Novel Algorithms and Methodology to Help Unravel Secrets that Next Generation Sequencing Data Can Tell

Andrei-Alin Popescu

School of Computer Science

University of East Anglia

A thesis submitted for the degree of

Doctor of Philosophy

September 2015

I would like to dedicate this thesis to my parents and close friends who have supported me throughout this thesis.

Acknowledgements

I would like to thank my primary supervisor Dr. Katharina T. Huber, University of East Anglia, for her support and patience throughout my degree. I would also like to thank Dr Andrea Harper, Dr Martin Trick and Prof Ian Bancroft for their supervision, work and collaboration, and the Norwich Research Park for their financial support for the duration of my degree.

Declaration

No portion of the work referred to in this thesis has been submitted in support of an application for another degree or qualification at this or any other university or other institute of learning.

Statement of Originality

I certify that this thesis, and the research to which it refers, are the product of my own work, and that any ideas or quotations from the work of other people, published or otherwise, are fully acknowledged.

List of Publications

The first two publications were the result of attempting to guide STRUCTURE by providing estimates for K using distance-based phylogenetics methods. The latter two publications were the result of attempts to provide fast alternatives to existing population structure inference methods.

- K.T. Huber. and A-A. Popescu Lassoing and corralling rooted phylogenetic trees. *Bull Math Biol.*, 75:444–465, 2013.[1]

- A-A. Popescu, K.T. Huber., E. Paradis ape 3.0: New tools for distance-based phylogenetics and evolutionary analysis in R. *Bioinformatics*, 28:1536–1537, 2012.

- A-A. Popescu , A.L Harper, M. Trick, I. Bancroft and K.T. Huber. A novel and fast approach for population structure inference using kernel-PCA and optimization. *Genetics*, 198(4):1421–31, 2014. doi: 10.1534/genetics.114.171314

- A-A. Popescu, K.T. Huber. PSIKO2: a fast and versatile tool to infer population stratification on various levels in GWAS *Bioinformatics*, 31: Epub, 2015.
[2]

[1] Author order alphabetical.

[2] The respective journals of these publications have granted the author permission to include them in this thesis.

Abstract

The genome of an organism is its complete set of DNA nucleotides, spanning all of its genes and also of its non-coding regions. It contains most of the information necessary to build and maintain an organism. It is therefore no surprise that sequencing the genome provides an invaluable tool for the scientific study of an organism. Via the inference of an evolutionary (phylogenetic) tree, DNA sequences can be used to reconstruct the evolutionary history of a set of species. DNA sequences, or genotype data, has also proven useful for predicting an organisms' phenotype (i. e. observed traits) from its genotype. This is the objective of association studies.

While methods for finding the DNA sequence of an organism have existed for decades, the recent advent of Next Generation Sequencing (NGS) has meant that the availability of such data has increased to such an extent that the computational challenges that now form an integral part of biological studies can no longer be ignored. By focusing on phylogenetics and Genome-Wide Association Studies (GWAS), this thesis aims to help address some of these challenges. As a consequence this thesis is in two parts with the first one centring on phylogenetics and the second one on GWAS.

In the first part, we present theoretical insights for reconstructing phylogenetic trees from incomplete distances. This problem is important in the context of NGS data as incomplete pairwise distances between organisms occur frequently with such input and ignoring taxa for which information is missing can introduce undesirable bias. In the second part we focus on the problem of inferring population stratification between individuals in a dataset due to reproductive isolation. While powerful methods for doing this have been proposed in the literature, they tend to struggle when faced

with the sheer volume of data that comes with NGS. To help address this problem we introduce the novel PSIKO software and show that it scales very well when dealing with large NGS datasets.

Contents

List of Figures

Glossary

Definitions

admixed dataset A biological dataset whose individuals have genotype information coming from a number of reproductively isolated ancestral populations.

association studies: Genetic studies attempting to find links between genotype and observed traits.

cherry: For a phylogenetic tree T, a cherry represents two leaves with a common parent vertex.

competing tree: For a set of cords $\mathcal{L}$, we say that T' is a competing tree of T with respect to $\mathcal{L}$ if for any $\{x,y\} \in \mathcal{L}$ both T and T' induce the same distance between x and y.

to corral: A set of cords $\mathcal{L}$ corrals T if it is a weak lasso of T.

dendrogram: Rooted phylogenetic tree such that all the path lengths from the root to the leaves are equal.

diversity panel Biological dataset representing genotype sampled from a diverse population of individuals from a species. They are usually produced in order to allow the inference of a link between genotype and phenotype.

edge-weight lasso: We call a lasso an edge-weight lasso if it uniquely determines the edge-weights of a fixed topology phylogenetic tree.

founder populations Ancestral populations of a sample which are assumed to have been in reproductive isolation.

genetic marker measured difference at the sequence level between organisms under investigation.

Genome-Wide Association Studies: Association studies which consider whole organism genomes.

Local Ancestry Inference: Inferring the ancestral population that gave rise to a genomic region of interest of an admixed individual.

weak lasso: A weak lasso determines a topology and refinements of that topology.

lasso: We call a set of cords $\mathcal{L}$ a lasso if it contains enough information to capture certain properties of a phylogenetic tree.

linkage equilibrium: Two markers are said to be in linkage equilibrium if there is no tendency for them to be found together in individuals.

$\mathcal{L}$-isometric: For a set of cords $\mathcal{L}$, we say that T' and T are $\mathcal{L}$-isometric if for any $\{x,y\} \in \mathcal{L}$ both T and T' induce the same distance between x and y.

Marker Assisted Selection using genetic marker information to attempt to predict the phenotype of an organism to better select for desirable traits.

Next Generation Sequencing: An umbrella term covering recent high-throughput genotyping technologies.

non-degenerate tree: A rooted phylogenetic tree on X is called non-degenerate if it contains at least one interior vertex other than the root.

path length: The length of a path in an edge-weighted graph is the sum of all weights of edges present in that path.

population structure reproductive isolation amongst individuals of a sample.

pseudo-Cherry: For a given phylogenetic tree T on X, we call a subset of leaves of T which have the same parent a pseudo-cherry.

Q-**matrix**: For an admixed dataset comprising n individuals arising from K ancestral populations, the Q-matrix is an $n \times K$ matrix indicating for each of the n individuals, the proportion of their genotype that came from each of the K ancestral populations.

Quantitative Trait Locus: Locus linked to a phenotypic trait of interest.

sequence alignment: A sequence alignment is a matrix whose rows represent sequenced individuals and whose columns represent positions in the alignment. They are constructed from sequence data to attempt to reveal similarities among sequences.

Single Nucleotide Polymorphism: single base-pair difference between sequenced samples at a given locus.

set of cords: For a set of taxa X and a distance matrix D, a set of cords represents elements $\{x, y\} \in \binom{X}{2}$ such that distance between x and y is present in D.

star tree: A star tree is a phylogenetic tree with a unique interior vertex.

strong lasso: A strong lasso is both an edge-weight and a topological lasso.

topological lasso: We call a lasso a topological lasso if it uniquely determines the topology of a tree.

triplet: A rooted phylogenetic tree on three leaves is called a triplet.

Chapter 1

Introduction

Arguably, one of the most ambitious scientific projects of our time is the 1000 genome project [14], which aims to capture the genetic variability of humans by obtaining genetic information from a large number of individuals from all geographic regions of the world. The potential applications of such information are boundless, ranging from shedding light into the historic origins of modern human populations, to studying the susceptibility of individuals to disease and thus personalised medicine. Despite its obvious practical utility, the 1000 genome project would not have been possible without the advent of Next Generation Sequencing (NGS), which made such a wide-scale study of genetic diversity feasible. For this reason, NGS data holds great promise in scientific development for the foreseeable future.

To better understand the power (and challenges) of NGS, we first elaborate on sequencing techniques used before NGS technologies became popular. The genome of an organism is its complete set of DNA nucleotides, spanning all of its genes and inclusive of its non-coding regions. It contains all the information necessary to build and maintain an organism. It is therefore no surprise that access to its genome via nucleotide sequencing provides an invaluable tool for the scientific study of an organism. Examples of such studies include elucidating the evolutionary past of a set of species (via e. g. the inference of a phylogenetic tree [6]), or predicting an organisms phenotype (i. e. observed traits) from its genotype which is the objective of association studies [124].

Attempting to obtain the DNA sequence of an organism of interest has historically received a lot of interest. One of the earliest sequencing technologies to obtain wide-

spread use is *Sanger sequencing* [107]. Although it has led to interesting early results, one of its major disadvantages is its high cost and relatively low throughput. Alternative genotyping technologies include *Restriction Fragment Length Polymorphism (RFLP)* [105], *Random Amplified Polymorphic DNA (RAPD)*, *Simple Sequence Repeats* (SSR, or microsatellites) [2], *Amplified Fragment Length Polymorphism* (AFLP) [118]. (see [124] and [13] for an overview of the advantages and disadvantages of each of these technologies). Although different in detail on how they reveal genetic differences in the organisms under consideration, they all represent a cheaper alternative to Sanger sequencing and also tend to produce more data. It is therefore no surprise that a number of methods have been proposed to analyse data generated by them (e. g. [98, 41, 43] to name just a few).

While ideally all present genetic differences would be useful and desirable to have, identifying them all tends to be very difficult to realise in practice. To overcome this, the aforementioned technologies output genetic information in the form of so called (genetic) markers, which are a form of observable variation of the genome of the sample under investigation. They have been traditionally used as stand-ins for large unsequenced portions of the whole genome. The number of found markers tends to be much lower than the length of a genome, resulting in potentially patchy genome coverage. Nevertheless, they proved very useful in early genetic studies. Having said that, studies such as the 1000 genome project would have been infeasible with them. The technology that made such large-scale projects a reality is NGS, which is an umbrella term covering recent high-throughput genotyping technologies [53]. The principal marker generated by such technologies is the Single Nucleotide Polymorphism (SNP) [76], and the key characteristic of all technologies covered by this term is the abundance of marker they produce.

Recognizing the potential of NGS for many biological problems has meant that NGS technologies are quickly becoming the norm in genotyping. They have the advantage of providing vast amounts of data at a very low cost. In turn, this has allowed for unprecedented detail in genomic studies [15] and has made whole genome sequencing a reality [117, 14]. NGS is not restricted to DNA sequences [47] and also has applications in medicine [11]. While the advantages of NGS are indisputable, the abundance of data produced by it has introduced hard computational challenges, ranging from basic tasks such as data storage [67] to more advanced topics such as

reconstructing evolutionary scenarios from this data they produce. Furthermore popular tools for analysing genetic marker data (e. g. STRUCTURE, see Chapter 2, Section 2.5.3) tend to struggle when applied to NGS data, some taking weeks to run even on powerful computing clusters for even modest-sized NGS datasets [51]. Similarly, sequence-based phylogenetic methods quickly become impractical on datasets of this scale. In this thesis, we focus on some of the challenges faced by phylogenetics and association studies when faced with NGS data.

Model-based phylogenetic approaches [27, 58, 37] have proven to be very powerful approaches to reconstructing the evolutionary history of extant species. However, with the explosion of dataset size brought about by NGS, such methods tend to suffer long runtimes due to the explosion of the number of trees which must be searched [60] as the space of such trees is dependent on the number of taxa (e. g. organisms or species) to be analysed. Distance-based methods which work by inferring phylogenies from pairwise distances (measures of dissimilarity) between extant species can potentially provide a solution for this. Although they have been criticised for reducing the complexity of genotype data to a single number (viewed as a measure of how different two extant species are), they have proven to work reasonably well in practice [41]. Additionally, due to their relatively low runtime complexity (compared to model-based methods such as e. g. likelihood or Bayesian methods), they are suitable for working with NGS data. Output from them can be used in phylogenetics studies as-is, or can be used as a starting point to speed up more sophisticated methods [27, 58].

While distance-based methods are very well studied [106, 17, 41], NGS data introduces new challenges for them, in the form of missing data, i. e. pairs of extant species for which distances are not known. There are several factors that could lead to this, such as incomplete sample selection or the failure of an experimental assay. Ignoring missing distances can introduce undesirable bias [55]. Novel distance-based approaches that can cope with missing data might therefore provide an avenue for carrying out phylogenetic inference with the potential of avoiding the need to repeat potentially costly experiments in the event of failure. While significant work has been carried out to attempt to reconstruct tree-like evolutionary scenarios (also called phylogenetic trees) from incomplete distances [17, 73], many important questions still remain, including the following one. Given that there is considerable redundancy in a distance matrix, for which pairs of taxa are the distances required so that the available

incomplete distances uniquely determine a phylogenetic tree? This question is relevant because it could provide researchers with an answer to the question of whether the collected data (with potential missing values) is sufficient for a phylogenetic analysis, or if improvements in accuracy should be obtained by collecting more data or repeating failed experiments to attempt to fill gaps in the data. This problem, however, has been found to be NP-hard in general [33], but recent work has provided answers to this question under restricted circumstances [56]. In Chapter 3, we present theoretical work that addresses the question of when an incomplete distance matrix uniquely determines a special kind of phylogenetic tree, known as an ultrametric tree [57].

A further area where NGS data has proven highly useful is association studies. Here, one attempts to find links between an individual's genotype and phenotype. Put differently, the aim is to infer which regions of a genome encode traits of interest. Association studies can be done by generating purpose-built populations with desirable properties. They are known as *mapping populations*, and for them, finding genotype-phenotype links is straight-forward [13] using linkage analysis. While the computational tools for mapping populations are well understood and can be applied even to NGS data, they have several drawbacks. Firstly, obtaining a mapping population can be quite a laborious and costly endeavour [87, 24]. Secondly, since such populations are synthetically produced, they may not contain sufficient genotypic diversity to resolve causative loci (e. g. SNPs) at a satisfactory level [94].

To address these limitations of linkage analysis [124], an alternative method known as association mapping or linkage disequilibrium mapping has been proposed [103]. Instead of using a purpose-built population, such methods use samples from naturally occurring populations. Not only does this eliminate the need for (potentially) difficult to obtain mapping populations, but it also exploits the naturally occurring genetic diversity of samples to provide increased mapping resolution and number of genetic markers. In the form of potentially spurious links between genotype and phenotype, association mapping also poses computational and statistical challenges. These are mainly attributable to unaccounted-for relatedness between natural samples (accessions) which can occur due to population structure or familial relatedness [124, 98].

A method that has proven very successful in accounting for such confounding factors in association mapping are *Mixed Linear Models* (MLM) [122]. They work by attempting to fit a linear model to a dataset where the predictor variables are the genomic

markers and the predicted value is the phenotype data. MLMs have been extended to also take into account population stratification and relatedness between individuals, and have been shown to reduce false-positive association between genotype and phenotype [122]. Recent work has allowed MLMs to also be able to efficiently deal with NGS data [123, 70, 71]. For MLM methods to work effectively however, population structure and familial relatedness must be computed, and while the latter can be done efficiently [3], the former still poses a significant challenge for NGS data. Due to this, the second part of the thesis is dedicated to this problem. In Chapter 5, we present the novel PSIKO software tool. This includes details concerning its implementation as well as a demonstration of its effectiveness for even large NGS datasets. As detailed there, PSIKO provides a significant boost in speed from already existing approaches, while also producing results that are strikingly similar to those approaches.

Reflecting the two areas of research considered, the organisation of the thesis is structured into two distinct but interrelated parts. Part I is dedicated to phylogenetic analysis applied to NGS data, and Part II to association studies applied to such data. Preceding them is a literature review for the areas. We next present details on the various remaining chapters of this thesis.

Chapter 2 In this chapter we introduce background concepts relating to graph theory and phylogenetic trees. We then review some of the most popular approaches for phylogenetic tree reconstruction, focusing mainly on distance based methods. Subsequent to this, we review population structure inference from sequence data. We formalise the problem from a computer science perspective, and then review popular algorithms which aim to tackle it.

Part I: Phylogenetic methods for dealing with NGS data

Owing to their importance in the study of NGS data, we dedicate the first part of the thesis to the study of phylogenetic inference from distances. Due to the potential of NGS data for incomplete pair-wise distance information we focus on work to help address this problem. We provide details of both novel theoretical work and efficient implementation of existing algorithms for tree reconstruction from incomplete distances in popular open-source phylogenetics packages.

Chapter 3 This chapter is based on [57]. We present novel theoretical results per-

taining to reconstructing a special kind of phylogenetic tree (ultrametric tree) from incomplete distances. We mainly discuss when such a tree can be uniquely reconstructed (lassoed) by an incomplete distance matrix.

Chapter 4 This chapter is based on [91] and concerns the implementation of several existing phylogenetic reconstruction methods that take as input an incomplete distance matrix. We present technical details pertaining to their implementation and also benchmark them against existing implementations of said methods to show that, in most cases, our implementation of a method in question is more efficient.

II: Population Structure inference from NGS data

As is well documented [51], population structure can lead to spurious genotype-phenotype associations. Algorithms to correct for this tend to have a long runtime, making them impractical for NGS data.

Chapter 5 This chapter is based on [92], and concerns the introduction of the novel PSIKO algorithm, which aims to efficiently deal with (global) ancestry inference for NGS datasets. We study PSIKO's ability to make inferences about population structure, and compare it against existing state-of-the-art methods, showing its very fast runtime.

Chapter 6 This chapter is based on [93] and concerns the introduction of the PSIKO2 algorithm, an extension of PSIKO to also allow for localised ancestry inference of genomic regions. Being able to do this is especially important when studying recently admixed individuals, as it provides population history resolution that would be hard to obtain via global ancestry inference - see Chapter 5. We assess PSIKO2's ability to infer local ancestry by comparing it to state of the art approaches which aim to also solve this problem.

Chapter 7 In this chapter, we present concluding remarks with regards to the research presented in this thesis. We review our major results and also present avenues of potential future work that could be of interest.

