T,f)}(v)$ defined as follows. Let $l_1, \ldots, l_{k_v}$ denote the children of v that are leaves of T and let $v_1, \ldots v_{p_v}$ denote the children of v that are also interior vertices of T. Note that $k_v = 0$ or $p_v = 0$ might hold but not both. Put $\binom{\emptyset}{2} = \binom{\langle 1 \rangle}{2} = \emptyset$. Then we set

$$\mathcal{L}_{(T,f)}(v) := \bigcup_{\{i,j\} \in \binom{\langle k_v \rangle}{2}} \{l_i l_j\} \cup \bigcup_{\{i,j\} \in \binom{\langle p_v \rangle}{2}} \{f(L(v_i)) f(L(v_j))\} \cup \bigcup_{i \in \langle k_v \rangle,\ j \in \langle p_v \rangle} \{l_i f(L(v_j))\}.$$

Note that $|\mathcal{L}_{(T,f)}(v)| \geq 1$ must hold for all $v \in \mathring{V}(T)$. Finally, we set

$$\mathcal{L}_{(T,f)} := \bigcup_{v \in \mathring{V}(T)} \mathcal{L}_{(T,f)}(v).$$

To illustrate these definitions, consider the $X = \{a, \ldots, f\}$-tree T' depicted in Figure 6.1 (iii). Let σ denote the lexicographic ordering of the elements in X. Then the map

$f : cl(T') \to X$ defined by setting

$$f(\{c,d\}) = c, \;\; f(\{e,f\}) = e, \;\; \text{and}\; f(X - \{b\}) = a$$

is a cluster marker map for T' and σ and $\mathcal{L}_{(T,f)}$ (or more precisely the graph $\Gamma(\mathcal{L}_{(T',f)})$) is depicted in Figure 6.1 (i).

To help establish Theorem 19, we require some intermediate results which are of interest in their own right and which we present next. To this end, we denote for a vertex $v \in \mathring{V}(T) - \{\rho_T\}$ by $T(v)$ the $L(v)$-tree with root v obtained from T by deleting the parent edge of v.

Lemma 10. *Suppose T is a non-degenerate X-tree, σ is a total ordering of X, and $f : cl(T) \to X$ is a cluster marker map for T and σ. Then the following hold*

(i) $\mathcal{L}_{(T,f)}$ is a minimal topological lasso for T.

(ii) $\Gamma(\mathcal{L}_{(T,f)})$ is connected.

(iii) If v and w are distinct interior vertices of T then $|\bigcup \mathcal{L}_{(T,f)}(v) \cap \bigcup \mathcal{L}_{(T,f)}(w)| \leq 1$.

(iv) Suppose $x \in X$. Then there exist distinct vertices $v, w \in \mathring{V}(T)$ such that

$$x \in \bigcup \mathcal{L}_{(T,f)}(v) \cap \bigcup \mathcal{L}_{(T,f)}(w)$$

if and only if there exists some $u \in \mathring{V}(T) - \{\rho_T\}$ such that $x = f(L(u))$.

Proof. For all $v \in \mathring{V}(T)$, set $\mathcal{L}(v) = \mathcal{L}_{(T,f)}(v)$.

(i) This is an immediate consequence of Theorem 16 and the respective definitions of the set $\mathcal{L}(v)$ where $v \in \mathring{V}(T)$ and the graph $G(\mathcal{L}', v)$ where $\mathcal{L}'$ is a set of cords of X and v is again an interior vertex of T.

(ii) This is an immediate consequence of Proposition 1 and Lemma 10 (i).

(iii) This is an immediate consequence of the fact that, for all vertices $u \in \mathring{V}(T)$ and all $x, y \in \bigcup \mathcal{L}(u)$ distinct, we have $u = lca_T(x, y)$.

(iv) Let $x \in X$ and assume first that there exist distinct vertices $v, w \in \mathring{V}(T)$ such that $x \in \bigcup \mathcal{L}(v) \cap \bigcup \mathcal{L}(w)$ but $x \neq f(L_T(u))$, for all $u \in \mathring{V}(T) - \{\rho_T\}$. Then x must be a leaf of T that is simultaneously adjacent with v and w which is impossible. Thus, there must exist some $u \in \mathring{V}(T)$ such that $x = f(L(u))$.

Conversely, assume that $x = f(L(u))$ for some $u \in \mathring{V}(T) - \{\rho_T\}$. Then $x \in L(u)$ and so there must exist an interior vertex w of $T(u)$ that is adjacent with x. Hence, $x \in \bigcup \mathcal{L}(w)$. Let v denote the parent of u in T which exists since $u \neq \rho_T$. Then $x = f(L(u)) \in \bigcup \mathcal{L}(v)$ and so $x \in \bigcup \mathcal{L}(v) \cap \bigcup \mathcal{L}(w)$, as required. □

Note that $u \in \{v, w\}$ need not hold for u, v and w as in the statement of Lemma 10 (iv). Indeed, suppose T is the $X = \{a, b, c, d\}$-tree with unique cherry $\{a, b\}$ and d adjacent with the root ρ_T of T. Let σ denote the lexicographic ordering of X and let $f : cl(T) \to X$ be (the unique) cluster marker map for T and σ. Set $x = b$, $v = lca_T(a, b)$, $w = \rho_T$. Then $x = f(L(u))$ where $u = lca_T(a, c)$ and $x \in \bigcup \mathcal{L}(v) \cap \bigcup \mathcal{L}(w)$ but $u \notin \{v, w\}$.

Proposition 3. *Suppose T is a non-degenerate X-tree, σ is a total ordering of X, and $f : cl(T) \to X$ is a cluster marker map for T and σ. Then $\Gamma(\mathcal{L}_{(T,f)})$ is a connected block graph and every block of $\Gamma(\mathcal{L}_{(T,f)})$ is of the form $\Gamma(\mathcal{L}_{(T,f)}(v))$, for some $v \in \mathring{V}(T)$.*

Proof. For all $v \in \mathring{V}(T)$, set $\mathcal{L}(v) = \mathcal{L}_{(T,f)}(v)$ and put $\mathcal{L} = \mathcal{L}_{(T,f)}$. We claim that if C is a cycle in $\Gamma(\mathcal{L})$ of length at least three then there must exist some $v \in \mathring{V}(T)$ such that C is contained in $\Gamma(\mathcal{L}(v))$. Assume to the contrary that this is not the case, that is, there exists some cycle $C : u_1, u_2, \ldots, u_l, u_{l+1} = u_1$, $l \geq 3$, in $\Gamma(\mathcal{L})$ such that, for all $v \in \mathring{V}(T)$, we have that C is not a cycle in $\Gamma(\mathcal{L}(v))$. Without loss of generality, we may assume that C is of minimal length. For all $i \in \langle l \rangle$, put $v_i = lca_T(u_i, u_{i+1})$. Then, by the construction of $\Gamma(\mathcal{L})$, we have for all such i that $u_i u_{i+1}$ is an edge in $\Gamma(\mathcal{L}(v_i))$ and, by the minimality of C, that $v_i \neq v_j$ for all $i, j \in \langle l \rangle$ distinct. Put $Y = V(C)$ and let $T' = T|_Y$ denote the Y-tree obtained by restricting T to Y. Note that $lca_T(u_i, u_{i+1}) = lca_{T'}(u_i, u_{i+1})$ holds for all $i \in \langle l \rangle$. Thus, the map $\phi : E(C) \to \mathring{V}(T')$ defined by putting $u_i u_{i+1} \mapsto lca_T(u_i, u_{i+1})$, $i \in \langle l \rangle$, is well-defined. Since $|E(C)| = l$ and for any finite set Z with three or more elements a Z-tree has at most $|Z| - 1$ interior vertices, it follows that there exist $i, j \in \langle l \rangle$ distinct such that $\phi(u_i, u_{i+1}) = \phi(u_j, u_{j+1})$. Consequently, $v_i = lca_T(u_i, u_{i+1}) = lca_T(u_j, u_{j+1}) = v_j$ which is impossible and thus proves the claim. Combined with Lemma 10 (ii) and (iii), it follows that $\Gamma(\mathcal{L})$ is a connected block graph. That the blocks of $\Gamma(\mathcal{L})$ are of the required form is an immediate consequence of the construction of $\Gamma(\mathcal{L})$. □

To be able to establish that $\mathcal{L}_{(T,f)}(v)$ is indeed a distinguished minimal topological lasso for T and f as above, we require a further concept. Suppose $A, B \subseteq X$ are two distinct non-empty subsets of X. Then A and B are said to be *compatible* if $A \cap B \in \{\emptyset, A, B\}$. As is well-known (see e. g. [47, 147]), for any X-tree T' and any two vertices $v, w \in V(T')$ the subsets $L(v)$ and $L(w)$ of X are compatible.