Chapter 2

Background

2.1 Chapter Summary

In this chapter we provide background on the relevant areas considered in this thesis. The first part is concerned with reviewing relevant concepts and algorithms in phylogenetic analysis. The second part reviews some of the major concepts and popular algorithms in association studies.

2.2 Introductory concepts in graph theory

In this section, we introduce some of the basic graph-theoretical concepts used below. We follow [56]. We start with the concept of a *graph*. A graph G, is an ordered pair, consisting of a set $V(G)$ of vertices and a set $E(G) \subseteq \binom{V(G)}{2}$ of edges. If $e = \{a,b\}$ is an edge of G, then we say that a and b are *adjacent* and that e is *incident* on a and b. The *degree* of a vertex of G is the number of edges in G incident with it. A *path* in G is a sequence of pairwise distinct vertices $v_1, v_2, ..., v_n \in V(G), n \geq 1$, such that $\{v_i, v_{(i+1)}\} \in E(G)$, $1 \leq i < n-1$. A *cycle* in G is a path $P : v_1, v_2, ..., v_n$ where, in addition, we have an edge between v_1 and v_n. A graph is said to be *connected* if there is a path between all pairs of two of its vertices. A *subgraph* G' of G is a graph such that $V(G') \subseteq V(G)$ and $E(G') \subseteq E(G)$, and if $\{a,b\} \in E(G')$ then $a,b \in V(G')$. If G' is a subgraph of G then we say that G' *spans* G if $V(G) = V(G')$. We say that G' is an *induced subgraph* of G if for any two vertices $x,y \in V(G')$, we have that

$\{x,y\} \in E(G')$ precisely if $\{x,y\} \in E(G)$. A supergraph of G is a graph of which G is a subgraph. A *connected component* of a graph G is a maximal induced subgraph H of G that is connected, that is, no supergraph of H that is also an induced subgraph of G is connected. If we associate to a graph G a function $w : E(G) \longrightarrow \mathbb{R}_{\geq 0}$ then we say that (G,ω) is a *weighted graph* and call $\omega(e)$ the *weight* of the edge $e \in E(G)$. Moreover, we call ω the *edge-weight function* of G. Furthermore if to each edge in a graph G we associate a direction, then we say that G is a *directed* graph. We call the directed edges in a directed graph G *arcs*, and denote an arc starting at some $a \in V(G)$ and ending at some $b \in V(G)$ by (a,b). The set of arcs of a directed graph G is denoted by $A(G)$. A *directed path* in a directed graph G is a sequence of pairwise distinct vertices $v_1, v_2, ..., v_n \in V(G)$ such that $(v_i, v_{i+1}) \in A(G)$.

2.3 (Unrooted) phylogenetic trees

In this section, we review the concept of a phylogenetic tree on a finite, non-empty set X. Such trees have been introduced in the literature within an unrooted and rooted context and are generally used to model the evolution of a set X of extant taxa. We first provide a formal definition and then show how such trees can be characterised in terms of certain combinatorial objects that arise from e. g. sequence data. We will mainly focus on tree reconstruction from distances (i.e. measures of dissimilarity) between elements of X, reviewing some of the most popular methods that have been introduced for this for both complete and incomplete distance sets. We start by introducing further terminology concerning graphs. Assume from now on that X is always a finite non-empty set.

A *tree* is a connected graph that contains no cycles (see e. g. Figure 2.1). An *unrooted phylogenetic tree T on X* is a tree with leafset X which does not have degree-2 vertices. If the set X is of no relevance to the discussion, then we refer to a phylogenetic tree on X as just a *phylogenetic tree*. Then a *leaf* of T is a vertex of degree 1. For example a is a leaf of T in Figure 2.1. Any other vertex of T is said to be an *interior vertex*. A phylogenetic tree is called *binary* if all interior vertices are of degree 3. Suppose T is a phylogenetic tree. The *set of all leaves* of T (the *leafset*) is denoted by $L(T)$. We call an edge incident with a leaf x the *pendant* edge of x and denote it by p_x. In Figure 2.1, the edge $\{v,a\}$ is a pendant of a.

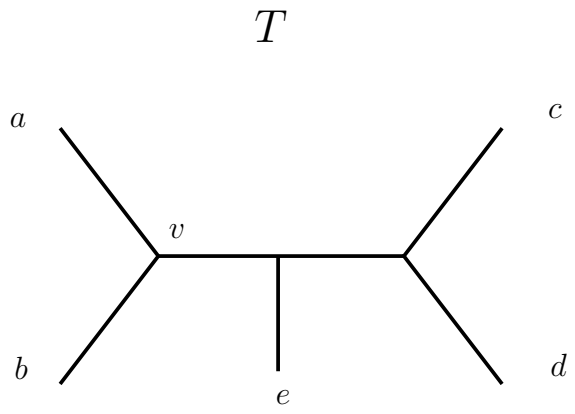


Figure 2.1: An example phylogenetic tree on leaves $X = \{a, b, c, d, e\}$.

We denote by (T, ω) the edge-weighted phylogenetic tree T on X with associated edge-weight function $\omega : E(T) \longrightarrow \mathbb{R}_{\geq 0}$. For (T, ω) an edge-weighted phylogenetic tree, we denote the set of edges on the path between two vertices u, v of T by $E_{(T,\omega)}(u, v)$. The *phylogenetic distance* $d_{(T,\omega)}(a, b)$ (as defined in [38]) between two leaves a, b of T is simply

$$d_{(T,\omega)}(a,b) = \sum_{e \in E_{(T,\omega)}(a,b)} \omega(e)$$

We say that two phylogenetic trees $T_1 = (V_1, E_1)$ and $T_2 = (V_2, E_2)$ on X are *equivalent* if there is a bijective map $\varphi : V_1 \longrightarrow V_2$, such that $\{x, y\} \in E_1$ if and only if $\{\varphi(x), \varphi(y)\} \in E_2$, and $\varphi(x) = x$ holds for all $x \in X$. See Figure 2.2 for an illustrating example.

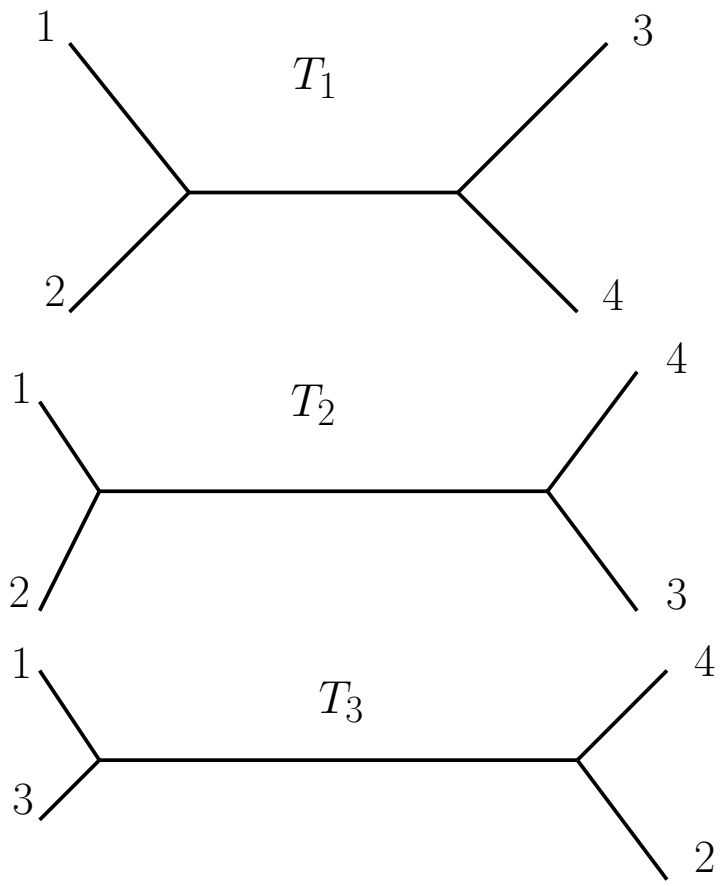


Figure 2.2: Example of three phylogenetic trees T_1, T_2 and T_3 on $X = \{1,2,3,4\}$. T_1 is equivalent to T_2. T_3 is not equivalent to either T_1 or T_2.

2.3.1 Rooted Phylogenetic Trees

A *rooted phylogenetic tree* T on X is a rooted tree with leafset X and with a specially chosen non-leaf vertex ρ of T called the *root* of T. Note that the root may be of degree two. For convenience, we will assume that all rooted phylogenetic trees are always directed away from the root. Unless stated otherwise, all phylogenetic trees considered in Section 2.3 are assumed to be unrooted. It should be noted that all definitions for unrooted phylogenetic trees carry over to the rooted framework in a canonical way (see e.g. [110] for details). Furthermore, suppose T is a rooted phylogenetic tree. Then we say that a vertex v of T is an *ancestor* of a vertex u, denoted by $v \preceq_G u$ if and only if

there is a directed path between v and u. See for example Figure 2.3.

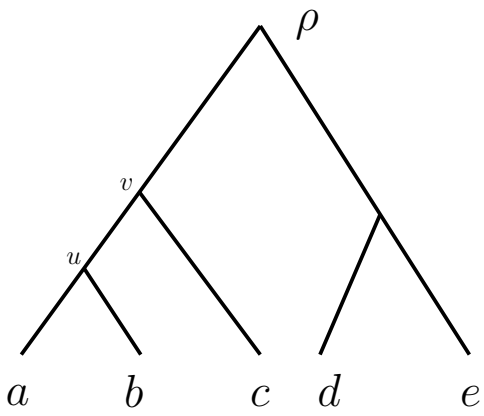


Figure 2.3: An example rooted phylogenetic tree on leaves $X = \{a,b,c,d,e\}$. The vertex v is an ancestor of u.

2.4 Characterising Phylogenetic Trees

We next look into how phylogenetic trees can be characterised by simple combinatorial objects. Such characterisations are useful as these alternative objects are generally more easily obtained from biological data than phylogenetic trees themselves, and thus might lend themselves to approaches for reconstructing phylogenetic trees from them. We first look at objects describing unrooted phylogenetic trees, called splits and also at how to build phylogenetic trees from sequences. We conclude with looking at how one might obtain rooted and unrooted edge-weighted phylogenetic trees from distances.

2.4.1 Characterisation of unrooted phylogenetic trees in terms of splits

In this section, we review how certain combinatorial objects known as *splits* can be used to reconstruct phylogenetic trees. We begin with some definitions and then review what conditions splits need to satisfy in order to give rise (in a well defined sense) to a (unique up to equivalence) phylogenetic tree. We finish by presenting an algorithm which can be used to construct phylogenetic trees from splits.

A *split* $S = \{A, B\}$ of X is a bipartition of X into two non-empty, disjoint subsets $A, B \subseteq X$ such that $A \cup B = X$. It is usual to denote S by $A|B$, or, equivalently $B|A$. A split is said to *separate* two distinct elements x, y of X if either $x \in A$ and $y \in B$ or $y \in A$ and $x \in B$. A split $A|B$ is called *trivial* if $|A| = 1$ or $|B| = 1$. A set of splits Σ on X is called a *split system* on X. Note that if $A = \{a_1, a_2, ..., a_k\}$ and $B = \{b_1, b_2, ..., b_l\}$ for $k, l \geq 1$ then we will write $a_1a_2...a_k|b_1b_2...b_l$ rather than $\{a_1a_2...a_k\}|\{b_1b_2...b_l\}$. A *weighted split system* is a tuple (Σ, ω), where $\omega : \Sigma \longrightarrow \mathbb{R}_{\geq 0}$ is a weight function on Σ.

We next explore the relationship between phylogenetic trees on X and split systems on X. Suppose T is a phylogenetic tree on a set X. Then T *induces* a split system $\Sigma(T)$ on X in the following way. Suppose e is an edge of T. Then removing e from T results in the split $S_e = A|B$, where A is the set of all leaves present in one of the resulting two connected components of T, and B is the set of all leaves of T in the other. One by one, we continue in the same fashion for all edges in T, and put $\Sigma(T) = \{S_e | e \in E(T)\}$. Then we say T *displays* a split S on X if $S \in \Sigma(T)$. More generally, we say that a split system Σ on X is displayed by T if $\Sigma \subseteq \Sigma(T)$. Note that a *weighted split system*, $(\Sigma(T), \omega_{\Sigma(T)})$, $\omega_{\Sigma(T)} : \Sigma(T) \longrightarrow \mathbb{R}_{\geq 0}$, can be obtained from an edge-weighted phylogenetic tree (T, ω) by setting $\omega_{\Sigma(T)}(S_e) = \omega(e)$, for all $e \in E(T)$. See Figure 2.4 for an example.

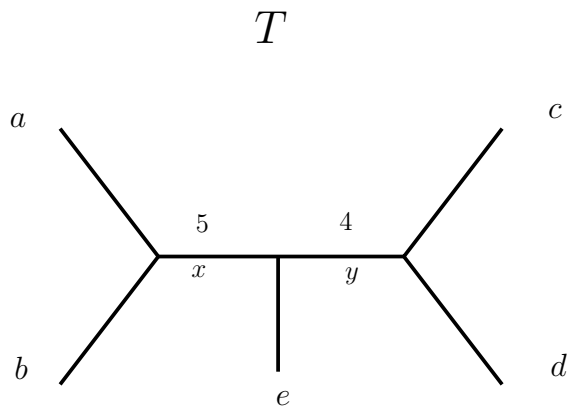


Figure 2.4: A phylogenetic tree with two weighted edges x and y. We have that $S_x = ab|ecd$ and $\omega_{\Sigma(T)}(S_x) = 5$. Similarly $S_y = cd|abe$ and $\omega_{\Sigma(T)}(S_y) = 4$.

Motivated by this observation, we are interested in the following two questions for

the remainder of this section:

(i) Which conditions does a split system Σ need to satisfy such that a phylogenetic tree can be reconstructed from it, which displays Σ as described above.

(ii) If a phylogenetic tree T can be constructed from a split system Σ when is that tree the only tree (up to equivalence) that displays Σ.

Our review will be based on [48]. One of the most important results in phylogenetics [8] (which addresses the above questions) states that a split system Σ on X is *compatible* if and only if for all pairs of distinct splits $S_1 = A_1|B_1$, $S_2 = A_2|B_2$ in Σ, one of the following intersections is empty:

$$A_1 \cap A_2, A_1 \cap B_2, B_1 \cap A_2, B_1 \cap B_2 \tag{2.1}$$

Given that any two splits displayed by a tree satisfy Condition 2.1, it follows that checking if a split split system Σ can be represented by a phylogenetic tree T such that $\Sigma = \Sigma_T$ can be done in $O(|\Sigma|^2)$ time.

While this result simultaneously addresses the above two questions, it is not immediately clear how to reconstruct the phylogenetic tree defined by a compatible split system Σ. To answer this question, we now turn our attention to the special case that Σ is compatible. Given a compatible set Σ of splits on X, there exists a reconstruction method which returns the phylogenetic tree on X which displays precisely Σ. It is called *tree popping* and was first described in [79]. It works by taking the splits in Σ in an arbitrary order and constructs a sequence of phylogenetic trees $T_1, T_2, \ldots, T_k$, $k = |\Sigma|$, such that for all $2 \leq i \leq k$ tree T_i differs from T_{i-1} in that it displays the split S_i, in addition to all the splits displayed by T_{i-1}. This is done by letting T_0 be a single-vertex tree, whose vertex v is labelled by the set X. Now suppose that T_{i-1} has been generated by processing splits $S_1, S_2, \ldots, S_{i-1}$. Then T_i is produced from T_{i-1} by adding a new vertex and labelling that vertex such that the split S_i is also displayed by tree T_i. See Figure 2.5 for an example. See [110] for details on how this approach can be extended to general partitions and Chapter 4 for our implementation of it into the `ape` software package ([86]).

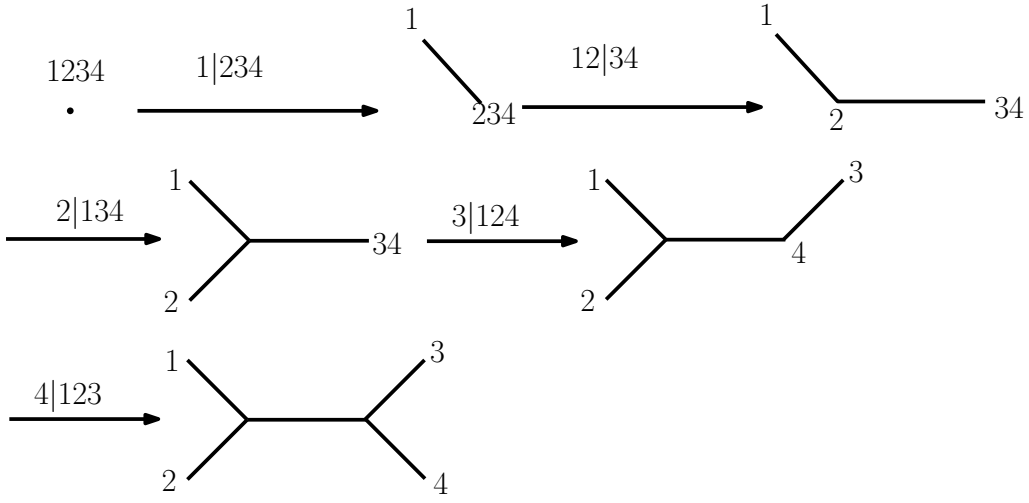


Figure 2.5: Steps of the tree popping approach, for a split system containing the split 12|34 and all the trivial splits. The splits which expand the tree are shown on the arrows.