Theorem 19. *Suppose T is a non-degenerate X-tree, σ is a total ordering of X and $f : cl(T) \to X$ is a cluster marker map for T and σ. Then $\mathcal{L}_{(T,f)}$ is a distinguished minimal topological lasso for T.*

Proof. For all $v \in \mathring{V}(T)$ put $\mathcal{L}(v) = \mathcal{L}_{(T,f)}(v)$ and put $\mathcal{L} = \mathcal{L}_{(T,f)}$. In view of Proposition 3 and Lemma 10 (i), it suffices to show that $\Gamma(\mathcal{L})$ is claw-free. Assume to the contrary that this is not the case and that there exists some $x \in X$ that is contained in the vertex set of $m \geq 3$ blocks $A_1, \dots, A_m$ of $\Gamma(\mathcal{L})$. Then, by Proposition 3, there exist distinct interior vertices $v_1, \dots, v_m$ of T such that, for all $i \in \langle m \rangle$, we have $V(A_i) = \bigcup \mathcal{L}(v_i) \subseteq L(v_i)$. Since for all $v, w \in V(T)$ distinct, the sets $L(v)$ and $L(w)$ are compatible, it follows that there exists a path P from ρ_T to x that contains the vertices $v_1, \dots, v_m$ in its vertex set. Without loss of generality we may assume that $m = 3$ and that, starting at ρ_T and moving along P the vertex v_1 is encountered first then v_2 and then v_3. Note that $cl_{L(v_i)}(T) \neq \emptyset$, for $i = 1, 2$. Since T is a tree and so x can neither be adjacent with v_1 nor with v_2 it follows that there must exist for $i = 1, 2$ some $B_i \in cl_{L(v_i)}(T)$ such that $x = f(B_i)$. But this is impossible as $B_2 \in cl_{L(v_1)}(T)$ and so $f(B_1) \neq f(B_2)$ as f is a cluster marker map for T and σ. □

6.5 Characterisation of distinguished minimal topological lassos

In this section, we establish the converse of Theorem 19 which allows us to characterise distinguished minimal topological lasso of non-degenerate X-trees. We start with a well-known construction for associating an unrooted tree to a connected block graph (see e. g. [42]). Suppose that G is a connected block graph. Then we denote by T_G the (unrooted) tree associated to G with vertex set $Cut(G) \cup Block(G)$ and whose edges are of the from $\{a, B\}$ where $a \in Cut(G)$, $B \in Block(G)$ and $a \in B$. Note that if a vertex $v \in V(T_G)$ is a leaf of T_G then $v \in Block(G)$.

Suppose T is a non-degenerate X-tree and $\mathcal{L}$ is a distinguished minimal topological lasso for T. Let v denote an interior vertex of T whose children are $v_1 \dots, v_l$ where $l = |ch(v)|$. Then Corollary 3 combined with Proposition 2 implies that for all $i \in \langle l \rangle$ there exists a unique leaf $x_i \in L(v_i)$ of T such that, for all $i, j \in \langle l \rangle$ distinct, $x_i x_j \in \mathcal{L}$ and $\{x_1, \dots, x_l\} = V(B_v)$. Since $\Gamma(\mathcal{L})$ is claw-free, every vertex of B_v is contained in at most one further block of $\Gamma(\mathcal{L})$. Thus, if $w \in V(B_v)$ and $w \in V(B)$ holds too for some block $B \in Block(\Gamma(\mathcal{L}))$ distinct from B_v then w must be a cut vertex of $\Gamma(\mathcal{L})$. For every vertex $v' \in \mathring{V}(T)$ that is the child of some vertex $v \in \mathring{V}(T)$, we denote the unique element $x \in L(v')$ contained in $V(B_v)$ by $c_{B_{v'}}$ in case $x \in Cut(\Gamma(\mathcal{L}))$. Note that it is not difficult to observe that, in the tree $T_{\Gamma(\mathcal{L})}$, the vertex $c_{B_{v'}}$ is the vertex adjacent with B_v that lies on the path from B_v to $B_{v'}$.

The following result lies at the heart of Theorem 20 and establishes a crucial relationship between the non-root interior vertices of T and the cut vertices of $\Gamma(\mathcal{L})$.

Lemma 11. *Suppose T is an X-tree and $\mathcal{L}$ is a distinguished minimal topological lasso for T. Then the map*

$$\theta : \mathring{V}(T) - \{\rho_T\} \to Cut(\Gamma(\mathcal{L})) : \;\; v \mapsto c_{B_v}$$

is bijective.

Proof. Clearly, θ is well-defined and injective. To see that θ is bijective let $T^{-}_{\Gamma(\mathcal{L})}$ denote the tree obtained from $T_{\Gamma(\mathcal{L})}$ by suppressing all degree two vertices. Then $Block(\Gamma(\mathcal{L})) = V(T^{-}_{\Gamma(\mathcal{L})})$ and Corollary 3 implies that $|Block(\Gamma(\mathcal{L}))| = |\mathring{V}(T)|$ as $\Gamma(\mathcal{L})$ is a block graph. Since $\Gamma(\mathcal{L})$ is claw-free, we clearly also have $|Cut(\Gamma(\mathcal{L}))| = |E(T^{-}_{\Gamma(\mathcal{L})})|$. Combined with the fact that f $|V(T')| = |E(T')| + 1$ holds for every tree T', it follows that $|Cut(\Gamma(\mathcal{L}))| = |Block(\Gamma(\mathcal{L}))| - 1 = |\mathring{V}(T)| - 1 = |\mathring{V}(T) - \{\rho_T\}|$. Thus, θ is bijective. □

Armed with this result, we are now ready to establish the converse of Theorem 19 which yields the aforementioned characterisation of distinguished minimal topological lassos of non-degenerate X-trees.

Theorem 20. *Suppose T is a non-degenerate X-tree and $\mathcal{L}$ is a set of cords of X. Then $\mathcal{L}$ is a distinguished minimal topological lasso for T if and only if there exists a total ordering σ of X and a cluster marker map f for T and σ such that $\mathcal{L}_{(T,f)} = \mathcal{L}$.*

Proof. Assume first that σ is some total ordering of X and that $f : cl(T) \to X$ is a cluster marker map for T and σ. Then, by Theorem 19, $\mathcal{L}_{(T,f)}$ is a distinguished minimal topological lasso for T.

Conversely assume that $\mathcal{L}$ is a distinguished minimal topological lasso for T and consider an embedding of T into the plane. By abuse of terminology, we will refer to this embedding of T also as T. We start with defining a total ordering σ of X. To this end, we first define a map $t : \mathring{V}(T) - \{\rho_T\} \to \mathbb{N}$ by setting, for all $v \in \mathring{V}(T) - \{\rho_T\}$, $t(v)$ to be the length of the path from ρ_T and v. Put $h = \max\{t(v) : v \in \mathring{V}(T) - \{\rho_T\}\}$ and note that $h \geq 1$ as T is non-degenerate.

Starting at the left most interior vertex v of T for which $t(v) = h$ holds and moving, for all $l \in \langle h \rangle$, from left to right, we enumerate all interior vertices of T but the root. We next put $n = |X|$ and $X = \langle n \rangle$ and relabel the elements in X such that when traversing the circular ordering induced by T on $X \cup \{\rho_T\}$ in a counter-clockwise fashion we have $\rho_T, 1, 2, 3, \ldots, n, \rho_T$. To reflect this with regards to $\mathcal{L}$, we relabel the elements of the cords in $\mathcal{L}$ accordingly and denote the resulting distinguished minimal topological lasso for T also by $\mathcal{L}$.

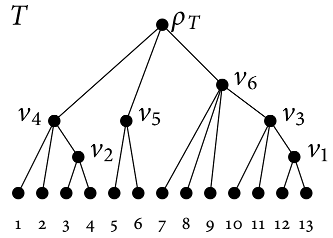


Figure 6.3: For $X = \langle 13 \rangle$ and the depicted X-tree T, the enumeration of the interior vertices of T considered in the proof of Theorem 20 is indicated in (i). With regards to this enumeration and the distinguished minimal topological lasso $\mathcal{L}$ for T pictured in the form of $\Gamma(\mathcal{L})$ in (ii), the total ordering σ of X considered in that proof restricted to the elements in $\{\theta(v_1), \dots, \theta(v_6)\}$ is $3, 5, 12, 1, 10, 7$.

By Lemma 11, the map $\theta : \mathring{V}(T) - \{\rho_T\} \to Cut(\Gamma(\mathcal{L}))$ defined in that lemma is bijective. Put $m = |Cut(\Gamma(\mathcal{L}))|$ and let $v_1, v_2, \dots, v_m$ denote the enumeration of the vertices in $\mathring{V}(T) - \{\rho_T\}$ obtained above. Also, set $Y = X - \{\theta(v_i) : i \in \langle m \rangle\}$. Let $y_1, y_2, \dots, y_l$ denote an arbitrary but fixed total ordering of the elements of Y where $l = |Y|$. Then we define σ to be the total ordering of X given by

$$\sigma : \theta(v_1), \theta(v_2), \dots, \theta(v_{i-1}), \theta(v_i), \theta(v_{i+1}), , \dots, \theta(v_m), y_1, y_2, \dots, y_l$$

where $\theta(v_1)$ is the minimal element and y_l is the maximal element. Note that if $v \in \mathring{V}(T)$ is the parent of a pseudo-cherry C of T then $\theta(v) = \min_\sigma C$.

We briefly interrupt the proof of the theorem to illustrate these definitions by means of an example. Put $X = \langle 13 \rangle$ and consider the X-tree T depicted in Figure 6.3 (i) (ignoring the labelling of the interior vertices for the moment) and the distinguished minimal topological lasso $\mathcal{L}$ for T pictured in the form of $\Gamma(\mathcal{L})$ in Figure 6.3 (ii). Then the labelling of the interior vertices of T gives the enumeration of those vertices considered in the proof of Theorem 20. The total ordering σ of X restricted to the elements in $\{\theta(v_1), \dots, \theta(v_6)\}$ is $3, 5, 12, 1, 10, 7$.

Returning to the proof of the theorem, we claim that the map $f : cl(T) \to X$ given, for all $A \in cl(T)$, by setting $f(A) = \theta(lca(A))$ is a cluster marker map for T and σ where for all such A we put $lca(A) = lca_T(A)$. Indeed, suppose $A \in cl(T)$. Then

$\theta(lca(A)) = c_{B_{lca(A)}} \in L(lca(A))$ holds by construction. We distinguish between the cases that $cl_A(T) \neq \emptyset$ and that $cl_A(T) = \emptyset$. If $cl_A(T) \neq \emptyset$ then since θ is bijective it follows that $\theta(lca(A)) \neq \theta(v)$ holds for all descendants $v \in \mathring{V}(T)$ of $lca(A)$. Combined with the definition of σ, we obtain $f(A) = \theta(lca(A)) = \min_\sigma(A - \{\theta(lca(D)) : D \in cl_A(T)\}) = \min_\sigma(A - \{f(D) : D \in cl_A(T)\})$, as required. If $cl_A(T) = \emptyset$ then, as was observed above, $f(A) = \theta(lca(A)) = \min_\sigma A$. Thus, f is a cluster marker map for T and σ, as claimed.

It remains to show that $\mathcal{L}_{(T,f)} = \mathcal{L}$. To see this note first that, by Theorem 19, $\mathcal{L}_{(T,f)}$ is a distinguished minimal topological lasso for T. Since Lemma 8 implies that any two minimal topological lasso for T must be of the same size and thus $|\mathcal{L}_{(T,f)}| = |\mathcal{L}|$ holds, it therefore suffices to show that $\mathcal{L} \subseteq \mathcal{L}_{(T,f)}$. Suppose $a, b \in X$ distinct such that $ab \in \mathcal{L}$. Then there exists some interior vertex $v \in \mathring{V}(T)$ such that $v = lca_T(a, b)$. Hence, $a, b \in V(B_v)$. We claim that $ab \in \mathcal{L}_{(T,f)}(v)$. To establish this claim, we distinguish between the cases that (i) $a \in ch(v)$ and (ii) that $a \notin ch(v)$.

Assume first that Case (i) holds, that is, a is a child of v. If $b \in ch(v)$ then the claim is an immediate consequence of the definition of $\mathcal{L}_{(T,f)}(v)$. So assume that $b \notin ch(v)$. Let $v' \in \mathring{V}(T)$ denote the child of v for which $b \in L(v)$ holds. Then $b = c_{B_{v'}} = \theta(v') = f(L(v'))$ follows by the observation preceding Lemma 11 combined with the fact that $b \in V(B_v)$. Hence, $ab = af(L(v')) \in \mathcal{L}_{(T,f)}(v)$, as claimed.

Assume next that Case (ii) holds, that is, a is not a child of v. In view of the previous subcase it suffices to consider the case that $b \notin ch(v)$. Let $v', v'' \in \mathring{V}(T)$ denote the children of v such that $a \in L(v')$ and $b \in L(v'')$. Then, again by the observation preceding Lemma 11 combined with the fact that $a, b \in V(B_v)$, we have $a = c_{B_{v'}} = \theta(v') = f(L(v'))$ and $b = c_{B_{v''}} = \theta(v'') = f(L(v''))$ and so $ab = f(L(v'))f(L(v'')) \in \mathcal{L}_{(T,f)}(v)$ follows, as claimed. This concludes the proof of the claim and thus the proof of the theorem. □

Recall that Theorem 16 tells us that a set $\mathcal{L}$ of cords of X is an equidistant lasso for an X-tree T if and only if, for every vertex $v \in \mathring{V}(T)$, the graph $G(\mathcal{L}, v)$ has at least one edge.

Since for σ some total ordering of X and $f : \mathring{V}(T) - \{\rho_T\} \to X$ a cluster marker map for T and σ the graphs $G(\mathcal{L}_{(T,f)}, v)$ clearly satisfy this property for all $v \in \mathring{V}(T)$, it follows that $\mathcal{L}_{(T,f)}$ is also an equidistant lasso for T and thus a strong lasso for T.

Defining a strong lasso $\mathcal{L}$ of an X-tree to be *minimal* in analogy to when a topological lasso is minimal, Theorem 20 implies

Corollary 4. *Suppose T is a non-degenerate X-tree, $\mathcal{L}$ is a set of cords of X, σ is a total ordering of X, and $f : cl(T) \to X$ is a cluster marker map for T and σ. Then $\mathcal{L}_{(T,f)}$ is a minimal strong lasso for T.*

6.6 Heredity of distinguished minimal topological lassos

In this section, we turn our attention to the problems of characterising when a distinguished minimal topological lasso of an X-tree T induces a distinguished minimal topological lasso for a subtree of T and, conversely when distinguished minimal topological lassos of X-trees can be combined to form a distinguished minimal topological lasso of a supertree for those trees (see e. g. [15] for more on such trees). This will also allow us to partially answer the rooted analogue of a question raised in [46] for supertrees within the unrooted framework. To make this more precise, we require further terminology. Suppose $\mathcal{L}$ a set of cords of X and $Y \subseteq X$ is a non-empty subset. Then we set

$$\mathcal{L}|_Y = \{ab \in \mathcal{L} : a, b \in Y\}.$$

Clearly, $\Gamma(\mathcal{L}|_Y)$ is the subgraph of $\Gamma(\mathcal{L})$ induced by Y but $Y = \bigcup \mathcal{L}|_Y$ need not hold. Moreover, if $\mathcal{L}$ is a minimal topological lasso for an X-tree T and $|Y| \geq 3$ such that every interior vertex of T is also an interior vertex of $T|_Y$ then Theorem 16 implies that $\mathcal{L}|_Y$ is a minimal topological lasso for $T|_Y$. In particular, $\Gamma(\mathcal{L}|_Y)$ must be connected in this case. The next result is a strengthening of this observation.

Theorem 21. *Suppose T is an X-tree, $\mathcal{L}$ is a distinguished minimal topological lasso for T,* *and $Y \subseteq X$ is a subset of size at least three. Then $\mathcal{L}|_Y$ is a distinguished minimal topological lasso for $T|_Y$ if and only if $\Gamma(\mathcal{L}|_Y)$ is connected.*

Proof. Assume first that $\mathcal{L}|_Y$ is a distinguished minimal topological lasso for $T|_Y$. Then, by Proposition 1, $\Gamma(\mathcal{L}|_Y)$ is connected.

Conversely, assume that $\Gamma(\mathcal{L}|_Y)$ is connected. Then the statement clearly holds if T is the star tree on X. So assume that T is non-degenerate. Let $Y \subseteq X$ be of size at least three and assume first that $T|_Y$ is the star tree on Y. We claim that $\Gamma(\mathcal{L}|_Y)$ is a clique. Assume to the contrary that this is not the case, that is, there exist elements $y, y' \in Y$ distinct such that $yy' \notin \mathcal{L}$. Since $\Gamma(\mathcal{L}|_Y)$ is connected, there must exist a path $P : x_1 = y, x_2, \ldots, x_l = y'$, $l \geq 2$, in $\Gamma(\mathcal{L}|_Y)$ from y to y'. Since the vertex set of $\Gamma(\mathcal{L}|_Y)$ is Y, it follows that $X' = \{x_1, x_2, \ldots, x_l\} \subseteq Y$. Combined with the fact that $lca_T(x, x') = lca_T(Y)$ holds for all $x, x' \in X'$ distinct as $T|_Y$ is a star tree on Y, we obtain $X' \subseteq V(B_{lca_T(Y)})$. Thus, $yy' \in \mathcal{L}$ which is impossible and thus proves the claim. That $\mathcal{L}|_Y$ is a distinguished minimal topological lasso for $T|_Y$ is a trivial consequence.

So assume that $T|_Y$ is non-degenerate. Since $\mathcal{L}$ is a distinguished minimal topological lasso for T, Theorem 20 implies that there exists a total ordering ω of X and a cluster marker map $f_\omega : cl(T) \to X$ for T and ω such that $\mathcal{L} = \mathcal{L}_{(T, f_\omega)}$. Moreover, Lemma 10 (iv) implies that the cut-vertices of $\Gamma(\mathcal{L})$ are of the form $f_\omega(L_T(v))$ where $v \in \mathring{V}(T)$.

To see that $\mathcal{L}|_Y$ is a distinguished minimal topological lasso for $T|_Y$ and some total ordering of Y note first that the restriction σ of ω to Y induces a total ordering of Y. Furthermore, the aforementioned form of the cut-vertices of $\Gamma(\mathcal{L})$ combined with the assumption that $\Gamma(\mathcal{L}|_Y)$ is connected implies that, for all $A \in cl(T)$ with $A \cap Y \neq \emptyset$, we must have $f_\omega(A) \in Y$. For all $A \in cl(T|_Y)$ denote by A^T the set-inclusion minimal superset of A contained in $cl(T)$. Then since f_ω is a cluster marker map for T and ω it follows that the map

$$f_\sigma : cl(T|_Y) \to Y : A \mapsto f_\omega(A^T)$$

is a cluster marker map for $T|_Y$ and σ. By Theorem 20 it now suffices to establish that

$\mathcal{L}|_Y = \mathcal{L}_{(T|_Y,f_\sigma)}$. Since both $\mathcal{L}|_Y$ and $\mathcal{L}_{(T|_Y,f_\sigma)}$ are minimal topological lassos for $T|_Y$ and so $|\mathcal{L}|_Y| = |\mathcal{L}_{(T|_Y,f_\sigma)}|$ is implied by Lemma 8 it suffices to show that $\mathcal{L}|_Y \subseteq \mathcal{L}_{(T|_Y,f_\sigma)}$.