2.4.2 Phylogenetic trees from sequences

The idea behind sequence-based tree reconstruction is to represent a taxon (e.g a species) x as a sequence, that is a string of symbols over a finite alphabet $\mathcal{A}$ (e.g $\mathcal{A} = \{A, G, C, T\}$ in the case of DNA sequences). In order to reconstruct a phylogenetic tree from a set of n sequences, an alignment needs to be constructed first for the sequences. This is represented by a matrix, such that each column of this matrix represents a *position* in the alignment, and the number of columns l is the length of the alignment. In addition to the initial alphabet, this matrix may contain a special gap character commonly denoted by $-$. The exact way of computing such an alignment has $O(l^n)$ runtime complexity, which severely limits its applicability to large datasets. To alleviate this problem, a greedy heuristic has been proposed in [54], which simultaneously constructs a phylogenetic tree (called a "guide tree") while aligning an input set of sequences. This greedy heuristic is prone to propagating suboptimal choices made early on in the algorithm, thus affecting the overall accuracy of the output alignment. To address this problem, a consistency principle was proposed in [83], which lead to the development of a generation of more accurate aligners such as T-Coffee

[83], Clustal-W [12] and MUSCLE [28]. More recently, Clustal-Omega has been proposed, which seems to hold great promise for aligning even large NGS datasets [111]. Key to its fast runtime is an embedding of the input sequence in an appropriately chosen space, which allows for highly efficient computation of the guide tree required for efficient alignment.

Given a set X of $|X| = n$ sequences, and an alignment over this set, the number of non-isomorphic binary phylogenetic trees that can be obtained for such an alignment is equal to $3 \times 5 \times ... \times (2n-5)$. For modern datasets, this number tends to be huge, implying that an exhaustive search is infeasible. To overcome this some heuristics have been developed to reconstruct such trees from this type of data. The most popular ones are the Maximum Parsimony ([29]), Maximum Likelihood ([36]) and Bayesian ([77]) based, and each of them employs a different optimization criterion.

2.4.3 Phylogenetic trees from distances

A popular method for reconstructing phylogenetic trees on some set X is via distances on X. Intuitively, the aim is to measure and display the degree of relatedness between two elements of X based on how far apart they are with respect to certain measures of dissimilarity. If the measure of dissimilarity captures the evolutionary signal in the data, then closely related elements in X should be grouped closer together in the returned phylogenetic tree T on X than those that are more distantly related. Distance methods are popular as phylogenetic tree reconstruction algorithms based on distances are efficient. However they tend to lose phylogenetic signal as they work on secondary data (distances from sequences) rather than primary data (sequences). Nonetheless, with an appropriately calculated distance, they have proven successful in modern phylogenetic studies as a means of getting a quick snapshot of the data or as a way of restricting the search of the MP and the ML approaches to hopefully only promising regions of the search space ([114]).

This section is concerned with distance based edge-weighted phylogenetic tree reconstruction and is structured as follows. We start by clarifying the link between phylogenetic trees on X and distances on X and then review some of the popular methods for constructing phylogenetic trees from distances.

We start by formalising the concept of a distance. A *distance* d on X is a symmetric

map $d : X \times X \longrightarrow \mathbb{R}_{\geq 0}$. As explained in Section 2.3, an edge-weighted phylogenetic tree (T, ω) on X gives rise to a distance $d_{(T,\omega)}$ on X. Now that we know how to go from edge-weighted phylogenetic trees on X to distances on X, we can ask the converse question: given a distance d on X, when can we go from d to an edge-weighted phylogenetic tree (T, ω) on X such that $d = d_{(T,\omega)}$?

2.4.3.1 Characterising edge-weighted phylogenetic trees in terms of distances

Interestingly, the distance $d_{(T,\omega)}$ constructed above from an edge-weighted phylogenetic tree (T, ω) is *additive*, where we say that a distance D on X is *additive* if for any $a, b, c, d \in X$ the following *four-point condition* holds ([9]):

$$D(a,b) + D(c,d) \leq max(D(a,c) + D(b,d), D(b,c) + D(a,d)) \tag{2.2}$$

If D is an additive distance on X, then there exists a unique edge-weighted phylogenetic tree (T, ω) such that $d_{(T,\omega)} = D$ ([9]). In this case we say that D is *displayed* by (T, ω). Therefore one of the main problems of interest in distance-based phylogenetic tree reconstruction is the following: given an additive distance D on X, how can we find an edge-weighted phylogenetic tree (T, ω) on X such that D is displayed by (T, ω). Since distances coming from real data are seldom additive, it is also of interest to develop methods that find 'good' edge-weighted phylogenetic tree estimates from distances which do not respect the four-point condition (i. e. inequality 2.2).

2.4.3.2 Reconstructing edge-weighted phylogenetic trees from distances

One of the most widely used algorithms for reconstructing phylogenetic trees from distances is the *neighbour joining algorithm* (NJ), described in [106]. For a given distance D on X it essentially iteratively reconstructs an edge-weighted phylogenetic tree of *minimum length* (i.e sum of lengths of all edges), with the restriction that the path length between any two leaves is at least as big as the respective entry in D. It does so by employing a greedy heuristic that approximates D [44].

To ease the representation of an outline of NJ, we call the elements of X taxonomic units. On a high level, NJ works as follows. Given a distance D on X. Then choose a pair x, y that minimises a certain criterion (intuitively, this criterion minimises the

total sum of edge-weights obtained if we merge x and y into a cherry). Next 'merge' x and y into a taxonomic unit z and calculate a new distance D' from D (following [106]) for the set $X' = X - \{x,y\} \cup z$. Also, compute the length of the edges $\{x,z\}$ and $\{y,z\}$. Finally, set $X = X'$ and repeat the above until $|X| = 2$. When this stage is reached, create a tree with two leaves and label them with the two elements of X. By reversing the merging process described above an edge-weighted phylogenetic tree on the original set X is then reconstructed. A more detailed explanation of the above process may be found in [44]. It is known that NJ is *consistent* in the sense that if $D = d_{(T,\omega)}$ for some edge-weighted phylogenetic tree (T,ω), then up to equivalence with T, (T,ω) will be returned by NJ.

Due to noise and variability in the data, it is generally too much to hope for that a given distance is additive. To alleviate this problem several NJ-like methods have also been proposed such as BioNJ [41], MVR [43], UNJ [42]. Based on simulation studies, all have been shown to have higher topological accuracy than NJ when the input distances are not additive ([17]). Common to all these methods is that they follow the same agglomerative framework as NJ, see [17]. Implementations of the above alternatives for NJ can be found in the package PhyD* ([17]).

Another method used for reconstructing phylogenetic trees from distances is the triangles method ([49]). Before describing this method we need to introduce some further terminology. We start with introducing a generalised form of a cycle, known as a $k-dcycle$. Let $k \geq 1$ and let $G = (V,E)$ be a graph. If we can find a set $C \subseteq E$, such that every vertex to which an edge in C is incident has degree at least $k+1$, then we say that C is a $k-dcycle$. For instance a cycle is a $1-dcycle$. A complete graph with 4 vertices is a $2-dcycle$. A graph that has no $k-dcycles$ is said to be $k-dacyclic$. If a graph is $k-dacyclic$ it is called a $k-d\ tree$. See Figure 2.6 for an example of a 2-d tree, or $twotree$ for short. In what follows, twotrees are assumed to have edge-weights.

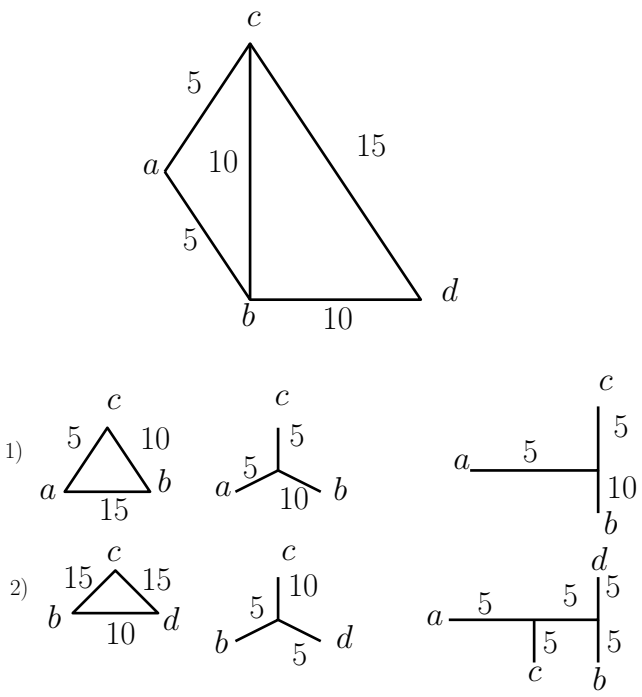


Figure 2.6: (i) An example of a twotree with leaves $\{a,b,c,d\}$. Edge weights are shown on the respective edges. The induced subgraph on $\{a,b,c\}$ is a triangle. (ii) The steps indicated as (1) and (2) carried out by the triangle method to transform the twotree in (i) into an edge-weighted phylogenetic tree.

With the above in mind, we next outline the triangle method. Given a distance D on X with $n = |X| \geq 2$, it essentially comprises of two parts. The first one is a greedy reconstruction of a twotree G_{n-2} associated to D, and the second comprises of reconstructing an edge-weighted phylogenetic tree (T_{n-2}, ω_{n-2}) on X from (G_{n-2}, ω_{n-2}), where $|X| = n$.

We start with presenting an outline of the first part of the algorithm. For D and X as above, the reconstruction of G_{n-2} from D is simply an adaptation of Prim's minimum-spanning tree algorithm [96]. We begin by choosing two elements x, y from X such that $D(x,y)$ is the smallest over all pairs of elements in X. We then create an edge-weighted graph G_0 with $V(G_0) = \{x,y\}$ and $E(G_0) = \{\{x,y\}\}$, with edge- weight function $\omega_{G_0} : E(G_0) \longrightarrow \mathbb{R}_{\geq 0}$, defined by putting $\omega_{G_0}(\{x,y\}) = D(x,y)$. Note that G_0 is clearly a twotree. We now look for an element $x \in X$ such that the distance $L(x)$

between it and G_0 is minimal. For Y a non-empty set, D' a distance on Y and G a graph with $V(G) \subseteq Y$, this distance was defined in [49] to be

$$L(x) = \frac{1}{2} min_{y,z \in Y, y \neq z}(D'(x,y) + D'(x,z) - D'(y,z)).$$

Suppose $y,z \in V(G_0)$ minimize the distance between x and G_0. Then we add two edges $e_1 = \{x,y\}$ and $e_2 = \{x,z\}$ to $E(G_0)$. This results in a new twotree G_1. We set $\omega_{G_1}(e_1) = D(x,y)$ and $\omega_{G_1}(e_2) = D(x,z)$. We continue adding elements of X in this manner until $V(G_{n-2}) = X$ holds, thus obtaining a sequence (G_i, ω_{G_i}) of edge-weighted twotrees, $0 \leq i \leq n-2$. This concludes the first part of the triangles method. An iteration of it is shown in Figure 2.7.

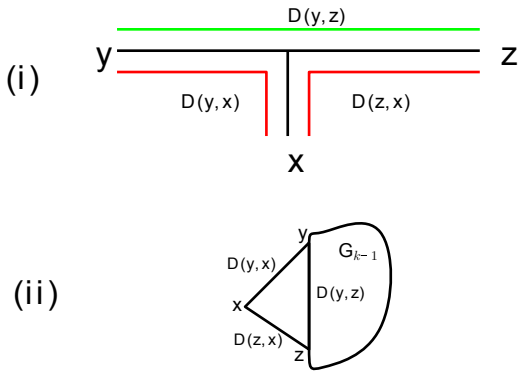


Figure 2.7: An iteration of the two-tree construction of the triangles method. (i) shows how the distance between x and an already constructed two-tree containing y and z is computed. The green distance is subtracted from the red distances. We are then left with twice the distance between x and the path between y and z, hence why we need to divide this difference by two. By finding the pair y,z minimising this quantity, we get the distance between x and the two-tree. (ii) shows how an already existing two-tree G_{k-1} containing y and z is extended to also contain x. The new vertex is attached to G_{k-1} via the edges $\{x,y\}$ and $\{x,z\}$, with edge weights as indicated.

The second part of the triangle method is concerned with obtaining an edge-weighted phylogenetic tree from the twotree found in step 1. For this, we take the twotrees in the order in which they were constructed in the first step. Let $x_1 = a$, $x_2 = b, x_3 ..., x_n$ denote the order in which the elements of X have been added to G_0. We first obtain from

(G_1, ω_{G_1}) an edge-weighted star-tree (T_1, ω_1) on $\{x_1, x_2, x_3\}$, that is an edge-weighted phylogenetic tree whose leaves are x_1, x_2, x_3 and which has a single interior vertex, u, such that

$$\omega_1(\{u, x_1\}) = \frac{1}{2}[D(x_1, x_2) + D(x_1, x_3) - D(x_2, x_3)],$$

$$\omega_1(\{u, x_2\}) = \frac{1}{2}[D(x_2, x_1) + D(x_2, x_3) - D(x_1, x_3)],$$

$$\omega_1(\{u, x_3\}) = \frac{1}{2}[D(x_3, x_1) + D(z, x_2) - D(x_1, x_2)].$$

See Figure 2.6 for a graphical representation of the second part of the triangle method. The justification for the above equations is similar to Figure 2.7, (i).

Suppose now that we have constructed an edge-weighted phylogenetic (T_{i-1}, ω_{i-1}) obtained from considering all twotrees up to G_{i-1}, $i \geq 2$. We now wish to obtain the edge-weighted phylogenetic tree (T_i, ω_i) from (T_{i-1}, ω_{i-1}) by considering (G_i, ω_{G_i}). Suppose $x \in V(G_i) \setminus V(G_{i-1})$, and that $\{x, y\}, \{x, z\} \in E(G_i) \setminus E(G_{i-1})$. We set $\alpha = \frac{1}{2}(D(z, x) + D(z, y) - D(y, x))$. Then we choose an edge $e' = \{v_1, v_2\}$ on the path between y and z in (T_{i-1}, ω_{i-1}) and subdivide it in T_i by a vertex v_x such the length of the path from z to v_x in T_i is α. We also add the edge $\{v_x, x\}$ to $E(T_i)$. We define $\omega_i : E(T_i) \longrightarrow \mathbb{R}_{\geq 0}$ as follows.

$$\omega_i(e) = \begin{cases} \omega_{i-1}(e) & \text{if } e \in E(T_{i-1}) \setminus \{e'\} \\ \frac{1}{2}(D(y, x) + D(x, z) - D(y, z)) & \text{if } e = \{v_x, x\} \\ \alpha - \sum\limits_{e \in E_{(T_{i-1}, \omega_{i-1})}(z, v_2)} \omega_{i-1}(e) & \text{if } e = \{v_x, v_2\} \\ \omega_{i-1}(\{v_1, v_2\}) - \omega_i(\{v_x, v_2\}) & \text{if } e = \{v_1, v_x\} \end{cases} \tag{2.3}$$

We proceed in this fashion until we obtain (T_{n-2}, ω_{n-2}). See Figure 2.8 for a depiction of the above equations. See Chapter 4 for our implementation of the triangles method into ape[91].

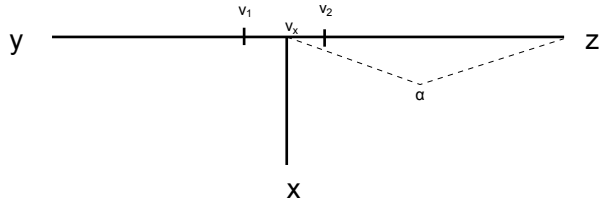


Figure 2.8: A depiction of how the new leaf x is attached to an already constructed phylogenetic tree on the path between y and z. The edge $\{v_1, v_2\}$ is subdivided by a vertex v_x to which we attach x. Edge weights are computed as in Equation 2.3.

2.4.3.3 Characterising rooted phylogenetic trees from distances

The algorithms presented in Section 2.4.3.2 work well for constructing edge-weighted phylogenetic trees. While such trees are useful for viewing groupings in the data, they suffer from the fact that they do not easily lend themselves to identifying e.g. ancestral relationships in the data. To this end, rooted edge-weighted phylogenetic trees (see Section 2.3) have been proposed. In this section we look at how edge-weighted *rooted* phylogenetic trees can be constructed from distances.

One way of obtaining edge-weighted rooted phylogenetic trees from distances is by first obtaining an edge-weighted phylogenetic tree (using e.g. NJ) and then rooting such a tree using for example, an outgroup. This method is advantageous because it provides a good way of going from edge-weighted unrooted phylogenetic trees to edge-weighted rooted phylogenetic trees which are easier to interpret from a biological point of view. It however suffers from the fact that finding an outgroup can be problematic.

Another popular way of obtaining edge-weighted rooted phylogenetic trees from distances on a set X is by assuming a *molecular clock*. Roughly speaking, this assumption postulates that all individuals in X evolve at a constant evolutionary rate. While it has been observed that such an assumption does not hold in general, if all individuals under consideration are from the same species, as is the case in many population-genetic studies, this assumption is reasonable ([51]). Put differently, the molecular clock assumption is assumed to be satisfied for studies concerning, for example, intra-specific evolution of a selectively neutral gene. As all elements in X are assumed to evolve at constant rates, an immediate consequence of this assumption on an edge-weighted phylogenetic tree (T, ω) depicting the evolution of taxa in X is the

identifiability of the root of T. In the ideal case it is located at equal path length from all leaves in X, in which case (T,ω) is called a *dendrogram*, or an *ultrametric tree*.