Suppose $ab \in \mathcal{L}|_Y$, that is, $ab \in \mathcal{L}$ and $a, b \in Y$. Since Y is the leaf set of $T|_Y$, there must exist a vertex $v \in \mathring{V}(T|_Y)$ such that $v = lca_{T|_Y}(a,b)$. Clearly, $v \in \mathring{V}(T)$. If a and b are both adjacent with v in T then a and b are also adjacent with v in $T|_Y$. Thus $ab \in \mathcal{L}_{(T|_Y,f_\sigma)}(v)$ in this case. So assume that at least one of a and b is not adjacent with v in T. Without loss of generality let a denote that vertex. Then since $ab \in \mathcal{L} = \mathcal{L}_{(T,f_\omega)}$, it follows that there must exist a unique child $v' \in \mathring{V}(T)$ of v such that $a \in L_T(v')$ and $a = f_\omega(L_T(v'))$. Hence, $a \in V(B_v)$ and a cut-vertex of $\Gamma(\mathcal{L})$.

We claim that $v' \in \mathring{V}(T|_Y)$. Assume for contradiction that $v' \notin \mathring{V}(T|_Y)$. Then since f_ω is a cluster marker map for T and ω, it follows that a cannot be a cut vertex in $\Gamma(\mathcal{L}|_Y)$. Since $\Gamma(\mathcal{L})$ is a claw-free block graph, no edge in the unique block $B' \in Block(\Gamma(\mathcal{L})) - \{B_v\}$ that also contains a in its vertex set can therefore be incident with a in $\Gamma(\mathcal{L}|_Y)$. Since $\Gamma(\mathcal{L}|_Y)$ is assumed to be connected, to obtain the required contradiction it now suffices to show that there exists some $c \in Y \cap L_T(v')$ distinct from a such that every path from c to b in $\Gamma(\mathcal{L})$ crosses a. But this is a consequence of the facts that v is not the parent of a in $T|_Y$ and, implied by Proposition 1, that the subgraph $\Gamma_{v'}(\mathcal{L})$ of $\Gamma(\mathcal{L})$ induced by $L_T(v')$ is the connected component of $\Gamma(\mathcal{L})$ containing a obtained from $\Gamma(\mathcal{L})$ by deleting all edges in B_v that are incident with a. This concludes the proof of the claim

To conclude the proof of the theorem, note that if b is adjacent with v in $T|_Y$ then $ab = f_\omega(L_T(v'))b = f_\omega((L_{T|_Y}(v'))^T)b = f_\sigma(L_{T|_Y}(v'))b \in \mathcal{L}_{(T|_Y,f_\sigma)}(v) \subseteq \mathcal{L}_{(T|_Y,f_\sigma)}$. If b is not adjacent with v in $T|_Y$ then there exists a child $v'' \in \mathring{V}(T)$ of v such that $b = f_\omega(L_T(v''))$. In view of the previous claim, we have $v'' \in \mathring{V}(T|_Y)$. But now arguments similar to the ones used before imply that $ab \in \mathcal{L}_{(T|_Y,f_\sigma)}(v) \subseteq \mathcal{L}_{(T|_Y,f_\sigma)}$. □

We now turn our attention to supertrees which are formally defined as follows. Suppose $\mathcal{T} = \{T_1, \dots, T_l\}$, $l \geq 1$, is a set of Y_i-trees T_i with $Y_i \subseteq X$ and $|Y_i| \geq 3$, $i \in \langle l \rangle$, and T is an X-tree. Then T is a called a *supertree* of $\mathcal{T}$ if T displays every tree in $\mathcal{T}$ where we say that some X-tree T *displays* some Y-tree T' for $Y \subseteq X$ with $|Y| \geq 3$ if $T|_Y$ and T' are equivalent. More precisely, we have the following result which relies on the fact that in case $\mathcal{L}$ is a distinguished minimal topological lasso for a *binary* X-tree T, that is, every vertex of T but the leaves has two children, $\Gamma(\mathcal{L})$ must be a path. In particular, $\mathcal{L}$ induces a total ordering of the elements in X in this case. For $Y \subseteq X$ a non-empty subset of X, we denote the maximal and minimal element in Y with regards to that ordering by $\min_{\mathcal{L}}(Y)$ and $\max_{\mathcal{L}}(Y)$, respectively.

Corollary 5. *Suppose X' and X'' are two non-empty subsets of X such that $X = X' \cup X''$ and $X' \cap X'' \neq \emptyset$ and T' and T'' are X'-trees and X''-tree, respectively. Suppose also that $\mathcal{L}'$ and $\mathcal{L}''$ are distinguished minimal topological lassos for T' and T'', respectively, such that $\mathcal{L}'|_{X' \cap X''} = \mathcal{L}''|_{X' \cap X''}$ and $\Gamma(\mathcal{L}''|_{X' \cap X''})$ is connected. If T is a binary X-tree that displays both T' and T'' then $\mathcal{L} = \mathcal{L}' \cup \mathcal{L}''$ is a distinguished minimal topological lasso for T if and only if* $\min_{\mathcal{L}'}(X' \cap X'') \in \{\min_{\mathcal{L}'}(X'), \min_{\mathcal{L}''}(X'')\}$ *and* $\max_{\mathcal{L}'}(X' \cap X'') \in \{\max_{\mathcal{L}'}(X'), \max_{\mathcal{L}''}(X'')\}$.

Continuing with the assumptions of Corollary 5, we also have that if

$$\min_{\mathcal{L}'}(X' \cap X'') \in \{\min_{\mathcal{L}'}(X'), \min_{\mathcal{L}''}(X'')\}$$

and

$$\max_{\mathcal{L}'}(X' \cap X'') \in \{\max_{\mathcal{L}'}(X'), \max_{\mathcal{L}''}(X'')\}$$

holds then $\mathcal{L}' \cup \mathcal{L}''$ is a (minimal) strong lasso for T as every minimal topological lasso for an X-tree is also an equidistant lasso for that tree. However, not all strong lassos for T are of this form. An example for this is furnished for $X' = \{a, c, d\}$ and $X'' = \{a, b, c\}$ by the X'-tree T', the X''-tree T'' and the $X' \cup X''$-tree T depicted in Figure 6.4 along with the set $\mathcal{L}' = \{cd\}$ and $\mathcal{L}'' = \{ab, bc\}$ of cords of X' and X'', respectively. Clearly, T is a supertree of $\{T', T''\}$ and $\mathcal{L} = \mathcal{L}' \cup \mathcal{L}''$ is a strong lasso for T but $\mathcal{L}'$ is not even an equidistant lasso for T'. Investigating further the interplay between minimal topological

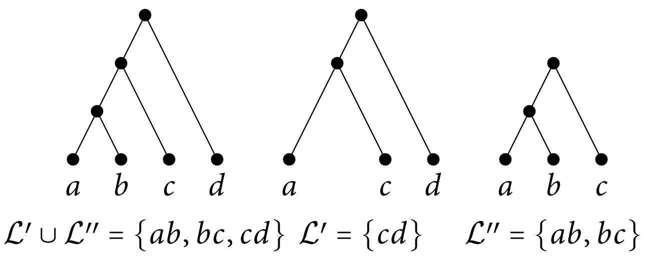

a b c d
a c d
a b c
$\mathcal{L}' \cup \mathcal{L}'' = \{ab, bc, cd\}$
$\mathcal{L}' = \{cd\}$
$\mathcal{L}'' = \{ab, bc\}$

Figure 6.4: For $X' = \{a, c, d\}$ and $X'' = \{a, b, c\}$ the $X' \cup X''$-tree T is a supertree for the depicted X' and X'' trees T' and T'', respectively. Clearly, $\mathcal{L}' = \{cd\}$ and $\mathcal{L}'' = \{ab, bc\}$ are sets of cords of X' and X'', respectively, and $\mathcal{L} = \mathcal{L}' \cup \mathcal{L}''$ is a strong lasso for T but $\mathcal{L}'$ is not even an equidistant lasso for T'.

lassos for X-trees and minimal topological lassos for supertrees that display them might therefore be of interest.

We conclude with returning to Figure 6.1 which depicts two non-equivalent X-trees that are topologically lassoed by the same set $\mathcal{L}$ of cords of X. In fact, $\mathcal{L}$ is even a minimal topological lasso for both of them. Understanding better the relationship between X-trees that are topologically lassoed by the same set of cords of X might also be of interest to study further.

6.7 Conclusion

In this chapter we introduced a special type of lasso called a distinguished minimal topological lasso for which the associated $\Gamma(\mathcal{L})$ graph is a claw-free block graph. We have shown that for such a lasso $\mathcal{L}$ each block in $\Gamma(\mathcal{L})$ corresponds to one interior vertex in the tree (Theorem 17 and Corollary 3). A distinguished minimal topological lasso is special in that for any X-tree a minimal topological lasso for it can be transformed into a distinguished minimal topological lasso by repeated application of a simple rule. We characterised these lassos in terms of a cluster marker map and have given a criterion for inheritance by subtrees and supertrees.

Chapter 7

Conclusion and Further Work

In this thesis we have investigated several problems related to analysing distances using techniques known as clustering. With regards to partitional clustering we have seen that a central task is to define a criterion with which to determine the suitability of any particular clustering solution. We have an intuitive sense of what we would like to obtain in a clustering, that is one where the clusters are tightly packed and well separated, but we have shown that even very similar criteria which have been used for this purpose can produce wildly different results in the worst case (Section 3.5). We have also shown that the widely used centroid-distance criterion is not simultaneously a criterion for homogeneity and separation under the homogeneous Euclidean-overlap metric making it less useful as a general purpose clustering criterion.

Even if the sum-of-squares criteria are deemed to be good enough, it is now known that clustering under the centroid-distance criterion is an NP-hard problem in Euclidean space [4, 106]. We have shown that the associated decision problem is NP-complete even under a simple p-valued metric where $p \geq 3$. However, the problem is solvable in polynomial time for a 2-valued metric. Similarly, we have shown that the decision problem associated with all-squares clustering is NP-complete both with Euclidean space and with a p-valued metric when $p \geq 3$. The problem is solvable in polynomial time with a 2-valued metric when the dataset is a set, but the problem remains NP-complete even for a 2-valued metric when the dataset is a multiset (Section 3.3).