Suppose D is a distance on X. Similar to the unrooted case, if there exists an edge-weighted rooted phylogenetic tree (T,ω) on X such that $d_{(T,\omega)} = D$, we say that (T,ω) *displays* D. A distance D on X is displayed by a unique (up to equivalence) edge-weighted rooted phylogenetic tree if the following condition, known as the *ultrametric condition* holds for any $a,b,c \in X$ ([19]):

$$D(a,b) \leq max(D(a,c),D(b,c)) \tag{2.4}$$

Another way of obtaining edge-weighted[1] rooted phylogenetic trees on X is by attempting to transform a distance on X such that it respects the ultrametric condition. One way to do this is via the *Farris transform* ([34]). Alternatives include [64, 69]. Once such a distance matrix is computed one can apply Algorithm 1 to obtain a rooted phylogenetic tree on X.

2.4.3.4 Constructing rooted edge-weighted phylogenetic trees from distances

In this section, we review algorithms for constructing rooted edge-weighted phylogenetic trees from distances. Given a distance d on X, *Ascending Hierarchical Clustering* (AHC, Algorithm 1) can be used to produce a dendrogram (T,ω) on X. On a high level, AHC starts by putting each element of X in its own cluster. Then it chooses the two clusters that are closest under some distance D' on clusters on X and then merges them together into a new one. Using D', the distance between this new cluster and all remaining clusters are then computed. The algorithm continues in this fashion until there is just one cluster left. There are many flavours of this algorithm, all of which will produce the dendrogram (T,ω) for which $d = d_{(T,\omega)}$ holds if d is ultrametric, or a dendrogram which approximates d as closely as possible (in a least squared sense) if it is not. The only step where the flavours of this algorithm differ is in the computation of the distances between the newly merged cluster and the already given clusters. With C_i, C_j and C_u denoting three clusters, and C_k the cluster obtained by merging C_i and C_j, some of the types of distances for clusters of X considered are as follows:

[1] Note that some of the edge-weights might be negative which, strictly speaking implies that the reconstructed rooted phylogenetic tree is not a rooted *edge-weighted* phylogenetic tree.

- single linkage: $D'(C_k,C_u) = min(D'(C_i,C_u),D'(C_j,C_u))$
- complete linkage: $D'(C_k,C_u) = max(D'(C_i,C_u),D'(C_j,C_u))$
- average linkage: $D'(C_k,C_u) = 0.5(D'(C_i,C_u)+D'(C_j,C_u))$
- Wards minimum variance method [119]:

$$D'(C_k,C_u) = \frac{|C_i|+|C_u|}{|C_i|+|C_j|+|C_u|}D'(C_i,C_u) + \frac{|C_j|+|C_u|}{|C_i|+|C_j|+|C_u|}D'(C_j,C_u)$$
$$-\frac{|C_u|}{|C_i|+|C_j|+|C_u|}D'(C_i,C_j)$$

Algorithm 1 Ascending Hierarchical Classification (AHC)

Input: a distance d on a set X and a distance D on the powerset of X, s.t. $D(\{x\},\{y\}) = d(x,y)$ for all $x,y \in X$

Output: a dendrogram (T,ω) on X

$k = n = |X|$
$V(T) = \{v_x\}$ for all $x \in X$, $E(T) = \emptyset$
ret=$\bigcup_{x\in X}\{\{x\}\}$
index($\{x\}$)=0, for all $\{x\} \in$ ret
while $k > 1$ **do**
  let C_i, C_j be two clusters in ret such that $D(C_i,C_j)$ is minimal
  ret$\leftarrow$ ret$\backslash\{C_i,C_j\}$
  $C_k = C_i \cup C_j$
  ret$\leftarrow$ ret$\cup\{C_k\}$
  compute $D(C_k,C_u)$ for all $C_u \in$ret
  index(C_k) $= D(C_i,C_j)/2$
  $V(T) = V(T) \cup \{v_{C_k}\}$
  add the edge $e_1 = (v_{C_k}, v_{C_i})$ to $E(T)$
  add the edge $e_2 = (v_{C_k}, v_{C_j})$ to $E(T)$
  $\omega(e_1)$ =index(C_k)-index(C_i)
  $\omega(e_2)$ =index(C_k)-index(C_j)
  $k \leftarrow k-1$
end while

2.4.4 Characterising phylogenetic trees in terms of incomplete distances

As we have seen above, if a distance D on X respects certain conditions (Equation 2.4), D is displayed by a unique (up to equivalence of the underlying rooted phylogenetic tree) edge-weighted phylogenetic tree. However, what is needed is that the value $D(x,y)$ is known for all $x,y \in X$. But with real data this value need not always be available. Missing distances can occur due to e.g. experimental error, unreliable reads for example for genomic data, or incomplete taxonomic coverage ([17]). We are thus left with an incomplete distance on X, and are interested in finding an edge-weighted rooted/unrooted phylogenetic tree (T,ω) which correctly represents the given known distance values of D. Unlike the case where D is given for all pairs of elements in X, (T,ω) need not be uniquely determined by D if D has missing values (see Figure 2.9 for an example in the unrooted case).

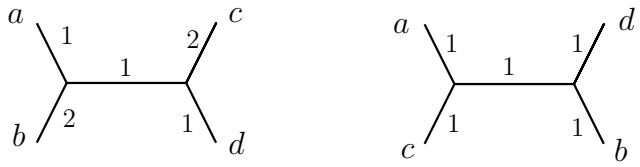


Figure 2.9: Suppose D is a distance on $X = \{a,b,c,d\}$ where only the distances between the pairs (a,b) and (c,d) are known. Then both trees depicted above with edge-weights as indicated have the same value for (a,b) and (c,d).

The structure of the remainder of this section is as follows. We begin by defining some concepts which are necessary for discussing incomplete distances. We then look at how incomplete distances define unrooted, and later rooted, phylogenetic trees. Finally, we look at algorithms which can be used to reconstruct phylogenetic trees from incomplete distances.

2.4.4.1 Incomplete distances preliminaries

In this section we review some preliminary concepts required to explain known results on how incomplete distances uniquely determine phylogenetic trees. Suppose that

(T,ω) is an edge-weighted phylogenetic tree on X. In what follows we consider an incomplete distance D on X in terms of a set of *cords*, i.e a set $\mathcal{L} \subseteq \binom{X}{2}$, where $\{x,y\} \in \mathcal{L}$ if and only if $D(x,y)$ is given. We denote an element $\{x,y\} \in \mathcal{L}$ as xy. If (T',ω') is a further edge-weighted phylogenetic tree, we say that (T,ω) and (T',ω') are $\mathcal{L}$-*isometric* if $d_{(T,\omega)}(x,y) = d_{(T'\omega')}(x,y)$ holds for all cords $xy \in \mathcal{L}$. Loosely speaking this means that (T',ω') is a competing tree of (T,ω) with respect to $\mathcal{L}$. We say that T' *refines* a phylogenetic tree T on X if, up to equivalence, T can be obtained from T' by collapsing edges of T' (see e. g. [110]). In that case, we will also call T' a *refinement* of T. Note that every phylogenetic tree is its own refinement.

As stated above, incomplete distances do not always uniquely define a phylogenetic tree. A set $\mathcal{L} \subseteq \binom{X}{2}$ with $\bigcup_{A\in\mathcal{L}} A = X$ is called a *lasso* if it contains some information which allows us to recover an edge-weighted phylogenetic trees to a certain extent (e. g. its edge-weights or topology). The following types of lassos have been proposed in [26] (and further studied in [56]) to capture the different ways in which an incomplete distance on X can uniquely determine an edge-weighted phylogenetic tree. Suppose T is a phylogenetic tree on X. We say that $\mathcal{L} \subseteq \binom{X}{2}$ is

(i) an *edge-weight lasso for* T if, for all proper edge-weightings ω and ω' of T, we have that $\omega = \omega'$ holds whenever (T,ω) and (T,ω') are $\mathcal{L}$-isometric

(ii) a *topological lasso for* T if, for every phylogenetic tree T' on X and any edge-weightings ω of T and ω' of T', respectively, we have that T and T' are equivalent whenever (T,ω) and (T',ω') are $\mathcal{L}$-isometric.

(iii) is a *strong lasso for* T if $\mathcal{L}$ is simultaneously an edge-weight and a topological lasso for T.

(iv) a *weak lasso for* T if, for every phylogenetic tree T' on X and any edge-weightings ω of T and ω' of T', respectively we have that T' is a refinement of T whenever (T,ω) and (T',ω') are $\mathcal{L}$-isometric.

We present examples of each kind of lasso in Figure 2.10.

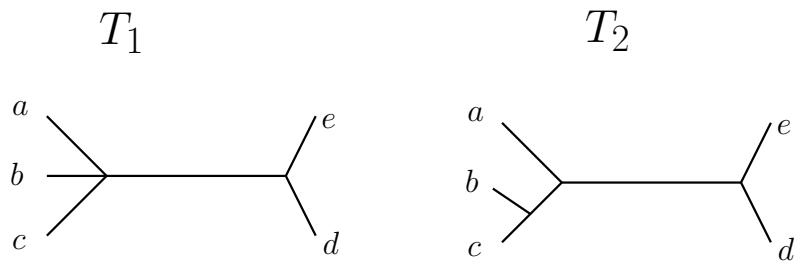


Figure 2.10: Suppose $\mathcal{L} = \{ab, ac, bc, be, ae, ce, ed, cd\}$ is a set of cords on $X = \{a, b, c, d, e\}$. Then $\mathcal{L}$ is both an edge-weight and a topological (and hence strong) lasso for the phylogenetic tree T_1 and $\mathcal{L}' = \mathcal{L} - \{bc\}$ is a weak lasso for T_1. Moreover, T_2 is a refinement of T_1.

For $\mathcal{L}$ as above, we denote by $\Gamma(\mathcal{L})$ the graph with vertex set X and edge set $\mathcal{L}$. See Figure 2.11 for an example. It was shown in [26] that if $\mathcal{L}$ is a topological lasso, then $\Gamma(\mathcal{L})$ is connected and if $\mathcal{L}$ is an edge-weight lasso, then $\Gamma(\mathcal{L})$ is strongly non-bipartite (i.e. every connected component is non-bipartite). The converse of the previous statements does not hold (see Figure 2.11).

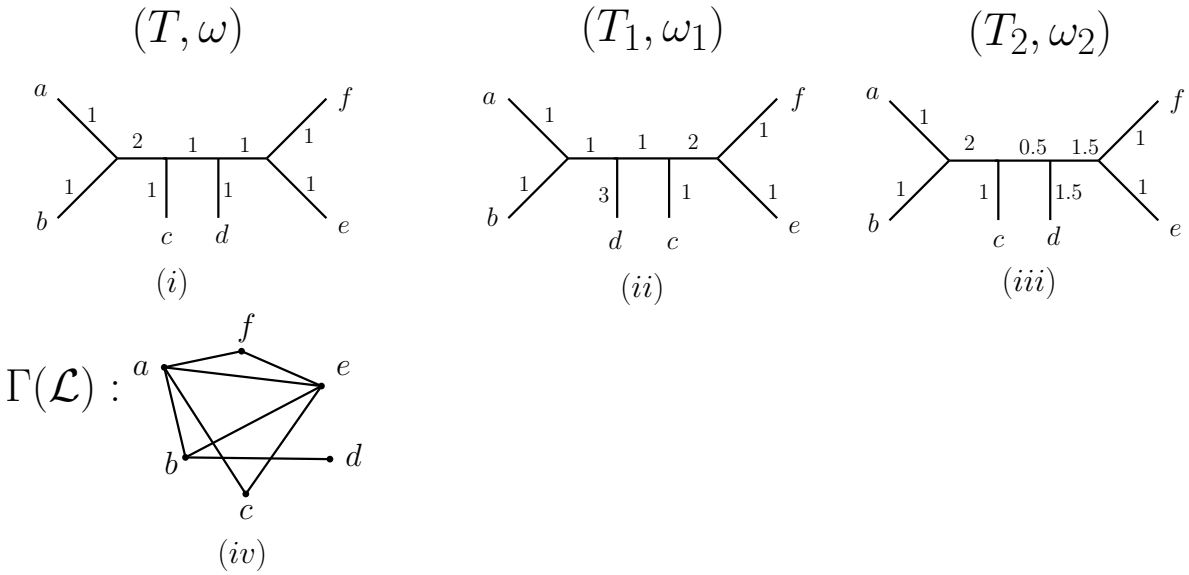


Figure 2.11: For the set $\mathcal{L} = \{ab, ae, be, ac, ce, bd, af, ef\}$ on $X = \{a,b,c,d,e,f\}$, the $\Gamma(\mathcal{L})$ graph is shown in (iv). As can be easily checked, (T,ω) and (T_1,ω_1) are $\mathcal{L}$-isometric, but T_1 is not equivalent with T. Hence $\mathcal{L}$ is not a topological lasso for T. Again as can be easily checked, (T,ω) and (T_2,ω_2) are $\mathcal{L}$-isometric and T_2 is equivalent with T_1 but $\omega_1 \neq \omega_2$, hence $\mathcal{L}$ is not an edge-weight lasso for T.

2.4.4.2 Characterising unrooted trees in terms of incomplete distances

In this section we present some known properties when a set $\mathcal{L} \subseteq \binom{X}{2}$ of cords uniquely determines a phylogenetic tree on X. All of the results reviewed here were established in [26], and the reader is referred to that paper for details on the proofs. We begin by reviewing certain types of sets of cords called *covers* which are known to be lassos for phylogenetic trees and then review the concept of a shelling.

Covers For the following let $\mathcal{L}$ be a set of cords and let T denote a phylogenetic tree on X. For every interior vertex of T, we denote the connected components of T obtained by deleting v as $C_1, C_2, ..., C_n$. If for every C_i we can find a set S_i of leaves of C_i such that for any pair C_l, C_j there exists $a \in S_l, b \in S_j$ such that $ab \in \mathcal{L}$ then we call $\mathcal{L}$ a *cover* of T. See Figure 2.12 for a schematic illustration of this property. In addition $\mathcal{L}$ is called a *triplet cover* for T if each of the sets S_i is of size one (i.e. one representative

leaf is chosen for each connected component at each interior vertex). The name stems from the fact that for binary phylogenetic trees each vertex is covered by a triplet of leaves.

It was shown in [26] that triplet covers are edge-weight lassos for binary phylogenetic trees. The authors of that paper also showed that if a set $\mathcal{L}$ of cords is a triplet cover for two $\mathcal{L}$-isometric edge-weighted phylogenetic trees (T,ω) and (T',ω'), then T and T' must be equivalent and $\omega = \omega'$.

Another type of cover of interest is the pointed cover. Suppose we fix an element $x \in X$. We say that $\mathcal{L}$ is a *x-pointed cover* of a phylogenetic tree T if $\mathcal{L}$ is a cover, and furthermore for every interior vertex v of T the following holds. There exist elements a, b of X such that $ax \in \mathcal{L}, bx \in \mathcal{L}$, and v lies simultaneously on the paths from a and b, a and x and b and x. If, for a cover $\mathcal{L}$ on X we have an element $x \in X$ as above, then we call $\mathcal{L}$ a pointed cover. In [26] it was shown that pointed covers are strong lassos of binary phylogenetic trees.

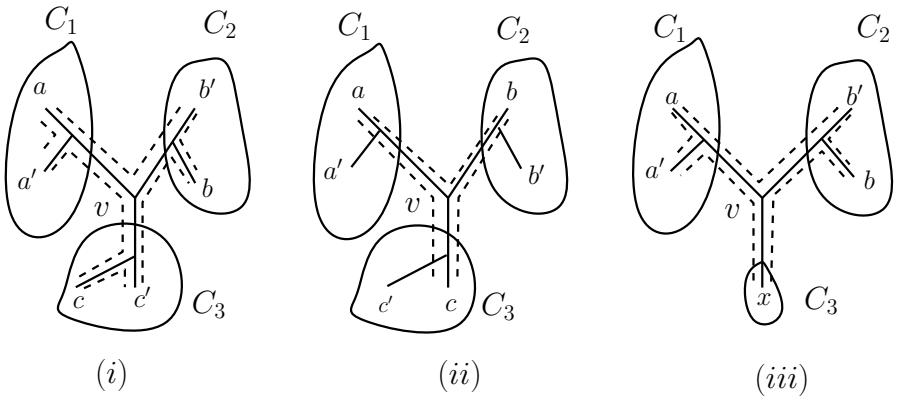


Figure 2.12: For v an interior vertex of a phylogenetic tree T and a,a',b,b',c,c' leaves of T, we illustrate the conditions needed to be satisfied by v for the types of covers reviewed here. A cover is depicted in (i). A triplet cover is depicted in (ii), and an x-pointed cover is depicted in (iii). For all 3 types of covers, we depict a cord xy between two leaves in $\mathcal{L}$ by a dashed line from x to y. For each example, we depict the connected components obtained by deleting v.

Suppose we have a split $A|B$ of a set of taxa X. Then the set $A \vee B = \{ab : a \in A, b \in B\}$ is called a *bipartite cover*. Suppose now T is binary. Then we say that

a bipartite cover $A \vee B$ is *t-strong* if for every cherry $C = \{a,b\}$ of T, we have that $A \cap C \neq \emptyset \neq B \cap C$. It was shown in [26] that t-strong bipartite covers are topological lassos but not edge-weight lassos (see Figure 2.13).

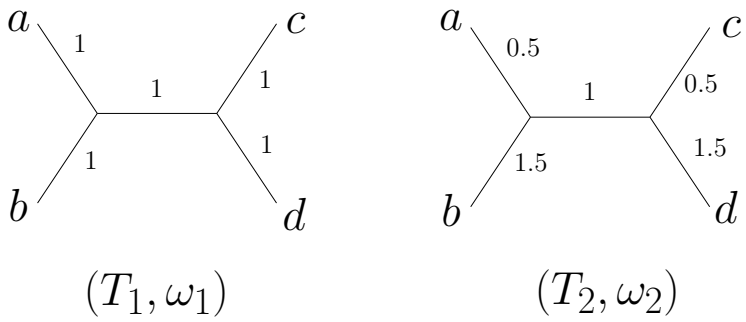


Figure 2.13: For the edge-weighted phylogenetic trees, (T_1, ω_1) and (T_2, ω_2), on $X = \{a,b,c,d\}$ depicted in $i)$ and $ii)$, the bipartite cover obtained from $ac|bd$ is t-strong. Note that $D_{(T_1,\omega_1)}(x,y) = D_{(T_2,\omega_2)}(x,y)$ for all $xy \in \{a,c\} \vee \{b,d\}$.