In the process of showing the worst case performance of the two sum-of-squares criteria we have developed a new metric for comparing clusterings called the assignment metric. In general this metric allows us to compare sets whose elements are themselves in a metric space. This allows us to distinguish clusterings by using the underlying metric space in a way which is not possible with other metrics. It is easily extendable to any metric, as well as to multiset and fuzzy clusterings (Section 3.4).

Our findings reinforce the view that partitional clustering is very difficult to do exactly. It is unclear if it even makes sense to try to solve clustering problems exactly since selection of a criterion is itself so difficult. Even so, heuristics for partitional clustering can often do a very good job when an exact solution to the problem is not required.

Our findings for hierarchical clustering are more positive. We extended the problem of simply building a hierarchy of clusters from a distance on a set X to that of building an edge-weighted X-tree. We then investigated the problem of building such a tree when only a partial distance on X is given. The theory of lassos allows us to decide whether a given set of distances over X is enough to uniquely determine the tree, with regards to topology and edge-weighting, that they came from.

In Chapter 5, we turned our attention to the problem of reconstructed an equidistant X-tree from partial distance information. We have developed an algorithm called LASSO for building a tree from partial distance information that returns a lasso and the unique equidistant tree lassoed by it. We showed using a simulation study that it is possible to recover much of the tree even when faced with a large number of missing distances. Our algorithm is resistant to noise in the data making it applicable to real biological datasets. We show that its performance on such a dataset is very good when faced with missing distances (Section 5.3.2).

We also illustrated the applicability of LASSO in a supertree context (Section 5.3.3). As part of this we compared LASSO with modified MINCUTSUPERTREE, a well known supertree algorithm, using the normalised Robinson-Foulds distance. We showed that the supertree produced by our method was closer to both of the original trees than the one produced by modified MinCutSupertree. In addition, since we are using distances our method produces an edge-weighted supertree while the MinCutSupertree requires weighting by another method.

In addition to the directions of further work outlined in Chapters 3, 6 and 5 there are number of interesting problems which suggest themselves. There are still many questions with regards to the computational complexity of the partitional clustering problems. Namely, we wonder whether it is possible to solve in polynomial time any of the approximation problems associated to the sum-of-squares optimisation problems or, in other words, can a near optimal solution be found to within some bounds of the optimal? Although we now know that exactly solving the sum-of-squares clustering problems is intractable unless $P = NP$, there have been several exact polynomial solutions for restricted versions of the problem. One could attempt to extend these solutions to accept larger input sizes thereby providing exact solutions in many practical cases.

The Lasso algorithm has $O(|\mathcal{L}_D|^2)$ runtime complexity, where $\mathcal{L}$ is the set of given distances on X. This means it is more applicable to cases where a given distance matrix is very sparse. The best case is when a minimal topological lasso of a binary tree is used as input since this results in $O(|X|^2)$ runtime. When a complete distance matrix is used then we have $O(|X|^4)$ runtime. In this case, other reconstruction techniques with $O(|X|^3)$ or $O(|X|^2)$ runtime for a complete matrix become more attractive. It may be possible, for example using ideas from the fast UPGMA algorithm (Section 4.3.2), to modify the algorithm such that the runtime complexity is smaller in terms of the size of X. This would allow the algorithm to remain applicable to complete distance matrices as well as sparse matrices.

Finally, in Chapter 6 we presented a study into the structural properties of lassos and, in particular, minimal topological lassos. Our investigation indicated a special type of minimal lasso called a distinguished minimal topological lasso. The $\Gamma(\mathcal{L})$ graph of such a lasso is a claw-free block graph which have been widely studied in many areas. These lassos are special since it is possible to turn any minimal topological lasso into a distinguished one by repeated application of a simple rule. We have characterised these lassos in terms of a cluster marker map and have given a criterion for when these lassos can be inherited by subtrees and supertrees, although it is easy to find examples where inheritance does not work for supertrees.

Chapter 8

Bibliography

[1] A.V. Aho, Y. Sagiv, T.G. Szymanski, and J.D. Ullman. Inferring a tree from lowest common ancestors with an application to the optimization of relational expressions. *SIAM Journal on Computing*, 10(3):405–421, 1981.

[2] D.H. Alexander, J. Novembre, and K. Lange. Fast model-based estimation of ancestry in unrelated individuals. *Genome Research*, 19:1655––1664, 2009.

[3] D. Aloise. *Exact Algorithms for Minimum Sum-of-Squares Clustering*. PhD thesis, Department of Mathematics and Industrial Engineering, École Polytechnique de Montréal, 2009.

[4] D. Aloise, A. Deshpande, P. Hansen, and P. Popat. NP-hardness of Euclidean sum-of-squares clustering. *Machine Learning*, 75:245–248, 2009. ISSN 0885-6125. doi: 10.1007/s10994-009-5103-0.

[5] P. Arabie and S.A. Boorman. Multidimensional scaling of measures of distance between partitions. *Journal of Mathematical Psychology*, 10(2):148–203, 1973.

[6] D. Arthur and S. Vassilvitskii. k-means++: The advantages of careful seeding. In *Proceedings of the Eighteenth Annual ACM-SIAM Symposium on Discrete Algorithms*, pages 1027–1035. Society for Industrial and Applied Mathematics, 2007.

[7] F.J. Ayala. Molecular clock mirages. *BioEssays*, 21:71––75, 1999.

[8] E. Bae, J. Bailey, and G. Dong. A clustering comparison measure using density profiles and its application to the discovery of alternate clusterings. *Data Mining and Knowledge Discovery*, 21:427–471, 2010. ISSN 1384-5810. doi: 10.1007/s10618-009-0164-z.

[9] G.H. Ball and D.J. Hall. A clustering technique for summarizing multivariate data. *Behavioral Science*, 12(2):153–155, 1967. ISSN 1099-1743. doi: 10.1002/bs.3830120210.

[10] A. Banerjee, S. Merugu, I.S. Dhillon, and J. Ghosh. Clustering with Bregman divergences. *The Journal of Machine Learning Research*, 6:1705–1749, 2005.

[11] E.M.L. Beale. Euclidean cluster analysis. *Bulletin of the International Statistical Institute*, 43(2):92–94, 1969.

[12] E.T. Bell. Exponential numbers. *The American Mathematical Monthly*, 41(7):411–419, 1934. ISSN 00029890.

[13] M. Bern and D. Eppstein. Approximation algorithms for geometric problems. In D.S. Hochbaum, editor, *Approximation Algorithms for NP-hard Problems*, pages 296–345. PWS Publishing Company, Boston,MA,USA, 1996.

[14] A. Bhattacharyya. On a measure of divergence between two statistical populations defined by their probability distributions. *Bulletin of the Calcutta Mathematical Society*, 35:99–109, 1943. ISSN 0008-0659.

[15] O.R.P. Bininda-Emonds. *Phylogenetic Supertrees: Combining Information to Reveal the Tree of Life*. Springer, 2004. ISBN 978-1-402-02328-6.

[16] R. Bouckaert, J. Heled, D. Kühnert, T. Vaughan, C.-H. Wu, D. Xie, M.A. Suchard, A. Rambaut, and A.J. Drummond. BEAST 2: A software platform for Bayesian evolutionary analysis. *PLoS Computational Biology*, 10:e1003537, 2014.

[17] D. Braun, J. Mayberry, A. Powers, and S. Schlicker. The geometry of the Hausdorff metric, 2003.

[18] M. Brinkmeyer, T. Griebel, and S. Boecker. Flipcut supertrees: Towards matrix representation accuracy in polynomial time. *Algorithmica*, 67:142–160, 2013.

[19] M.J. Brusco. A repetitive branch-and-bound procedure for minimum within-cluster sums of squares partitioning. *Psychometrika*, 71(2):347–363, 2006.

[20] D. Bryant. *Building Trees, Hunting for Trees, and Comparing Trees—Theory and Methods in Phylogenetic Analysis*. PhD thesis, University of Canterbury, 1997.

[21] D. Bryant, C. Semple, and M. Steel. Supertree methods for ancestral divergence dates and other applications. In O.R.P. Bininda-Emonds, editor, *Phylogenetic Supertrees*, volume 4 of *Computational Biology*, pages 129–150. Springer Netherlands, 2004. ISBN 978-1-4020-2329-3. doi: 10.1007/978-1-4020-2330-9_7.

[22] P. Brücker. On the complexity of clustering problems. *Optimization and Operations Research*, 157:45–54, 1978.

[23] F.T. Burbrink and T.A. Castoe. Molecular phylogeography of snakes. In *Snakes: Ecology and Conservation*, pages 38–77. Cornell University Press, 2009.

[24] F. Cao, J. Liang, and G. Jiang. An initialization method for the k-means algorithm using neighborhood model. *Computers & Mathematics with Applications*, 58(3): 474–483, 2009. ISSN 0898-1221. doi: 10.1016/j.camwa.2009.04.017.

[25] L.L. Cavalli-Sforza and A.W. Edwards. Phylogenetic analysis. models and estimation procedures. *The American Journal of Human Genetics*, 19(3:1):233–57, 1967.

[26] Z. Chen and X. Shixiong. k-means clustering algorithm with improved initial center. In *Second International Workshop on Knowledge Discovery and Data Mining*, pages 790–792. IEEE, 2009.

[27] P. Cimiano, A. Hotho, and S. Staab. Comparing conceptual, divise and agglomerative clustering for learning taxonomies from text. In *Proceedings of the 16th Eureopean Conference on Artificial Intelligence, ECAI'2004, including Prestigious Applicants of Intelligent Systems, PAIS 2004*, 2004.