Shelling Given a phylogenetic tree T on X and a set $\mathcal{L} \subseteq \binom{X}{2}$ of cords. If $\mathcal{L}$ lassos T, then it is hoped that the cords in $\mathcal{L}$ can somehow be used to infer other cords not given in $\mathcal{L}$. This gives rise to the concept of a shelling, which can be seen as a way of iteratively inferring a missing distance for the cords on $\binom{X}{2}$ from the cords contained in $\mathcal{L}$. Hence we say that $\mathcal{L}$ is *T-shellable* if the following holds. There exists an ordering $a_1b_1, a_2b_2, \ldots a_mb_m$ of the cords in $\binom{X}{2} \setminus \mathcal{L}$ such that for any a_ib_i there exists two pivot elements $x_i, y_i \in X \setminus \{a_i, b_i\}$ such that with $Y = \{a_i, b_i, x_i, y_i\}$ the following property is satisfied. The cord x_iy_i must be such that the quartet $a_ix_i|b_iy_i$ is displayed by T, and such that all elements of $\binom{Y}{2}$ except a_ib_i are in $\mathcal{L} \cup \{a_1b_1, a_2b_2, \ldots, a_{i-1}b_{i-1}\}$.

Any such ordering is called a *T-shelling* of $\binom{X}{2} \setminus \mathcal{L}$ and if a set $\mathcal{L}$ has such a T-shelling, it is called an *s-lasso* for T. It was shown in [26] that s-lassos of any phylogenetic tree T are in fact strong lassos of T. See Figure 2.14 for an example of a shelling.

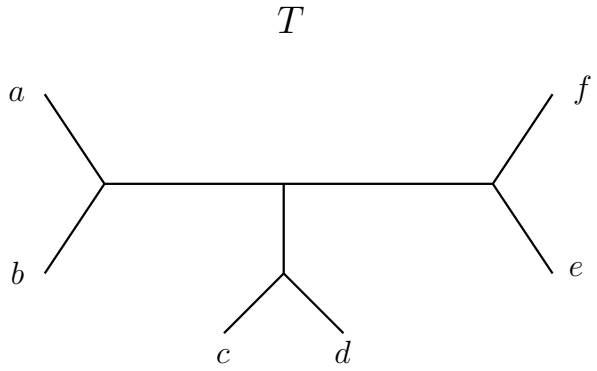


Figure 2.14: The set of cords $\mathcal{L} = \{ab, ae, be, af, fe, bc, cd, bd, ac, ec\}$ is a shelling for the depicted phylogenetic tree T on $X = \{a, b, c, d, e, f\}$. The missing cords in $\binom{X}{2}$ can be inferred from the quartets induced by T in the following order: bf, from quartet $ab|fe$; ad, from quartet $ab|cd$; ed, from quartet $ae|cd$; fc, from quartet $ac|fe$; fd, from quartet $af|cd$.

Note that although similar in nature, twotrees and shellings are two different concepts. More precisely, every twotree is a shelling, but there exist shellings that are not twotrees. An example of this fact is provided by the phylogenetic tree T in Figure 2.15(i) and the set $\mathcal{L}$ of cords depicted in Figure 2.15(ii) in the form of $\Gamma(\mathcal{L})$. This example was constructed by the author and is published in [56]. It is easy to check that the cords in $\binom{X}{2} \setminus \mathcal{L}$ have a shelling ordering given by $ab, ab', b'x, b'y, xa', xa'', ya', ya'', ba', ba''$, with pivot elements $(x,y), (a',a''), (a,b), (a,b), (a,b'), (a,b'), (a,b'), (a,b'), (a,b'), (a,b')$, respectively. But obviously $\Gamma(\mathcal{L})$ is not a twotree.

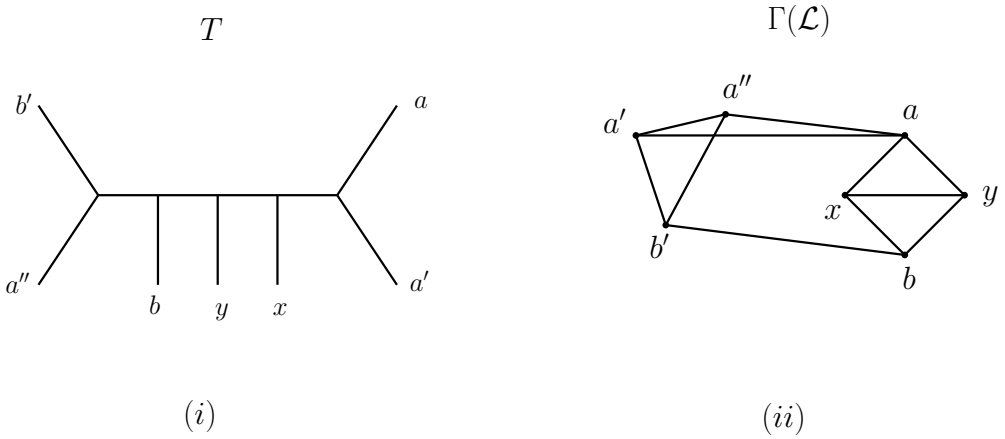


Figure 2.15: (i) a binary phylogenetic tree T. (ii) the $\Gamma(\mathcal{L})$ for a set $\mathcal{L}$ of cords.

2.4.4.3 Characterising rooted trees in terms of incomplete distances

A lot of work has been put into understanding when incomplete distances uniquely determine edge-weighted phylogenetic trees [56, 33]. While many interesting results have been established, which present certain easy to check conditions for when a set of cords can be an edge-weight, topological, weak/strong lasso, a general characterisation of such lassos is still missing [26]. In Chapter 3 we extend the above concepts to rooted phylogenetic trees.

2.4.5 Reconstructing phylogenetic trees from incomplete distances

In the previous section we reviewed results which told us how and to what extent incomplete distances determine phylogenetic trees. In this section we review algorithms which are used to reconstruct phylogenetic trees from incomplete distances. We start with algorithms for phylogenetic trees and continue with reviewing incomplete distance reconstruction of rooted phylogenetic trees.

2.4.5.1 Unrooted phylogenetic trees from incomplete distances

In [17] several methods have been proposed for reconstructing phylogenetic trees from incomplete distances. They adapt all of the steps mentioned for agglomerative methods for complete data to handle missing entries in the input distance matrix. The methods

have the advantage that they are very fast. Also simulations indicate that they work well in practice (see [17]). An alternative method for edge-weighted phylogenetic tree reconstruction from incomplete distances which does not use the agglomerative framework is presented in [49]. This method has the advantage that it is interesting from a theoretical point of view when it comes to studying edge-weighted phylogenetic tree reconstruction from incomplete distances.

The above methods attempt to handle missing distance information by building phylogenetic trees around the missing entries. Methods which make use of the additive and ultrametric (see Section 2.4.3) inequalities to fill in missing distances are presented in [74]. Then construction methods for complete data are used to produce an edge-weighted phylogenetic tree. We conclude by remarking that all of the methods in this section have been implemented in the popular R-based ape phylogenetics analysis package (see Chapter 4 for details).

2.4.5.2 Rooted phylogenetic trees from incomplete distances

In [19] a method for reconstructing dendrograms from incomplete distances was proposed. On a high level, this method estimates a complete distance matrix, D', from the incomplete input distance matrix D such that D' is the closest to D in the least-squared sense (see [19] for more details). Then Algorithm 1 is applied to D' to obtain a dendrogram.

2.5 Predictive Breeding, Association Mapping and Population Stratification

Predictive breeding is concerned with optimising artificial breeding practices for certain traits of interest, and although it is also carried out for animals, in this review we focus on plants. Breeding for plants that have desirable traits for the environment they are cultivated in is a central task for productive and qualitative agriculture. The classical strategy for doing this is via expensive and time-consuming field trials, which tends to involve the growing of a large number of different populations of crops, and selecting for future crop breeding only those individuals which have those traits. This method has many disadvantages, including:

- The necessity of field trials, which can only be conducted in particular locations and at particular times of the year
- The possibility of transmitting undesirable traits together with the traits of interest
- The difficulty of obtaining traits with low heritability

An alternative to the above strategy is predictive breeding, an example of which is *Marker-Assisted Selection* (MAS, see [101]). Here, instead of growing the plants in order to observe their phenotypic properties, one uses genetic information found in seeds to determine which traits the plant will exhibit once grown. This eliminates the need for expensive field trials and uncertainty in observed phenotypes. However a link between the plant's genotype and phenotype is required, and finding such a link is no easy task. Next generation sequencing has become central to developments in this field, by allowing for unprecedented amounts of input data to the above strategy. While it provides increased accuracy and power for MAS, it also introduces a series of computational challenges. Before going into more detail about genotype-phenotype mapping, we review some basic terminology below.

Quantitative traits are agriculturally important traits (e.g yield, disease resistance) that are controlled by possibly many genes. The particular locations of the traits on the genome are known as *Quantitative Traits Loci* (QTL) and checking for their presence or absence in a crop dataset allows one to make predictions about which traits each of those crops will exhibit.

Usually QTLs are identified relative to a *marker* which represents an observed difference between organisms or species that make up a dataset. With the advent of NGS, genetic markers are quickly becoming the norm in QTL-identification.

Genetic markers can be viewed as observed differences on the sequence level. They can be thought of as flags that can pinpoint the location of the gene on its genome. This is reflected in the terminology used for such markers. For example a genetic marker that is close to a gene is said to be *linked* to that gene. Like genes, genetic markers occupy certain positions on the genome, known as loci. A particular type of DNA marker is a *single nucleotide polymorphism* (SNP). Biologically, SNPs represent single base-pair differences between genotypes of interest.

The remainder of this section is structured as follows. We begin by giving a brief overview of approaches for finding genotype-phenotype association. Subsequent to this we go into more detail about *population structure*, which is a confounding factor in finding genotype-phenotype associations. We conclude by reviewing several popular approaches which aim to infer population structure from genotype data.

2.5.1 Genotype-Phenotype association

This section aims to review popular methodologies for genotype-phenotype association. There are two popular classes of methods for finding links between phenotypes and genotypes. We give a brief high-level review of both.

The first class is family-based linkage mapping, or just linkage mapping. In this case, a purpose-built population, called a mapping population, is created by breeding some founder individuals which differ for relevant traits of interest. The purpose is to obtain offspring that are homozygous (i. e. identical genes obtained from both parents) combinations of the founders' genotype. Once such a population is obtained, its constituents are genotyped and have their phenotypes observed. It is hoped that differences in phenotypes in the offspring could be explained by the observed genotype differences between them. Ideally, these offspring are sufficiently fine-grained recombination of the parental genotypes to allow pinpointing the markers linked with the traits of interest. In practice however, linkage mapping suffers from the fact that a mapping population of sufficient size and diversity is difficult to obtain. See [13] for details on how this approach works.

Another popular approach for identifying QTLs is *association mapping*, or *Linkage Disequilibrium Mapping (LD)*. As opposed to linkage mapping (which is applied to a purpose built mapping population), association mapping is applied to a collection of accessions (e. g. individuals) which spans multiple geographical regions, is of differing crop types, and is assumed to have had greater opportunities for recombination and thus a very diverse history. Association mapping therefore has a much higher resolution than linkage mapping rendering it preferable for identifying complex traits among very diverse natural populations (e. g. germplasm collections). A simple approach for finding genotype-phenotype association in such a dataset is via a case-control study, whereby one simply sequences those individuals which vary for a trait of interest and

effects statistical tests to attempt to find loci which can explain the variability in the phenotype. However, unaccounted for relationships between individuals of this natural sample can introduce false-positive associations [124]. A well known example where this has confounded a study is described in [50]. The authors of the study sought to find a link between genotype and chopstick usage, however the genomic region which they found linked to chopstick usage was subsequently found to be an immune system gene which had no relation to the original trait of interest. This is due to the prevalence of this gene in the sampled case population and not because of any functional links between it and chopstick usage.

As it turns out, this is an important problem in many such studies and must be controlled for when realising statistical tests. This confounding factor is called *population stratification (structure)*, and arises as a result of reproductive isolation amongst the individuals of a population. We continue by giving a brief overview of how population structure can arise and affect association studies and proceed with reviewing popular computational methods used for inferring it.

2.5.1.1 Population stratification

Population stratification (structure) may be viewed as reproductive isolation between individuals of a population. This can be due to, for example, social structure or geographical separation of species. Over time, allele frequencies between isolated groups tend to diverge. Now, if isolated groups differ for a trait of interest, all loci at which the two groups have different allele frequencies will be associated with the QTL for that trait ([123]). This gives rise to a high rate of spurious genotype-phenotype association. In this section we give a brief overview on what population structure is and how it may be modelled.

Suppose an initially homogeneous population is split into K different groups which cannot interact with each other for a prolonged period of time. Due to e. g. selective pressure or genetic drift, the allele frequencies of these K groups will start to differ from one another. Given enough time, said differences can become significant. Suppose now that the K populations are allowed to once again interact. Then the resulting offspring will be a mixture of the K populations. They are said to be *admixed*, and the K populations are called the *founder (populations)* of the dataset. The $n \times K$ Q-matrix

of an admixed dataset, n the number of individuals, stores in each row the proportion of an individual's genotype that came from one of the K founder populations. See Figure 2.16 for an example.

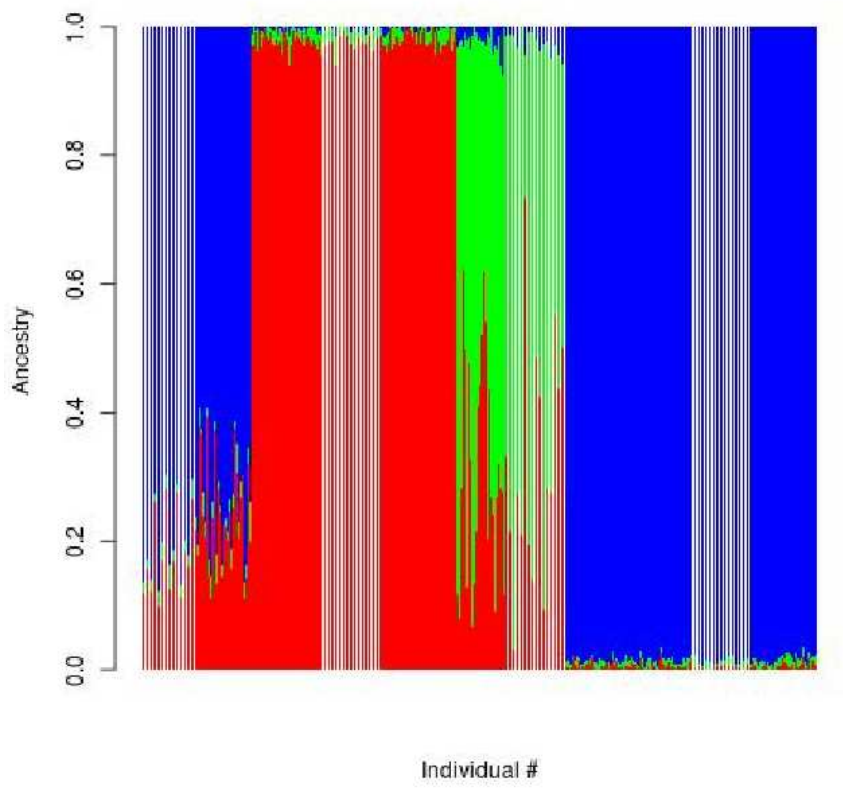


Figure 2.16: A Q-matrix depicted in terms of a barplot, with each colour corresponding to one of the three founder populations. The scale of the bars on the Y-axis represent proportions each of them contribute to individuals which label the X-axis.

Admixed datasets can lead to high false-positive rates in subsequent association studies. In order to correct for this effect, population structure needs to be included in the association model, via e. g. using a Mixed Linear Model (MLM) [123]. Most modern association tools (e. g. [4]) can take into account population structure which is generally either represented in the form of a Q-matrix, or in the form of principal components (PCs) obtained by applying PCA to the admixed dataset. Finding either is

of huge current interest, and in Section 2.5.2 we review popular methods for obtaining them.

2.5.2 Modelling Population Stratification

As alluded to above, *population stratification*, or *population structure*, can be seen as reproductive isolation between individuals of naturally occurring populations [97]. This has the effect of inducing genetic differences between isolated sub-populations which can cause spurious genotype-phenotype association. In this section we review a framework which has proven quite successful for modelling this effect [97]. To aid with this, we refer the reader to Figure 2.17 for a schematic representation. We also introduce common notation used throughout in the context of population structure.

Notation and terminology Throughout this section, we assume that $\mathbf{X}$ is a dataset given in terms of a $N \times L$ genotype matrix, comprising $L \geq 1$ loci (e. g. a SNP) and $N \geq 1$ individuals. Each individual i we view as a genotype vector $\mathbf{x}_i = (x_{il})$ where x_{il} denotes the genotype of i at locus l. For SNP datasets in particular, genotype information can be converted into a numeric matrix by simply having each entry store the count of the reference allele used to construct the dataset. If such information is not available, one could fix one of the alleles at the locus to be the reference allele [30]. We denote by $K \geq 0$ the number of ancestral (founder) populations of a dataset. For convenience, we refer to such an ancestral population as simply a founder. We denote by $\mathbf{P} = (p_{kl})_{\substack{1 \leq l \leq L \\ 1 \leq k \leq K}}$ a vector of frequency information, where each component of p_{kl} is a distribution of observed genetic information (e.g. alleles, genotypes) at locus l in founder k (see Q-matrix below). The actual genetic information and distribution used varies by method, hence we go into details about this in the subsections below. Furthermore, for convenience we denote by p_{klj} the probability of allele j given by the distribution p_{kl}.