[28] V.A. Confalonieri, A.S. Sequeira, L. Todaro, and J.C. Vilardi. Mitochondrial DNA and phylogeography of the grasshopper *trimerotropis pallidipennis* in relation to clinical distribution of chromosome polymorphisms. *Heredity*, 81:444–452, 1998.

[29] J. Corblet. *Étude iconographique sur l'arbre de Jessé*. Librairie Archéologique de Charles Blériot, Paris, 1860.

[30] G. Cormode and A. McGregor. Approximation algorithms for clustering uncertain data. In *Proceedings of the Twenty-seventh ACM SIGMOD-SIGACT-SIGART Symposium on Principles of Database Systems*, pages 191–200. ACM, 2008.

[31] A. Criscuolo and O. Gascuel. Fast NJ-like algorithms to deal with incomplete distance matrices. *BMC Bioinformatics*, 9(1):166, 2008. ISSN 1471-2105. doi: 10.1186/1471-2105-9-166.

[32] A. Criscuolo, V. Berry, E.J.P Douzery, and O. Gascuel. SDM: A fast distance-based approach for (super)tree building in phylogenomics. *Systematic Biology*, 55:740–755, 2006.

[33] C. Darwin. Notebook B: Transmutation of species, 1837–1838.

[34] C. Darwin. *On the Origin of Species by Means of Natural Selection, or The Preservation of Favoured Races in the Struggle for Life*. John Murray, London, 1859.

[35] P.J. De Groot, G.J. Postma, W.J. Melssen, and L.M.C. Buydens. Selecting a representative training set for the classification of demolition waste using remote NIR sensing. *Analytica Chimica Acta*, 392(1):67–75, 1999.

[36] A.R. De Leon and K.C. Carrière. A generalized Mahalanobis distance for mixed data. *Journal of Multivariate Analysis*, 92(1):174–185, 2005.

[37] C. De Rham. La classification hiérarchique ascendante selon la méthode des voisins réciproques. *Cahiers de l'Analyse des Données*, 5(2):135–144, 1980.

[38] G. de Soete. Ultrametric tree representations of incomplete dissimilarity data. *Journal of Classification*, 1(1):235–242, 1984.

[39] D. Defays. An efficient algorithm for a complete link method. *The Computer Journal*, 20(4):364–366, 1977. CLINK.

[40] M. Delattre and P. Hansen. Bicriterion cluster analysis. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, PAMI-2(4):277–291, 1980.

[41] M.M. Deza and E. Deza. *Encyclopedia of Distances*. Springer, 2009. ISBN 978-3-642-00233-5.

[42] R. Diestel. *Graph Theory*. Springer-Verlag, 2005. ISBN 978-3-540-26182-7.

[43] C. Ding and X. He. Cluster merging and splitting in hierarchical clustering algorithms. In *2002 IEEE International Conference on Data Mining*, pages 139–146. IEEE, 2002.

[44] G. Dobiński. Summirung der reihe $\sum n^m/n!$ für $m = 1, 2, 3, 4, 5, \ldots$. *Archiv der Mathematik und Physik*, 61:333–336, 1877.

[45] S. Doddi, M. Marathe, S. Ravi, D. Taylor, and P. Widmayer. Approximation algorithms for clustering to minimize the sum of diameters. *Algorithm Theory-SWAT 2000*, pages 41–51, 2000.

[46] A.W.M. Dress, K.T. Huber, and M. Steel. 'Lassoing' a phylogenetic tree I: Basic properties, shellings, and covers. *Journal of Mathematical Biology*, pages 1–29, 2011. ISSN 0303-6812. doi: 10.1007/s00285-011-0450-4.

[47] A.W.M. Dress, K.T. Huber, J. Koolen, V. Moulton, and A. Spillner. *Basic Phylogenetic Combinatorics*. Cambridge University Press, 2012. ISBN 978-0-521-76832-0.

[48] A.J. Drummond, S.Y.W. Ho, M.J. Phillips, and Rambaut A. Relaxed phylogenetics and dating with confidence. *PLoS Biology*, 4:e88, 2006.

[49] R. Durbin, S.R. Eddy, A. Krogh, and G. Mitchison. *Biological Sequence Analysis: Probabilistic Models of Proteins and Nucleic Acids*. Cambridge University Press, 1998. ISBN 978-0-521-62971-3.

[50] D.M. Endres and J.E. Schindelin. A new metric for probability distributions. *IEEE Transactions on Information Theory*, 49(7):1858–1860, July 2003. ISSN 0018-9448. doi: 10.1109/TIT.2003.813506.

[51] M. Erisoglu, N. Calis, and S. Sakallioglu. A new algorithm for initial cluster centers in k-means algorithm. *Pattern Recognition Letters*, 32(14):1701–1705, October 2011. ISSN 0167-8655. doi: 10.1016/j.patrec.2011.07.011.

[52] L. Euler. Solutio problematis ad geometriam situs pertinentis. *Commentarii academiae scientiarum Petropolitanae*, 8:128–140, 1741.

[53] B. Everitt. *Cluster Analysis*. Heinemann Educational Books Ltd, London, UK, 1980. ISBN 0-435-82296-9.

[54] M. Farach, S. Kannan, and T. Warnow. A robust model for finding optimal evolutionary trees. *Algorithmica*, 13(1-2):155–179, 1995.

[55] J. Felsenstein. *Inferring Phylogenies*. Sinauer Associates, 2003. ISBN 978-0-878-93177-4.

[56] J. Felsenstein. PHYLIP (phylogeny inference package), version 3.695, 2013. URL `http://evolution.genetics.washington.edu/phylip.html`.

[57] A.V. Fiacco and G.P. McCormick. The sequential unconstrained minimization technique for nonlinear programing, a primal-dual method. *Management Science*, 10(2):360–366, 1964.

[58] E. Forgy. Cluster analysis of multivariate data: Efficiency vs. interpretability of classifications. *Biometrics*, 21:768–769, 1965.

[59] E.B. Fowlkes and C.L. Mallows. A method for comparing two hierarchical clusterings. *Journal of the American Statistical Association*, 78(383):553–569, 1983. ISSN 01621459.

[60] A.L.N. Fred and A.K. Jain. Robust data clustering. *Computer Vision and Pattern Recognition, IEEE Computer Society Conference on*, 2:128, 2003. ISSN 1063-6919. doi: 10.1109/CVPR.2003.1211462.

[61] H.P. Friedman and J. Rubin. On some invariant criteria for grouping data. *Journal of the American Statistical Association*, pages 1159–1178, 1967.

[62] M.R. Garey and D.S. Johnson. *Computers and Intractability: A Guide to the Theory of NP-Completeness*. W. H. Freeman & Co., New York, NY, USA, 1979. ISBN 0-7167-1044-7.

[63] M.R. Garey, D.S. Johnson, and L. Stockmeyer. Some simplified NP-complete graph problems. *Theoretical Computer Science*, 1(3):237–267, 1976. ISSN 0304-3975. doi: 10.1016/0304-3975(76)90059-1.

[64] M.R. Garey, D.S. Johnson, and H.S. Witsenhausen. The complexity of the generalized Lloyd-max problem (corresp.). *Information Theory, IEEE Transactions on*, 28(2): 255–256, mar 1982. ISSN 0018-9448. doi: 10.1109/TIT.1982.1056488.

[65] O. Gascuel. BIONJ: An improved version of the NJ algorithm based on a simple model of sequence data. *Molecular Biology and Evolution*, 14:685–695, 1997.

[66] W. Gaul and M. Schader. Pyramidal classification based on incomplete dissimilarity data. *Journal of Classification*, 11(2):171–193, 1994.

[67] T.F. Gonzalez. On the computational complexity of clustering and related problems. *System Modeling and Optimization*, 38:174–182, 1982. ISSN 0170-8643.

[68] T.F. Gonzalez. Clustering to minimize the maximum intercluster distance. *Theoretical Computer Science*, 38:293–306, 1985. ISSN 0304-3975.

[69] I. Gronau and S. Moran. Optimal implementations of UPGMA and other common clustering algorithms. *Information Processing Letters*, 104(6):205–210, 2007.

[70] A. Guénoche and S. Grandcolas. Approximations par arbre d'une distance partielle. *Mathématiques, Informatique et Sciences Humaines*, 37(146):51–64, 1999.

[71] A. Guénoche, B. Leclerc, and V. Makarenkov. On the extension of a partial metric to a tree metric. *Discrete Mathematics*, 276(1):229–248, 2004.

[72] R.W. Hamming. Error detecting and error correcting codes. *The Bell System Technical Journal*, 29:147–160, 1950. ISSN 0005-8580.

[73] P. Hansen and M. Delattre. Complete-link cluster analysis by graph coloring. *Journal of the American Statistical Association*, pages 397–403, 1978.

[74] P. Hansen and B. Jaumard. Minimum sum of diameters clustering. *Journal of Classification*, 4:215–226, 1987. ISSN 0176-4268. doi: 10.1007/BF01896987.

[75] P. Hansen and B. Jaumard. Cluster analysis and mathematical programming. *Mathematical programming*, 79(1):191–215, 1997.

[76] F. Harary. *Graph Theory*. Addison-Wesley Publishing Co., 1972. ISBN 978-0-201-02787-7.

[77] J.A. Hartigan. *Clustering Algorithms*. Probability and Mathematical Statistics. John Wiley & Sons, Inc., 1975. ISBN 978-0-471-35645-5.

[78] J. He, M. Lan, C. Tan, S. Sung, and H. Low. Initialization of cluster refinement algorithms: A review and comparative study. In *Proceedings of 2004 IEEE International Joint Conference on Neural Networks*, volume 1. IEEE, 2004.

[79] M. Hellmuth, M. Hernandez-Rosales, K.T. Huber, V. Moulton, and P.F. Stadler. Orthology relations, symbolic ultrametrics, and co-graphs. *Journal of Mathematical Biology*, 66:399–420, 2013.

[80] K.T. Huber and G. Kettleborough. Distinguished minimal topological lassos. Submitted, 2014.

[81] K.T. Huber and A. Popescu. Lassoing and corralling rooted phylogenetic trees. *Bulletin of Mathematical Biology*, 75(3):444–465, 2013. ISSN 0092-8240. doi: 10.1007/s11538-013-9815-8. arXiv:1208.5594.

[82] K.T. Huber and M. Steel. Tree reconstruction from triplet cover distances. *The Electronic Journal of Combinatorics*, 21(2):2–15, 2014.

[83] L. Hubert and P. Arabie. Comparing partitions. *Journal of Classification*, 2:193–218, 1985. ISSN 0176-4268. doi: 10.1007/BF01908075.

[84] M. Inaba, N. Katoh, and H. Imai. Applications of weighted Voronoi diagrams and randomization to variance-based k-clustering. In *Proceedings of the Tenth Annual Symposium on Computational Geometry*, SCG '94, pages 332–339, New York, NY, USA, 1994. ACM. ISBN 0-89791-648-4. doi: 10.1145/177424.178042.

[85] A.K. Jain, M.N. Murty, and P.J. Flynn. Data clustering: A review. *ACM Computing Surveys (CSUR)*, 31(3):264–323, 1999. ISSN 0360-0300.

[86] S. Johnson. Hierarchical clustering schemes. *Psychometrika*, 32:241–254, 1967. ISSN 0033-3123. doi: 10.1007/BF02289588.

[87] J. Juan. Programme de classification hiérarchique par l'algorithme de la recherche en chaîne des voisins réciproques. *Cahiers de l'Analyse des Données*, 7(2):219–225, 1982.

[88] H.G. Kahrimanian. Analytical differentiation by a digital computer. Master's thesis, Temple University, Philadelphia, Pennsylvania, USA, May 1953.

[89] Richard M. Karp. Reducibility among combinatorial problems. In R. E. Miller and J. W. Thatcher, editors, *Complexity of Computer Computations*, pages 85–103. Plenum Press, New York, 1972.

[90] L. Kaufman and P.J. Rousseeuw. *Finding Groups in Data: An Introduction to Cluster Analysis*. John Wiley & Sons, Ltd., 1990. ISBN 978-0-471-73578-6.

[91] G. Kettleborough and V.J. Rayward-Smith. Optimising sum-of-squares measures for clustering multisets defined over a metric space. *Discrete Applied Mathematics*, 161(16-17):2499–2513, 2013. doi: 10.1016/j.dam.2013.04.015.

[92] G. Kettleborough, J. Dicks, I.N. Roberts, and K.T. Huber. Reconstructing (super)trees from data sets with missing distances: Not all is lost. *Molecular Biology and Evolution*, 2015. doi: 10.1093/molbev/msv027.

[93] S.S. Khan and A. Ahmad. Cluster center initialization algorithm for k-means clustering. *Pattern Recognition Letters*, 25(11):1293–1302, August 2004. ISSN 0167-8655. doi: 10.1016/j.patrec.2004.04.007.

[94] G. Klein and J.E. Aronson. Optimal clustering: A model and method. *Naval Research Logistics*, 38(3):447–461, 1991.

[95] G.J. Klir and B. Yuan. *Fuzzy Sets and Fuzzy Logic*, volume 4. Prentice Hall New Jersey, 1995.

[96] D.E. Knuth. *Fundamental Algorithms*, volume 1 of *The Art of Computer Programming*. Addison-Wesley Longman Publishing Co., Inc., Reading, Mass., USA, third edition, 1997. ISBN 0-201-89683-4.

[97] H.W. Kuhn. The Hungarian method for the assignment problem. *Naval Research Logistics Quarterly*, 2(1-2):83–97, 1955. ISSN 1931-9193.

[98] S. Kullback. *Information Theory and Statistics*. Dover Publications, Inc., New York, 1968.

[99] A. Kupczok. Consequences of different null models on the tree shape bias of supertree methods. *Systematic Biology*, 60:218–225, 2011.

[100] G. N. Lance and W. T. Williams. A general theory of classificatory sorting strategies 1. hierarchical systems. *Computer Journal*, 9:373–380, 1966.

[101] F.J. Lapointe and C. Levasseur. Everything you always wanted to know about the average consensus and more. In O.R.P. Bininda-Emonds, editor, *Phylogenetic*

Supertrees: Combining Information to Reveal the Tree of Life, pages 87–105. Kluwer Academic Publishers, 2004.

[102] B. Larsen and C. Aone. Fast and effective text mining using linear-time document clustering. In *Proceedings of the Fifth ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*, KDD '99, pages 16–22, New York, NY, USA, 1999. ACM. ISBN 1-58113-143-7. doi: 10.1145/312129.312186.

[103] V.I. Levenshtein. Binary codes capable of correcting deletions, insertions, and reversals. *Soviet Physics Doklady*, 10:707–710, 1966.

[104] G. Liti, D. M. Carter, A. M. Moses, J. Warringer, L. Parts, S. A. James, R. P. Davey, I. N. Roberts, A. Burt, V. Koufopanou, I. J. Tsai, C. M. Bergman, D. Bensasson, M. J. O'Kelly, A. van Oudenaarden, D. B. Barton, E. Bailes, A. N. Nguyen, M. Jones, M. A. Quail, I. Goodhead, S. Sims, F. Smith, A. Blomberg, R. Durbin, and E. J. Louis. Population genomics of domestic and wild yeasts. *Nature*, 458(7236):337–41, 2009.

[105] J. MacQueen. Some methods for classification and analysis of multivariate observations. In *Proceedings of the Fifth Berkeley Symposium on Mathematical Statistics and Probability*, volume 1, pages 281–297. California, USA, 1967.

[106] M. Mahajan, P. Nimbhorkar, and K. Varadarajan. The planar k-means problem is NP-hard. In *Proceedings of the 3rd International Workshop on Algorithms and Computation*, WALCOM '09, pages 274–285, Berlin, Heidelberg, 2009. Springer-Verlag. ISBN 978-3-642-00201-4. doi: 10.1007/978-3-642-00202-1_24.

[107] P.C. Mahalanobis. On tests and measures of group divergence. *Journal of the Asiatic Society of Bengal*, pages 541–588, 1930.

[108] V. Makarenkov. Une nouvelle méthode efficace pour la reconstruction des arbres additifs à partir des matrices de distances incomplètes. *Proceedings of the 8-ièmes Rencontres de la Société Francophone de Classification*, pages 238–244, 2001.

[109] P. Mantovani, G. van der Linden, M. Maccaferri, M.C. Sanguineti, and R. Tuberosa. Nucleotide-binding site (NBS) profiling of genetic diversity in durum wheat. *Genome*, 49:1473––1480, 2006.

[110] R. Maronna and P.M. Jacovkis. Multivariate clustering procedures with variable metrics. *Biometrics*, 30(3):499–505, 1974. ISSN 0006341X.

[111] F.H.C. Marriott. Optimization methods of cluster analysis. *Biometrika*, 69(2): 417–421, 1982.

[112] M. Meilă. Comparing clusterings: An axiomatic view. In *In ICML '05: Proceedings of the 22nd International Conference on Machine Learning*, pages 577–584. ACM Press, 2005.

[113] M. Meilă. Comparing clusterings—an information based distance. *Journal of Multivariate Analysis*, 98(5):873–895, 2007. ISSN 0047-259X. doi: 10.1016/j.jmva.2006.11.013.

[114] M. Meilă and D. Heckerman. An experimental comparison of model-based clustering methods. *Machine Learning*, 42(1):9–29, 2001. ISSN 0885-6125.

[115] O. Merle, P. Hansen, B. Jaumard, and N. Mladenovic. An interior point algorithm for minimum sum-of-squares clustering. *SIAM Journal on Scientific Computing*, 21 (4):1485–1505, 1999.

[116] B.G. Mirkin. *Mathematical Classification and Clustering.* Nonconvex Optimization and its Applications. Kluwer Academic Pubishers, 1996. ISBN 978-0-792-34159-8.

[117] B.G. Mirkin and L.B. Chernyi. Measurement of the distance between distinct partitions of a finite set of objects. *Automation and Remote Control*, 31:786–792, 1970.

[118] B. Misof, B. Meyer, von B.M. Reumont, P. Kück, K. Misof, and K. Meusemann. Selecting informative subsets of sparse suprematrics increases the chance to find correct trees. *BMC Bioinformatics*, 14:348, 2013.

[119] F. Murtagh. A survey of recent advances in hierarchical clustering algorithms. *The Computer Journal*, 26(4):354–359, 1983.

[120] F. Murtagh. Complexities of hierarchic clustering algorithms: State of the art. *Computational Statistics Quarterly*, 1(2):101–113, 1984.

[121] F. Murtagh and P. Contreras. Methods of hierarchical clustering. *arXiv preprint arXiv:1105.0121*, 2011.

[122] R.T. Ng and J. Han. CLARANS: A method for clustering objects for spatial data mining. *IEEE Transactions on Knowledge and Data Engineering*, 14(5):1003–1016, 2002.

[123] R.D.M. Page. Modified mincut supertrees. *In Proceedings of Workshop on Algorithms in Bioinformatics (WABI '02), Springer Lecture Notes in Computer Science*, 2452:537–552, 2002.