In addition, we denote by Q an *ancestry matrix* of $\mathbf{X}$ which is an $N \times K$ matrix which stores for all individuals $1 \leq i \leq N$ in each row the *admixture vector* q_i for all individuals $1 \leq i \leq N$. The K components of this admixture vector represent the proportion of i's genotype that came from each of the K founders. Also we denote by $\mathbf{Z}$ the *founder matrix* of $\mathbf{X}$, which is an $N \times L$ matrix, where each entry z_{il} denotes the

founder of locus l of individual i.

Assumptions Throughout this section we make the following assumptions. Firstly, it is assumed a given population has been split up into $K \geq 2$ sub-groups (founder populations), which are evolving independently of each other and thus induce reproductive isolation. As stated in Section 2.5.1.1, such a scenario can confound association studies, and in order to correct for its effects on a dataset $\mathbf{X}$, we need to estimate the ancestry matrix of $\mathbf{X}$. For this, we first need to formalise how the individuals of $\mathbf{X}$ have arisen. To this end, we assume that $\mathbf{X}$ contains individuals which have arisen from a naturally occurring population. Since $\mathbf{X}$ might display stratification, we distinguish between *admixed* and *non-admixed* individuals.

To generate an admixed individual i, one first randomly samples the founder of a locus l, $1 \leq l \leq L$, from a multinomial distribution parametrized by q_i. Denote that sample by z_{il}. Then the genotype of l is generated by drawing from $\mathbf{P}_{z_{il}l}$. This is repeated for all loci across all individuals, obtaining the genotype matrix $\mathbf{X}$, as well as the founder matrix $\mathbf{Z}$. Note that a non-admixed individuals is just a special case of the above scenario, in that one of the components of q_i has value 1 and all other components have value 0.

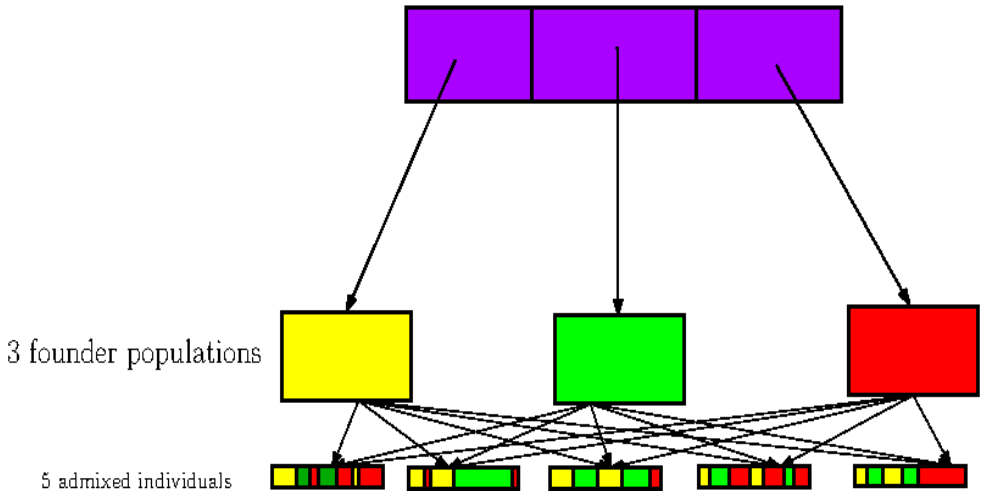


Figure 2.17: Schematic representation of population structure.

The above framework allows for modelling naturally occurring population struc-

ture with a good degree of accuracy. It acts as a blueprint for algorithms underpinning many popular population structure inference algorithms, some of which we review here. This also includes a brief review of computational techniques underpinning them, such as Markov-Chain Monte-Carlo [112] sampling to Expectation-Maximization (EM) [20].

2.5.3 STRUCTURE

STRUCTURE takes as input a matrix of genotypes (and optionally a value for K) and produces estimates for $\mathbf{P}$ and Q. In its review, we follow [98]. For that it makes assumptions on how the individuals of a population have arisen. We next review those assumptions, at the same time developing a framework within which we then present our review of STRUCTURE. Reflecting the centrality of Markov Chains to STRUCTURE's inner workings, we also review two popular approaches for constructing them: Metropolis-Hastings and Gibbs Sampling.

The model underpinning STRUCTURE models each of the $K \geq 2$ founders as a vector of allele frequencies for each locus. It also assumes that the marker loci are in *linkage equilibrium* with each other (i. e. transmitted independently to offspring), and that each founder is in Hardy-Weinberg equilibrium (i. e. no bias towards homozygous genotypes [113]). It takes as input a genotype matrix $\mathbf{X}$, and optionally K, and attempts to infer $\mathbf{P}$ and Q by sampling from the posterior distribution of these vectors given the observed data $\mathbf{X}$.

In essence STRUCTURE works by attempting to assign each individual's genome to one of the K founders whilst simultaneously attempting to estimate allele frequencies at each locus of an individual. For this, it combines a-priori assumptions about the nature of the model parameters (which we describe in relevant places below) with the observed data to obtain a *posterior distribution* of model parameters. Since this distribution is generally too complicated to compute, STRUCTURE employs an approach called Markov Chain Monte Carlo to sample from such a distribution. Since Markov Chains are central to this approach, we next review two popular algorithms for constructing such a chain. We then detail how the aforementioned framework (Section 2.5.2) for modelling population structure is applied to STRUCTURE.

Markov Chain - Monte Carlo

STRUCTURE relies on Markov Chains [45] to obtain random samples from a distribution. We present an outline of how such a chain may be constructed by first describing a discrete-time random process, of which a Markov-Chain is a special case. Before giving details below we remark that a discrete-time random process is generally used to model a system which is in a certain state at each step and which changes states between steps randomly. Each step can be seen as a moment in time. Examples of such a process include random walks, or random branching processes. Note that while the states of the system can be continuous in a discrete time random process, the steps must be discrete.

Formally, a *discrete-time random process* is a sequence of random variables indexed by a discrete variable (time). Then a *Markov Chain* is a discrete time random process where the next state of the chain depends only on the present state. More precisely, we have $P(\theta_n = x|\theta_1, \theta_2, ..., \theta_{n-1}) = P(\theta_n = x|\theta_{n-1})$, where $\theta_1, \theta_2, ...$ are the time-indexed random variables of the Markov Chain and x is some fixed value.

In the case of a discrete state-space, a Markov Chain can be represented as a $n_t \times n_t$ (n_t the number of states of the space) transition table τ, where each entry τ_{ij} of τ is the probability of moving from state i to state j. Where we have a *continuous* i. e. not discrete state-space (as is the case for STRUCTURE), the next state of the Markov Chain is drawn from a distribution conditioned only on the previous state.

Markov Chains with so-called stationary distributions (see below) are very useful to sample a vector θ from a statistical distribution, $D(\theta)$ on θ. In general $D(\theta)$ is too complicated to sample from directly. To overcome this problem the following approach (which we first outline and then formalise) is widely used. For a parameter vector θ, let $\pi(\theta)$ denote the stationary distribution of a Markov Chain on θ. This means that for large enough i (i. e. after a large enough number of steps), the state θ_i will represent approximately random samples from $\pi(\theta)$. For instance, if θ is a convex vector (i. e. it contains positive numbers which add up to 1), $\pi(\theta)$ could be a Dirichlet distribution, since samples from such a distribution are convex vectors. Note that certain conditions need to be respected by a Markov Chain for it to have such a distribution (see [82]). Since all Markov Chains under discussion here respect them, we will not go into detail about them.

Capturing the idea that after long enough time i the values θ_i will converge to random samples from $\pi(\theta)$, a Markov Chain has a *burn-in period* m. The chain elements obtained before θ_m are usually ignored. The above may be formalised as follows.

Suppose $\theta_1, \theta_2, \ldots$ is a Markov Chain for a parameter vector θ. Let m denote its burn-in period and $c \geq 0$ its *thinning interval* i. e. the states θ_{m+c}, θ_{m+2c},... give independent samples from the chain's stationary distribution $\pi(\theta)$. This procedure is called Markov Chain Monte Carlo sampling. We remark in passing that the quality of Markov Chain Monte Carlo sampling is dependent on the choice of m and c, and care has to be taken when choosing their values. See e. g. [98] for a discussion of appropriate values to consider. STRUCTURE uses algorithms like Metropolis-Hastings and Gibbs Sampling to construct a Markov Chain with stationary distribution $\pi(\theta)$.

Metropolis-Hastings For a parameter vector θ, Metropolis-Hastings [80] is an algorithm for constructing a continuous-state Markov Chain with stationary distribution $\pi(\theta)$ as follows. Assume that the value θ_t for θ at time t has been sampled. Given a user-specified distribution $Q(\theta'|\theta_t), t \geq 1$, (e. g. a Gaussian distribution with mean set to the previous state θ_t and a given variance) where θ' denotes a sample drawn by Metropolis-Hastings from $Q(\theta'|\theta_t)$. Then we associate to θ' a quantity α defined as:

$$\alpha = \frac{\pi(\theta')Q(\theta_t|\theta')}{\pi(\theta_t)Q(\theta'|\theta_t)}. \tag{2.5}$$

To obtain the state θ_{t+1} from θ_t Metropolis-Hastings puts $\theta_{t+1} = \theta'$ if $\alpha \geq 1$ (i.e. we accept θ' as our next state). Else, we set $\theta_{t+1} = \theta'$ with probability α or $\theta_{t+1} = \theta_t$ with probability $1 - \alpha$ (i.e. we accept θ' as the next state with probability α). It is shown in [80] that the Markov Chain $\theta_1, \theta_2, \ldots$ obtained in this way has stationary distribution $\pi(\theta)$. STRUCUTRE makes use of Metropolis-Hastings when sampling from the model with admixture.

Gibbs Sampling Gibbs Sampling [10, 45] is a further algorithm for constructing a Markov Chain with stationary distribution $\pi(\theta)$. Gibbs sampling naturally lends itself to cases where the full joint distribution $\pi(\theta)$ is hard to compute, but conditional distributions $\pi(\theta^1|\theta^2, \theta^3, ..., \theta^s)$ are easy to find for $\theta = (\theta^1, \theta^2, \theta^3, ..., \theta^s)$, where $s \geq 1$. In essence, it works as follows. Suppose our state at time t is in the form of

a vector $\theta_t = (\theta_t^1, \theta_t^2, ..., \theta_t^s)$ where for any step $t \geq 0$ and any $1 \leq i \leq s$, we denote by θ_t^i the value for component i at step t. Then θ_0 is initialised by giving it random values. Now suppose we want to sample θ_t^k, and that θ_t^l has already been sampled for all $1 \leq l \leq k-1$ and that θ_j , $1 \leq j \leq t-1$ has already been sampled. We draw θ_t^k from the distribution $\pi(\theta^k|\theta_t^1, \theta_t^2, ..., \theta_t^{k-1}, \theta_{t-1}^{k+1}, ..., \theta_{t-1}^s)$. For example, we draw θ_t^1 from $\pi(\theta^1|\theta_{t-1}^2, \theta_{t-1}^3, ..., \theta_{t-1}^s)$. Since conditional distributions for STRUCTURE's underlying population structure model are relatively easy to compute, Gibbs Sampling lies at the heart of the method.

Models used by STRUCTURE

Next, we review the population model employed by STRUCTURE, and how the general framework for modelling population structure in Section 2.5.2 applies to it. We first consider the case where no admixture is assumed, that is that every individual originates in only one of the $K \geq 2$ founders. We then present the more complicated case where we allow admixture. Given that structure accepts markers which are not always bi-allelic (microsatellites, RAPD, AFLP etc.) it is required that each locus l has J_l observed alleles across all individuals. Also assume that the number K of founders for the dataset is given.

Model with no admixture

Assuming each locus l of an individual in a population $\mathbf{X}$ has J_l alleles, we interpret the vectors in Section 2.5.2 as follows.

- $\mathbf{Z}$ is the founder vector of individual i of $\mathbf{X}$, whose components are denoted z_i
- $\mathbf{P}_{kl} = (p_{klj})$ is a vector of multinomial distributions for alleles $j = 1, 2, ..., J_l$ of locus l of founder $k = 1, 2..., K$

The purpose of the model is to estimate $\mathbf{Z}$ and $\mathbf{P}$ given $\mathbf{X}$. To do this STRUCTURE uses a Bayesian approach. More precisely, using Bayes' Theorem we have

$$P(\mathbf{Z}, \mathbf{P}|\mathbf{X}) \propto P(\mathbf{Z})P(\mathbf{P})P(\mathbf{X}|\mathbf{Z}, \mathbf{P}). \tag{2.6}$$

Put differently, the probability distribution of the founder vector $\mathbf{Z}$ and allele frequency vector $\mathbf{P}$ given $\mathbf{X}$ is proportional to the probability of the vector $\mathbf{Z}$ times that of the

vector $\mathbf{P}$ (both of which are given by some assumed prior distributions) times that of the probability of the data given $\mathbf{Z}$ and $\mathbf{P}$.

Since the resulting probability distribution $P(\mathbf{X}|\mathbf{Z},\mathbf{P})$ of $\mathbf{X}$ can generally not be computed exactly, Gibbs Sampling is used to sample from it. The vectors $\mathbf{Z}$ and $\mathbf{P}$ are found using certain summary statistics of the samples for $\mathbf{Z}$ and $\mathbf{P}$ (e. g. mean values).

With the model given in Equation 2.6 in mind, the framework in Section 2.5.2 can be applied as follows. Assume for the moment that the founder of an individual i is known. Then the genotypes at each locus are obtained by drawing a sample from the respective founder's allele frequencies (multinomial) distribution for that locus. More formally the probability $P(x_{il} = j|\mathbf{Z},\mathbf{P})$ of individual i having allele j at locus l is given as

$$P(x_{il} = j|\mathbf{Z},\mathbf{P}) = p_{z_i l j} \tag{2.7}$$

Central to Equation 2.6 are the computation of the quantities $P(\mathbf{Z})$ and $P(\mathbf{P})$, which are the prior distributions for $\mathbf{Z}$ and $\mathbf{P}$ respectively. If no prior knowledge of the proportions of individuals in each population is known a then to obtain $P(\mathbf{Z})$, a uniform distribution is assumed and we have

$$P(z_i = k) = \frac{1}{K} \tag{2.8}$$

To obtain a prior for $\mathbf{P}$, we assume the a-priori distribution $P(\mathbf{P})$ of allele frequencies p_{kl} of a population k at a particular locus l, to be modelled by a Dirichlet distribution (i.e a beta distribution for a multinomial) which we denote by $\mathcal{D}(\lambda_1, \lambda_2, ..., \lambda_{J_l})$. This distribution gives the probability of having a particular vector p_{kl} as the allele frequencies of founder k at locus l, and is used to model the allele frequencies of each locus independently. For all $1 \leq j \leq J_l$, the parameters λ_j control the expected frequency of allele j in the frequency vector p_{kl} for l. To simplify matters, STRUCTURE assumes all $\lambda_j = 1$ for all $1 \leq j \leq J_l$ which gives a uniform distribution of the allele frequencies of a locus.

To obtain random samples from the distribution given in Equation 2.6, STRUCTURE uses Gibbs Sampling as outlined in Algorithm 2.
where $\mathbf{Z}_t$ and $\mathbf{P}_t$ are estimates for $\mathbf{Z}$ and $\mathbf{P}$ respectively, at step t of the Gibbs Sampling algorithm. In other words, $\theta_t = (Z_t, P_t)$.

Algorithm 2 Gibbs Sampling for the model without admixture

Input: A genotype matrix $\mathbf{X}$.

Output: A series of random samples for $\mathbf{Z}$ and $\mathbf{P}$ from Equation 2.6.

Initialize $\mathbf{Z}$ by drawing values from the distribution given in Equation 2.8.

1. Sample $\mathbf{P}_t$ from $P(\mathbf{P}|\mathbf{X},\mathbf{Z}_{t-1})$
2. Sample $\mathbf{Z}_t$ from $P(\mathbf{Z}|\mathbf{X},\mathbf{P}_t)$

Without going into detail (see [98]), Step 1 in Algorithm 2 corresponds to estimating the allele frequencies given the founders are known and Step 2 corresponds to estimating the founders given the allele frequencies are known.

Model with admixture

We next review STRUCTURE's model to allow for admixture. Firstly the components z_i of the founder vector $\mathbf{Z}$ are replaced by z_{il}, representing the founder of the allele individual i has at locus l. The vector $\mathbf{P}$ of founder allele frequencies remains unmodified.

With the above in mind, the a-posteriori distribution from which STRUCTURE samples model parameters is:

$$P(\mathbf{Z},Q,\mathbf{P}|\mathbf{X}) \propto P(\mathbf{Z})P(Q)P(\mathbf{P})P(\mathbf{X}|\mathbf{Z},Q,\mathbf{P}). \tag{2.9}$$

Thus:

$$P(x_{il}|\mathbf{Z},\mathbf{P},Q) = p_{z_{il}lj} \text{ and } P(z_{il}=k|\mathbf{P},Q) = q_{ik}. \tag{2.10}$$

In other words, the probability $P(x_{il}|\mathbf{Z},\mathbf{P},Q) = p_{z_{il}lj}$ of individual i having allele j at locus l given the founder vector $\mathbf{Z}$, allele frequency vector $\mathbf{P}$ and ancestry matrix Q is taken to be the frequency $p_{z_{il}lj}$ of allele j at locus l in z_{il}, and the probability $P(z_{il}=k|\mathbf{P},Q) = q_{ik}$ of locus l having originated from founder k for individual i given the vectors $\mathbf{P}$ and Q is taken to be q_{ik}.