[124] M.K. Pakhira. A modified k-means algorithm to avoid empty clusters. *International Journal of Recent Trends in Engineering*, 1(1), 2009.

[125] G. Palubeckis. A branch-and-bound approach using polyhedral results for a clustering problem. *INFORMS Journal on Computing*, 9(1):30–42, 1997.

[126] H. Philippe, E.A. Snell, E. Bapteste, P. Lopez, P.W.H. Holland, and D. Casane. Phylogenomics of eukaryotes: Impact of missing data on large alignments. *Molecular Biology and Evolution*, 21(9):1740–1752, 2004.

[127] M.J.D. Powell. Restart procedures for the conjugate gradient method. *Mathematical Programming*, 12(1):241–254, 1977.

[128] J.K. Pritchard, M. Stephens, and P.J. Donnelly. Inference of population structure using multilocus genotype data. *Genetics*, 155:945–959, 2000.

[129] A. Queiroz and J. Gatesy. The supermatrix approach to systematics. *Trends in Ecology and Evolution*, 22:34–41, 2006.

[130] W.M. Rand. Objective criteria for the evaluation of clustering methods. *Journal of the American Statistical Association*, 66(336):846–850, 1971. ISSN 01621459.

[131] S.J. Redmond and C. Heneghan. A method for initialising the k-means clustering algorithm using kd-trees. *Pattern Recognition Letters*, 28(8):965–973, 2007.

[132] J.C. Reeves, E. Chiapparino, P. Donini, M. Ganal, J. Guiard, S. Hamrit, M. Heckenberger, X.-Q. Huang, M. van Kaauwen, E. Kochieva, R. Koebner, J. R. Law, V. Lea, V. Le Clerc, T. Van der Lee, F. Leigh, G. Van der Linden, L. Malysheva, A. E. Melchinger, S. Orford, J. C. Reif, M. Röder, A. Schulman, B. Vosman, C. Van der Wiel, M. Wolf, and D. Zhang. Changes over time in the genetic diversity of four major European crops: A report from the Gediflux framework 5 project. *Proceedings of the XVIIth EUCARPIA General Congress, Tulln, Austria, 8–11 September 2004*, pages 3–7, 2004.

[133] J.C. Reif, A.E. Melchinger, and M. Frisch. Genetical and mathematical properties of similarity coefficients applied in plant breeding and seed bank management. *Crop Science*, 45:1–7, 2005.

[134] A.P. Reynolds, G. Richards, B. De La Iglesia, and V.J. Rayward-Smith. Clustering rules: A comparison of partitioning and hierarchical clustering algorithms. *Journal of Mathematical Modelling and Algorithms*, 5(4):475–504, 2006. ISSN 1570-1166.

[135] D.F. Robinson and L.R. Foulds. Comparison of phylogenetic trees. *Mathematical Biosciences*, 53(1):131–147, 1981.

[136] B. Roure, D. Baurain, and H. Philippe. Impact of missing data on phylogenies inferred from empirical phylogenomic data sets. *Molecular Biology and Evolution*, 30(1):197–214, 2012.

[137] J. Rubin. Optimal classification into groups: An approach for solving the taxonomy problem. *Journal of Theoretical Biology*, 15(1):103–144, 1967. ISSN 0022-5193. doi: 10.1016/0022-5193(67)90046-X.

[138] S. Sahni and T. Gonzalez. P-complete approximation problems. *Journal of the ACM (JACM)*, 23(3):555–565, 1976.

[139] N. Saitou and M. Nei. The neighbor-joining method: A new method for reconstructing phylogenetic trees. *Molecular Biology and Evolution*, 4(4):406–425, 1987.

[140] M. Sanderson, M. McMahon, and M. Steel. Phylogenomics with incomplete taxon coverage: The limits to inference. *BMC Evolutionary Biology*, 10(1):155, 2010.

[141] S.M. Savaresi, D. Boley, S. Bittanti, and G. Gazzaniga. Cluster selection in divisive clustering algorithms. In *SDM*. SIAM, 2002.

[142] M. Sayar-Turet, S. Dreisigacker, H.-J. Braun, A. Hede, R. MacCormack, and L.A. Boyd. Genetic variation within and between winter wheat genotypes from Turkey, Kazakhstan and Europe as determined by NBS-profiling. *Genome*, 54:419–430, 2011.

[143] M. Schader and W. Gaul. The MVL (missing values linkage) approach for hierarchical classification when data are incomplete. In *Analyzing and Modeling Data and Knowledge*, pages 107–115. Springer, 1992.

[144] J. H. Schwartz and B. Maresca. Do molecular clocks run at all? A critique of molecular systematics. *Biological Theory*, 1:357–371, 2006.

[145] C. Scornavacca, V. Berry, V. Lefort, E.J.P Douzery, and V. Ranwez. PhySICIST: Cleaning source trees to infer more informative supertrees. *BMC Bioinformatics*, 9: 413, 2008.

[146] C. Semple and M. Steel. A supertree method for rooted trees. *Discrete Applied Mathematics*, 105(1):147–158, 2000.

[147] C. Semple and M. Steel. *Phylogenetics*. Oxford Lecture Series in Mathematics and its Applications. Oxford University Press, 2003. ISBN 978-0-198-50942-4.

[148] R. Sibson. SLINK: An optimally efficient algorithm for the single-link cluster method. *The Computer Journal*, 16(1):30–34, 1973.

[149] R.R. Sokal. A statistical method for evaluating systematic relationships. *University of Kansas Science Bulletin*, 38:1409–1438, 1958.

[150] H. Späth. *Cluster Analysis Algorithms*. Ellis Horwood Limited, Chichester, West Sussex, England, 1980. ISBN 0-85312-141-9.

[151] R.P. Stanley. *Enumerative Combinatorics, Volume I*. Number 49 in Cambridge Studies in Advanced Mathematics. Cambridge University Press, second edition, 2000. ISBN 978-0-521-66351-9.

[152] M. Steel. The complexity of reconstructing trees from qualitative characters and subtrees. *Journal of Classification*, 9(1):91–116, 1992.

[153] M. Steel and M.J Sanderson. Characterizing phylogenetically decisive taxon coverage. *Applied Mathematics Letters*, 23(1):82–86, 2010.

[154] M. Steinbach, G. Karypis, and V. Kumar. A comparison of document clustering techniques. Technical Report 00-034, Department of Computer Science and Engineering, University of Minnesota, Minneapolis, Minnesota, USA, 2000.

[155] В.И. Левенштейн. Двоичные коды с исправлением выпадений, вставок и замещений символов. *Доклады Академий Наук СССР*, 163:845–848, 1965. English language version: [103].

[156] M. Telgarsky and A. Vattani. Hartigan's method: k-means clustering without Voronoi. In *13th International Conference on Artificial Intelligence and Statistics (AISTATS)*, 2010.

[157] S. van Dongen. Performance criteria for graph clustering and Markov cluster experiments. Technical report, CWI (Centre for Mathematics and Computer Science), Amsterdam, The Netherlands, The Netherlands, 2000.

[158] A. Vattani. k-means requires exponentially many iterations even in the plane. In *Proceedings of the 25th Annual Symposium on Computational Geometry*, pages 324–332. ACM, 2009.

[159] C.S. Wallace and D.M. Boulton. An information measure for classification. *The Computer Journal*, 11(2):185–194, 1968.

[160] D.L. Wallace. A method for comparing two hierarchical clusterings: Comment. *Journal of the American Statistical Association*, 78(383):569–576, 1983. ISSN 01621459.

[161] J.T. Weir and D. Schluter. Calibrating the avian molecular clock. *Molecular Ecology*, 17:2321–2328, 2008.

[162] J.T. Weir and D. Schluter. Ice sheets promote species in boreal birds. *Proceedings of the Royal Society of London Biological Sciences*, 271:1881–1997, 2008.

[163] C. West, S.A. James, R.P. Davey, J. Dicks, and I.N. Roberts. Ribosomal DNA sequence heterogeneity reflects intraspecies phylogenies and predicts genome structure in two contrasting yeast species. *Systematic Biology*, 2014. doi: 10.1093/sysbio/syu019.

[164] S.S. Wilks. Multidimensional statistical scatter. In I. Olkin, S.G. Ghurye, W. Hoeffding, W.G. Madow, and H.B. Mann, editors, *Contributions to Probability and Statistics*, chapter 40. Stanford University Press, Stanford, California, USA, 1960.

[165] S.J. Willson. Constructing rooted supertrees using distances. *Bulletin of Mathematical Biology*, 66:1755–1783, 2004.

[166] A.I. Wirth. *Approximation Algorithms for Clustering*. PhD thesis, Princeton University, January 2005.

[167] Y. Xiao, W. Liu, Y. Dai, C. Fu, and Y. Bian. Using SSR markers to evaluate the genetic diversity of *Lentinula edodes*' natural germplasm in China. *World Journal of Microbiology and Biotechnology*, 26:527–536, 2010.

[168] M. Yedla, S.R. Pathakota, and T.M. Srinivasa. Enhancing k-means clustering algorithm with improved initial center. *International Journal of Computer Science and Information Technologies*, 1(2):121–125, 2010.

[169] P.N. Yianilos. Normalized forms for two common metrics. Technical report, NEC Research Institute, 2002.

[170] F. Yuan, Z. Meng, H. Zhang, and C. Dong. A new algorithm to get the initial centroids. In *Proceedings of International Conference on Machine Learning and Cybernetics*, volume 2, pages 1191–1193, 2004. doi: 10.1109/ICMLC.2004.1382371.

[171] L.A. Zadeh. Fuzzy sets. *Information and Control*, 8(3):338–353, 1965.