The prior distributions for $\mathbf{P}$ and $\mathbf{Z}$ are taken to be as in the no admixture case. To obtain a prior for Q, let β denote a real number. Then we use a Dirichlet distribution $\mathcal{D}(\beta_1,\beta_2,...,\beta_K)$, where $\beta_i = \beta$ for all $1 \leq i \leq K$. Here the parameter β controls the amount of admixture we expect to see in a given dataset. For $\beta >> 1$ the model

assumes individuals to have inherited genome portions from all founders in equal proportion, while for $\beta << 1$ the model assumes individuals to have originated from a smaller number of founders. Note that for β tending to 0, we obtain the model without admixture. If no prior knowledge of β is available, STRUCTURE attempts to learn about it from the dataset under consideration. For this it assumes that β follows a uniform prior distribution on the interval $[1, 10]$, and updates β using Metropolis-Hastings.

Algorithm 3 provides as summary of the approach taken by STRUCTURE for the case of admixture in a population to generate a Markov Chain with states $\theta_t = (\mathbf{P}_t, \mathbf{Z}_t, Q_t)$ and stationary distribution $P(\mathbf{Z}, \mathbf{P}, Q|\mathbf{X})$. Again the algorithm runs for a burn-in period and states found at increments of the thinning interval are considered independent samples of the stationary distribution.

Algorithm 3 Gibbs Sampling in case of admixture

Input: a genotype matrix $\mathbf{X}$
Output: samples from $P(\mathbf{Z}, \mathbf{P}, Q|\mathbf{X})$
Initialize $\mathbf{Z}$ by drawing samples from the distribution in Equation 2.8.
1. Sample Q_t,$\mathbf{P}_t$ from $P(\mathbf{P}, Q|\mathbf{X}, \mathbf{Z}_{t-1})$
2. Sample $\mathbf{Z}_t$ from $P(\mathbf{Z}|\mathbf{X}, \mathbf{P}_t, Q_t)$
3. Update β using Metropolis-Hastings

For both cases (i. e. with and without admixture) the means of the samples produced by Markov Chain Monte Carlo are used for inference of the desired values of the sought vectors, $\mathbf{P}, \mathbf{Z}$ and Q.

Inference of the number of founders

The problem of inferring the number K of founders for a dataset $\mathbf{X}$ is generally very difficult [98]. To tackle it the following approach is proposed in [98]. From Bayes' theorem we have

$$P(K|\mathbf{X}) \propto P(\mathbf{X}|K)P(K), \tag{2.11}$$

that is, the probability of K founders given a dataset $\mathbf{X}$ is proportional to the probability of the dataset given K founders multiplied by the probability of K founders. The computation of $P(X|K)$ can however be computationally challenging. We next review an approach to overcome this problem.

Suppose we generate M samples $\mathbf{Z}_j, \mathbf{P}_j, Q_j$ $1 \leq j \leq M$ via Algorithm 3. Then

$P(\mathbf{X}|K)$ can be approximated by

$$P(\mathbf{X}|K) \approx exp(-\widehat{\mu}/2 - \widehat{\sigma}/8)$$

where $\widehat{\mu}$ is an estimate of the mean given by

$$\widehat{\mu} = \frac{1}{M}\sum_{m=1}^{M} -2logP(\mathbf{X}|\mathbf{Z}_m, \mathbf{P}_m, Q_m)$$

and $\widehat{\sigma}$ is an estimate of the standard deviation given by

$$\widehat{\sigma} = \frac{1}{M}\sum_{m=1}^{M} (-2logP(\mathbf{X}|\mathbf{Z}_m, \mathbf{P}_m, Q_m) - \widehat{\mu})^2$$

.

We remark in passing that Equation 2.11 can be seen as an ad-hoc way of estimating K [98]. An alternative is given in [31].

2.5.4 ADMIXTURE

ADMIXTURE [1] is a model-based population structure inference tool. Unlike the Bayesian approach employed by the former, ADMIXTURE relies on maximum likelihood. For this, it makes use of a block relaxation technique as well as a quasi-Newtonian convergence acceleration criterion to considerably speed up computations, as compared to STRUCTURE. We now review the computational model underpinning ADMIXTURE.

2.5.4.1 Model employed by ADMIXTURE

ADMIXTURE assumes that a set $\mathbf{X}$ of observed genotypes is sampled from K founders, each with its own allele frequencies vector F_{kl} storing the frequency of a certain type of allele (minor or major) at locus l for founder k, where $1 \leq k \leq K$. The frequency vector $\mathbf{F}$ takes the role of the vector $\mathbf{P}$ in the framework outlined in Section 2.5.2 with the only difference being its components f_{kl} are binomial distributions parametrized by the respective allele frequency at locus l for k. Also, each individual i of $\mathbf{X}$ is assumed to be an admixture of each of the K founders, with q_{ik} storing the proportion of

i's genotype that came from founder k, $1 \leq k \leq K$ (see Section 2.5.2).

The framework in Section 2.5.2 applies to ADMIXTURE as follows. ADMIXTURE assumes all loci are *bi-allelic* that is, each locus in each of the K founders has a *major* (more frequent) and a *minor* (less frequent) allele. We can denote these alleles as 2 and 1, respectively. Then f_{kl} is the frequency of the minor allele at locus l in founder k. Note that this implies that f_{kl} is a binomial distribution. Furthermore, for each locus l of an individual i, its genotype x_{il} can have one of three values, 1/1 (two minor alleles), 1/2 (a major and a minor allele) and 2/2 (two major alleles). We denote them as $x_{il} = 2, 1$ or 0 respectively.

When sampling an individual's genome the framework in Section 2.5.2 is adapted as follows. Suppose we want to sample the allele at locus l for an individual i. Then we first sample the founder K of the respective locus from a multinomial distribution parametrized by row i of Q (i. e. the ancestry proportions of individual i). We then take two independent random samples from f_{kl}. Depending on the outcome of the two samples, we set i's allele at l to be $0, 1$ or 2. Suppose now we have some estimates for the allele frequency vector $\mathbf{F} = (f_{kl})$, $Q = (q_{ik})$. Then the likelihood of each of the above genotypes (0, 1 or 2) at a locus l for individual i is given by

$$
\begin{aligned}
P(x_{il} = 2) &= (\sum_{k}^{K} q_{ik} f_{kl})^2 \\
P(x_{il} = 1) &= (\sum_{k}^{K} q_{ik} f_{kl})(\sum_{k}^{K} q_{ik}(1 - f_{kl})) \\
P(x_{il} = 0) &= (\sum_{k}^{K} q_{ik}(1 - f_{kl}))^2
\end{aligned}
$$

With the above quantities and notation in mind, the log-likelihood $\Lambda(\mathbf{X}|\mathbf{F}, Q)$ of $\mathbf{X}$ can be conveniently written as

$$
\Lambda(\mathbf{X}|\mathbf{F}, Q) = \sum_{i}^{N} \sum_{l}^{L} \{x_{il} \log \sum_{k}^{K} q_{ik} f_{kl} + (2 - x_{il}) \log \sum_{k}^{K} q_{ik}(1 - f_{kl})\} \tag{2.12}
$$

2.5.4.2 ADMIXTURE Inference algorithm

ADMIXTURE achieves estimates of a Q-matrix by trying to maximise Equation 2.12. This optimisation is frequently realised using an *Expectation-Maximisation* (EM) algo-

rithm [82, 121]. Indeed this approach is also used by other programs such as FRAPPE [115]. However the sheer number of variables which need to be estimated quickly by both of them makes such an algorithm impractical for large-scale GWAS. To speed up inference of model parameters, ADMIXTURE makes use of two computational techniques, a block relaxation algorithm and a quasi-Newtonian convergence acceleration criterion. We first review the EM approach within a population structure framework and then touch upon those techniques.

EM algorithm A natural way of finding estimates for $\mathbf{F}$ and Q is to find maximum likelihood solutions for Equation 2.12. This, however, can be computationally challenging. To overcome this problem, EM algorithms have been used. It has proven to be a useful tool for inferring model parameters (θ=(Q,F) in our case) in cases where an observed data set (genotype data in our case) is assumed to be caused by a series of K hidden variables (founders in our case).

On a high-level, EM is an iterative algorithm which alternates between an *Expectation step (E-step)* and a *Maximization step (M-step)* to provide estimates of θ.

In our case the observed data is simply an input genotype matrix $\mathbf{X}$, while the hidden variables $\mathbf{Z}$ are the founders of each locus from each individual. More precisely, the components of $\mathbf{Z}$ are z_{il}, indicating the founder of locus l of individual i. The EM algorithm can then be described as outlined in Algorithm 4. In the E-step, we compute the expectation E_{ilk} that the founder of locus l of individual i is k. We base this calculation on our current estimates for Q and $\mathbf{F}$, which in the first step are chosen at random. In the M-step, we use the values of E_{ilk} to estimate new values for q_{ik}, the contribution of founder k to individual i and the frequency f_{lgk} of allele g at locus l in founder k. We proceed in this fashion until convergence of the algorithm, (i. e. when successive values of the likelihood differ by less than a certain threshold). Denote by $\mathbf{1}(\mathbf{x} = l)$ a vector that has the same length as another vector $\mathbf{x} = (x_i)_{1 \leq i \leq l}$, whose $i-th$ element is 1, if $\mathbf{x}_i = l$ and 0 otherwise. For example $\mathbf{x}' = (2,2,3)$, we have $1(\mathbf{x}' = 2) = (1,1,0)$. Furthermore we denote by $E(x)$ the expected value of a variable. Also for Algorithm 4, let $E_{t|ilk}$ denote the value of E_{ilk} at iteration t, and allow a similar notation for all other quantities involved.

The EM algorithm as implemented in FRAPPE can be slow to converge [1]. For this reason ADMIXTURE makes use of a block relaxation and a convergence accel-

Algorithm 4 EM algorithm

Input: a genotype matrix $\mathbf{X}$

Output: estimates of $\mathbf{F}$ and Q

Initialise Q_0 and F_0 randomly

Repeat for $t = 0, 1, 2, \ldots$ (until convergence)

1. E-step: compute $E_{t|ilk} = E(\mathbf{1}(z_{il} = k)|\mathbf{X}, \theta_t = (Q_t, \mathbf{F}_t))) = \frac{f_{t|lgk} q_{t|ik}}{\sum_{k'=1}^{K} f_{t|lxk'} q_{t|ik'}}$.

2. M-step: compute $f_{t+1|lgk} = \frac{\sum_{i=1}^{N} \mathbf{1}(x=g) E_{t|ilk}}{\sum_{i=1}^{N} E_{t|ilk}} q_{t+1|ik} = \frac{1}{2L} \sum_{l=1}^{L} E_{t|ilk}$.

eration method. Block relaxation exploits properties of the likelihood given in Equation 2.12 to apply an iterative and quick to converge algorithm for optimising it. More precisely, it sequentially optimises Q for $\mathbf{F}$ fixed, and then $\mathbf{F}$ for Q fixed. Each of these optimisation steps is carried out by a method similar to Newton's method, which is a numerical method for finding the differential of a function. This iterative algorithm is further sped up using convergence acceleration. We refer the reader to [1] for a more detailed explanation.

2.5.4.3 The cross validation technique and its application by ADMIXTURE

In order to infer a dataset's number K of founders, ADMIXTURE (and also sNMF- see Section 2.5.5) use *Cross Validation* (CV) [82]. This is a popular technique in machine learning that has proven useful in tuning model hyperparameters for a dataset (e. g. that model is the one assumed by ADMIXTURE and the hyperparameter is K). We next review CV on a general level and then outline how it is used by ADMIXTURE and sNMF in order to infer K.

Suppose we are given a model (such as the one assumed by ADMIXTURE for modelling population structure) and we want to tune its set of hyperparameters (e. g. K in the case of ADMIXTURE). Then the approach taken by CV is as follows. First the dataset is split into a number $N \geq 1$ of equally sized subsets $M_i, 1 \leq i \leq N$ called *folds*. To gauge the fitness of a fixed value of θ CV then proceeds as follows. A fold, say M_i, is masked and the model with parameters θ is run on the $N-1$ folds, $M_1, M_2, \ldots, M_{i-1}, M_{i+1} \ldots M_N$. By default, ADMIXTURE assumes $N = 5$ folds. The data used by ADMIXTURE for this is the genotype matrix. Rather than representing a fold as entire individuals or loci (representing rows or columns in this matrix),

each fold is actually a collection of cells in this matrix. This is illustrated in Figure 2.18, where each color represents a fold. As can be seen, each fold is a cell in this matrix. This is in contrast to the way CV is realised in the wider field of machine learning, where each fold usually represents a collection of observations (individuals). The goodness of fit of the model on M_i (which can be measured by a number of criteria, see [82] for some of them) is then reported. This process is repeated for all $1 \leq i \leq N$, and then the reported fits are merged over all i (by e. g. taking their average). This global fit is then used to report the goodness of fit. Frequently, a number of pre-chosen values for θ are tested in this way and the ones reporting the best fit are selected as values for θ. For example, the value for K that reports the minimal deviation (see below) is chosen as a dataset's number of founders.

The above procedure can be readily applied to the problem of inferring K as follows. For each tested value of K, each one of five folds is masked in turn (i. e. its genotype entries are set to missing values). See Figure 2.18 for an example. The model is then trained on the remaining genotype data, obtaining estimates of the allele frequency vector $\mathbf{F}$ and ancestry matrix Q. The masked values are then inferred in turn, by setting a masked entry x_{il} to its expected value based on the estimated $\mathbf{F}$ and Q, or more precisely, $\widehat{x_{il}} = E[x_{il}|Q,\mathbf{F}] = 2\sum_k q_{ik} f_{kl}$.

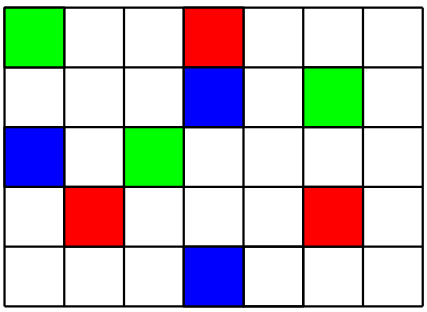


Figure 2.18: Example of CV masking. The chequered rectangle is a genotype matrix $\mathbf{X}$, where each square represents the genotype of an individual at a locus. Red, green and blue represent three different folds. These entries of $\mathbf{X}$ are set to missing values in turn and ADMIXTURE is run on the remaining genotype matrix.

The goodness of fit is then the deviation $d(x_{il}, \widehat{x_{il}})$ between the actual and the pre-

dicted genotype given by

$$d(x_{il},\widehat{x_{il}}) = x_{il}\log(x_{il}/\widehat{x_{il}}) + (2-x_{il})\log[(2-x_{il})/(2-\widehat{x_{il}})].$$

The value $d(x_{il},\widehat{x_{il}})$ is squared and averaged over all masked loci to produce the goodness of fit D_f for this particular fold f of the dataset in question i. e.

$$D_f = \frac{1}{LN}\sum_{i=1}^{N}\sum_{l=1}^{L} d(x_{il},\widehat{x_{il}})$$

We proceed in this manner for all five folds (masking a different one in each turn). The goodness of fit for K is then taken to be the average D_f value for all these folds, and the best estimate for K is the one that minimises this average.

2.5.5 sparse Non-negative Matrix Factorization (sNMF) for inferring population structure

Despite ADMIXTURE's computational efficiency, the ease with which NGS data can be generated has led to ever larger datasets on which even ADMIXTURE struggles [39]. In order to deal with such datasets, the sNMF tool was recently proposed in [39]. It boasts substantial increases in speed for large datasets. In addition it uses relaxed model assumptions (e. g. linkage equilibrium or Hardy-Weinberg equilibrium are not assumed- see Section 2.5.3) as opposed to other methods such as ADMIXTURE. In this section we provide a high-level description of the sNMF software.

Like ADMIXTURE, sNMF assumes N multi-locus individuals genotyped at L loci. Once again, each locus is assumed to originate from one of K pre-specified founders. Unlike ADMIXTURE, which assumes each locus to be a sample from a binomial distribution parametrized by the frequency of the minor (or major) allele in an ancestral gene pool, sNMF assumes each locus to be a sample from a founder-specific three-state multinomial, corresponding to 0, 1 or 2 copies of the major allele. More precisely, the probability $o_{il}(j)$ of locus l of individual i of a dataset having genotype j is given by:

$$o_{il}(j) = \sum_{k=1}^{K} q_{ik} g_{klj}, \tag{2.13}$$

where g_{klj} is the frequency of genotype j at locus l in founder k, $j = 0,1,2$, $1 \leq k \leq K$ and $1 \leq l \leq L$. The advantage of modelling population structure with genotype frequencies rather than allele frequencies is that assumptions such as Hardy-Weinberg equilibrium [113] can be dropped, thus making the technique also applicable to data sets for which it cannot be assumed.

Equation 2.13 can be written in matrix form as follows

$$\mathbf{O} = Q\mathbf{G},$$

where $\mathbf{O} = (o_{il})$ is an $N \times 3L$ matrix , Q is a genotype matrix of $\mathbf{X}$ and $\mathbf{G} = (g_{kl})$ is a $K \times 3L$ matrix. Using the framework in Section 2.5.2, an individual i's genotype can then be sampled on a locus by locus basis by letting $\mathbf{G}$ take the role of $\mathbf{P}$, and using Q as in Section 2.5.2. To be able to find $\mathbf{O}$, sNMF associates to a genotype matrix $\mathbf{X}$ an enhanced genotype matrix $\mathbf{X}^*$ as follows. To each locus l of individual i of $\mathbf{X}$ we associate three entries in $\mathbf{X}^*$, namely $\mathbf{X}^*_{il}$, $\mathbf{X}^*_{il+1}$ and $\mathbf{X}^*_{il+2}$ and put $\mathbf{X}^*_{il} = 1$ if $\mathbf{X}_{il} = 0$, $\mathbf{X}^*_{il+1} = 1$ if $\mathbf{X}_{il} = 1$ and $\mathbf{X}^*_{il+2} = 1$ if $\mathbf{X}_{il} = 2$. Furthermore, since, in general the matrices Q and $\mathbf{G}$ are not easy to find, sNMF estimates them using an iterative least-squares approach to minimise the equation

$$L(Q,G) = ||X^* - QG||_F^2, \tag{2.14}$$

subject to

$$\sum_{k=1}^{K} q_{ik} = 1, \sum_{j=0}^{2} g_{kl}(j) = 1 \tag{2.15}$$

where $q_{ik} \geq 0$ and $g_{kl} \geq 0$. For an $n \times m$ matrix $\mathbf{A} = (a_{ij})_{1 \leq i \leq n,\, 1 \leq j \leq m}$. $||A||_F = \sqrt{\sum_{i=1}^{k} \sum_{j=1}^{t} a_{ij}^2}$ is the *Frobenius norm* of A. Intuitively, Equation 2.14 measures the difference between the enhanced genotype matrix $\mathbf{X}^*$ and the predicted genotype matrix given by $Q\mathbf{G}$, with a smaller value for the Frobenius norm indicating a better fit of Q and $\mathbf{G}$ for the data.

Under the constrains in Equation 2.15, the matrices Q and $\mathbf{G}$ which minimise Equation 2.14 can be found by *Non-negative Matrix Factorisation (NMF)* [30]. While a great number of algorithms exist for this problem, sNMF makes use of *alternating non-negative least squares with an active set (ANLS-AS)* [62]. We outline this ap-

proach in Algorithm 5, where $\mathbf{e}_{1\times K}$ is a row vector having all entries set to 1, and $\underline{0}_{1\times N}$ is a row vector of length N with all entries set to zero and a is a non-negative regularisation parameter. Also, $\begin{pmatrix}\mathbf{A}\\ \mathbf{b}\end{pmatrix}$ denotes the matrix obtained by adding an extra row (i. e. vector $\mathbf{b}$) to a matrix $\mathbf{A}$. This problem is solved as in [30], and the obtained Q-matrix is then normalized so that rows sum up to one. Note that higher values of a encourage sparser solution matrices.

Algorithm 5 ANLS-AS algorithm used by sNMF

Input: a genotype matrix $\mathbf{X}$

Output: estimates of $\mathbf{G}$ and Q

Initialize Q randomly with non-negative values.

Repeat for $t = 0,1,2,...$(until improvements in Equation 2.14 amount to less than10^{-4}):

1. Find a matrix $\mathbf{G}$ such that $||\mathbf{X} - Q\mathbf{G}||_F^2$ is minimised. This can be easily done by solving a system of linear equations. To satisfy the constraints on $\mathbf{G}$, all negative entries are set to 0 and then $\mathbf{G}$ is normalized.

2. Find Q such that $\left|\left|\begin{pmatrix}\mathbf{G}^T\\ \sqrt{a}\mathbf{e}_{1\times K}\end{pmatrix}Q - \begin{pmatrix}\mathbf{X}^T\\ \underline{0}_{1\times N}\end{pmatrix}\right|\right|_F^2$ is minimised.

2.5.5.1 sNMF's approach to inferring K via Cross-Validation

Similar to ADMIXTURE, sNMF infers K by masking a portion of the genotype data and then attempts to assess sNMF's underpinning population stratification model's ability to infer the masked genotype values as a measure of fitness for K (see Equation 2.14). More precisely, a given input dataset is split into a training set and a test set, with 5% of the genotype data comprising the test set (chosen by randomly masking certain loci at certain randomly chosen individuals). sNMF is then run on the training set in order to infer estimates of Q and $\mathbf{G}$, denoted by $\hat{Q}$ and $\hat{\mathbf{G}}$ respectively. These estimates are then used to form a distribution of the values of the three possible genotypes $o_{il}^{pred}(j)$ at the masked locus l for individual i as follows

$$o_{il}^{pred}(j) = \sum_{k=1}^{K} q_{ik} g_{kl},\ j = 0,1,2$$

The goodness of fit for the value of K under consideration is then obtained by averaging the values of $-\log o_{il}^{pred}(x_{il})$ for each of the masked genotypes in the test

set. The values of K and a are chosen such that this goodness of fit is maximised.

2.5.6 PCA analysis for correcting for population structure

Contrary to the above model-based approaches, EIGENSTRAT is a non-model based approach for population structure inference. It relies on PCA, (see Chapter 5 for details of the approach). While its use of PCA ensures fast runtime, EIGENSTRAT does not return a Q-matrix which is very useful for the purposes of interpretability. However, it does return principal components (PC), which are also useful for correcting for population structure. We conclude our reviews of tools for population structure inference with a review EIGENSTRAT. We follow [95].

The input to EIGENSTRAT is an $M \times N$ matrix $\mathbf{X}^T = (x_{ji})_{1 \leq i \leq N, 1 \leq j \leq M}$, where M is the number of SNPs and N is the number of individuals, and x_{ji} is the genotype for SNP j in individual i (either 0,1 or 2). Then the row mean $\widehat{\mathbf{X}_j}$ of row j is first subtracted from each entry x_{ji} of each row j. The resulting quantity is then scaled by $p_j = (1 + \sum_{j=1}^{N} x_{ji})/(2 + 2N)$. The thus obtained matrix is denoted by $\mathbf{X}'$. Then an $N \times N$ matrix Ψ of covariance between the individuals in $\mathbf{X}'$ is computed. The k-th axis of variation is the eigenvector of Ψ with k-th highest eigenvalue. The ancestry a_{ik} of individual i along the k-th axis of variation then equals the i-th entry of the k-th eigenvector.

Using the found ancestry values a_{ik}, the adjusted genotype $x_{ji}^{adjusted}$ at a given SNP j for individual i along a given axis of variation k is calculated as:

$$x_{ji}^{adjusted} = x_{ij} - \gamma_i a_{jk},$$

where $\gamma_i = \sum_{j=1}^{N} a_{jk} x_{ij} / \sum_{j=1}^{N} a_{jk}^2$. Phenotype information f_j for individual j is adjusted similarly. The main purpose of these adjustments is to remove all correlation between ancestry (population structure) and observed genotype and phenotype. The ancestry adjusted phenotype and genotype for j are then used in the computation of a χ^2-statistic [95]. The ancestry adjusted phenotype is then used to test whether there is a significant association between a SNP i and a phenotypic trait of interest (see [95] for details).

I: Phylogenetics and NGS data

Chapter 3

Lassoing and corralling ultrametric trees

This chapter is an adaptation of

K.T. Huber. and A-A. Popescu Lassoing and corralling rooted phylogenetic trees. *Bull Math Biol.*, 75:444–465, 2013.

A-A. Popescu's contribution is writing first drafts for the proofs of the theorems and proposing the characterisation of weak lassos.

3.1 Chapter Summary

In this chapter we provide theoretical insight into the problem of uniquely determining phylogenetic trees from incomplete distance data. This problem is relevant for reconstructing phylogenetic trees from NGS data, for which distance-based reconstruction methods are popular.

3.2 Introduction

Years of selective breeding have resulted in large numbers of different varieties of, for example, oilseed rape and rice and also numerous animal breeds including dogs and

chicken. Genomewide association studies constitute a powerful tool to try and link the observed phenotypic variability between the varieties (we will collectively refer to a variety and a breed as a variety) with variations in the genomes of the varieties. A key component of such a study is phylogenetic clustering whereby one aims to construct a dendrogram for a set of individuals from within a variety of interest indicating levels of similarity between them. Using some sort of distance measure this similarity can be based on e. g. morphological traits such as grain type or Single Nucleotide Polymorphism (SNP) markers obtained through next generation sequencing technology (see e. g. [51, 81, 123] for examples of such studies).

From a formal point of view, a dendrogram can be thought of as a pair (T, ω) consisting of (i) a rooted tree T with leaf set a given non-empty finite set X (e. g. individuals), no degree two vertices except a distinguished vertex ρ_T of T called the root of T, and all other non-leaf vertices of T of degree at least three (we will refer to such a tree simply as an *X-tree*), and (ii) an edge-weighting $\omega : E(T) \to \mathbb{R}_{\geq 0}$ for T that is equidistant which means that the induced distance $D_{(T,\omega)}(\rho_T, x)$ from ρ_T to every leaf $x \in X$ of T is the same (as with all relevant concepts, we refer the reader to the next section for a precise definition). With and without the equidistance requirement such pairs (T, ω) have generated a lot of interest in the literature and so it is not surprising that numerous deep results for them are known provided the distance information from which to construct T and ω is complete in the sense that for all elements x and y in X the distance between x and y is given (see e. g. [25, 110]).

However even for data generated with modern sequencing technologies the required distance measures need not always be reliable (or may simply be missing) resulting in only partial distance information for dendrogram reconstruction. From the perspective of the aforementioned formalization of such a structure, the problem thus becomes (a) how to construct an *equidistant X-tree* (i. e. an X-tree with equidistant edge weighting) from distance information on only a subset of pairs of its leaves and, (b) if such a tree can be constructed from such a subset of its leaves, when is it lassoed (uniquely determined) by that set.

Although approaches for tackling the first problem exist in the form of, for example, an approach introduced in [19] not much is known about the second. A notable exception is a study in [26] carried out for the unrooted analogue of an equidistant X-tree. Viewing partial distance information on a set X as a set of *cords*, that is, subsets

of X of size two, the authors considered the following four natural interpretations of the above uniqueness problem. Namely, given a set $\mathcal{L} \subseteq \binom{X}{2}$ and an edge-weighted unrooted phylogenetic tree T on X

(i) when is the edge-weighting of T uniquely determined by $\mathcal{L}$?,

(ii) when is the shape i. e. the topology of T uniquely determined by $\mathcal{L}$?,

(iii) when are both the edge-weighting and the topology of T uniquely determined by $\mathcal{L}$?, and

(iv) when is the topology of T uniquely determined by $\mathcal{L}$ up to T being obtained from another X-tree by collapsing edges?.

Formalized as $\mathcal{L}$ being an edge-weight/topological/strong/weak lasso for a phylogenetic tree with leaf set X, the authors of [26] showed that all four concepts are distinct. Also, they presented results that allowed them to not only investigate the above types of lasso from a recursive point of view but also characterize under what circumstances a specifically constructed set of cords is a topological lasso (see Section 3.9 for more on this). However a characterization for the general case eluded them.

Replacing the concept of an edge-weight lasso by that of an equidistant lasso to reflect the fact that for the edge-weighted X-trees of interest here the induced distance from the root to any leaf of such a tree is the same, we show that for X-trees the situation changes. More precisely, we present for an X-tree T characterizations for when a set $\mathcal{L} \subseteq \binom{X}{2}$ is a weak lasso for T (Theorem 6), an equidistant lasso for T (Theorem 7), and for when it is a topological lasso for T (Theorem 9) in terms of the child edge graph $G_T(\mathcal{L}, v)$ that can be canonically associated to every non-leaf vertex v of T via its child edges. Our characterizations can be thought of as a spectrum on the connectedness of that graph with the extreme situations being an equidistant lasso and a topological lasso. They imply that every topological lasso and every non-empty weak lasso must be an edge-weight lasso (Corollaries 10 and 8) and that in case T is binary the notions of an equidistant lasso and a topological lasso (and thus a weak lasso) coincide. Consequently, every edge-weight/topological lasso is also a strong lasso in that case (Corollary 10). We also investigate two special types of sets of cords originally introduced in [26] in the light of our findings above. This investigation shows

in particular that it is possible for the concepts of an equidistant and a topological lasso to coincide without the X-tree they are referring to being binary.

The outline of the chapter is as follows. In the next section, we introduce relevant terminology. In Section 3.4, we present first characterizations for when a set of cords is a topological/equidistant/weak lasso for an X-tree (Theorem 2). In Section 3.5, we introduce the child-edge graph associated to a non-leaf vertex of an X-tree and present first properties of it concerning corraling sets of cords. In Section 3.6, we establish Theorem 6. In Section 3.7, we show Theorem 7 and in Section 3.8, we prove Theorem 9. In Section 3.9, we present two general ways for constructing for an X-tree T two different sets of cords of $\binom{X}{2}$ and discuss their properties in the context of lassoing and corraling T. We conclude with Section 3.10 where we also present some open problems.

3.3 Preliminaries

Denote by $V^o(T)$ the set of interior vertices of an X-tree T and by $V^o(T)^-$ the set of interior vertices of T that are not the parents of a pseudo-cherry. Let ω denote an *edge-weighting for* T. We call ω *equidistant* if

(E1) $D_{(T,\omega)}(x,\rho_T) = D_{(T,\omega)}(y,\rho_T)$, for all $x,y \in X$, and

(E2) $D_{(T,\omega)}(x,u) \geq D_{(T,\omega)}(x,v)$, for all $x \in X$ and any $u,v \in V(T)$ such that u is encountered before v on the path from ρ_T to x.

Note that Property (E2) implies that, for all interior vertices v of T and all leaves $x,y \in X$ of T for which v lies on the path from x to y, we have $D_{(T,\omega)}(x,v) = D_{(T,\omega)}(y,v)$. Also note that our definition of an equidistant edge-weighting is slightly different from the one given in [110] in so far that ω is a map into $\mathbb{R}_{\geq 0}$ and not into $\mathbb{R}$, as in [110].

Suppose $\mathcal{L} \subseteq \binom{X}{2}$ is a set of cords. Also suppose T' is a further X-tree and ω and ω' are edge-weightings for T and T', respectively. Canonically extending the corresponding concepts introduced in [26] for unrooted phylogenetic trees on X (and further studied in [56]) to X-trees, we say that $\mathcal{L}$ is

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

(ii) a *topological lasso for* T if, for every X-tree T' and any equidistant, proper edge-weightings ω of T and ω' of T', respectively, we have that T and T' are equivalent whenever (T,ω) and (T',ω') are $\mathcal{L}$-isometric.

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

(iv) a *weak lasso for* T if, for every X-tree T' and any equidistant, proper edge-weightings ω of T and ω' of T', respectively we have that T is refined by T' whenever (T,ω) and (T',ω') are $\mathcal{L}$-isometric.

Also, we say that a set $\mathcal{L} \subseteq \binom{X}{2}$ of cords is an *equidistant/topological/weak/strong lasso on* X if there exists an X-tree T such that $\mathcal{L}$ is an equidistant/topological/weak/strong lasso for T. For the convenience of the reader, we illustrate the above types of lassos in Figure 3.1 for $X = \{a,b,c,d,e\}$.

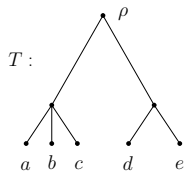


Figure 3.1: For $X = \{a,b,c,d,e\}$, the set $\mathcal{L} = \{ab,cd,de\}$ is an equidistant lasso for the depicted X-tree T, the set $\mathcal{L} = \{ab,ac,bc,bd,de\}$ is a topological lasso for T and also a strong lasso for T, and the set $\mathcal{L} = \{ab,bc,cd,de\}$ is a weak lasso for T.

Note that we will also say that a set $\mathcal{L}$ of cords of X *corrals* an X-tree T if $\mathcal{L}$ is a weak lasso for T. Also note that a topological lasso for an X-tree T is in particular a weak lasso for T, and that the notions of a topological lasso for T and a weak lasso for T coincide if T is binary. Finally note that for $\mathcal{L}$ to be a topological/equidistant lasso we must have that $\mathcal{L} \neq \emptyset$. However $\mathcal{L} \neq \emptyset$ need not hold for $\mathcal{L}$ to corral an X-tree as every subset of $\binom{X}{2}$ including the empty-set corrals the *star-tree on* X, that is the tree with a unique interior vertex and leaf set X.

3.4 A first characterization of a topological/weak/equidistant lasso

In this section we present a first characterization for when a set of cords of X is a topological/weak/equidistant lasso for an X-tree. To establish this characterization, we require further definitions and notations.

Suppose for the following that T is an X-tree and that $v \in V^o(T)$. Then we call an edge e of T incident with v that is not crossed by the path from the root ρ_T of T to v a *child edge* of v. If e is incident with v but lies on the path from ρ_T to v then we call it a *parent edge* of v. In the former case, we call the vertex incident with that edge but distinct from v a *child* of v and in the latter a *parent* of v. We call a vertex w of T distinct from v a *descendant* of v if there exists a path from v to w (possibly of length one) that crosses a child of v and 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 $L_T(v) := \{v\}$. Also if there is no ambiguity as to which X-tree T we are referring then we will write $L(v)$ rather than $L_T(v)$.

Suppose T is an X-tree. Then, for all $x \in X$, we denote the edge of T incident with x by e_x and the parent of x by v_x. Moreover, we call a non-empty subset $L \subsetneq X$ of leaves of T that all have the same parent $v \in V^o(T)$ a *pseudo-cherry* of T if $L = L(v)$. In that case, we also call v the *parent* of that pseudo-cherry. If $\{x_1, \ldots, x_k\}$, $k \geq 2$, is a pseudo-cherry of some X-tree T then we will sometimes write $x_1, \ldots, x_k$ rather than $\{x_1, \ldots, x_k\}$. Note that every X-tree on three or more leaves that is *non-degenerate*, that is, not the star-tree on X must contain at least one pseudo-cherry. Also note that in case $|L| = 2$ the definition of an pseudo-cherry reduces to that of a cherry in the usual sense (see e.g. [110]). In the special case that $|X| = 3$, say $X = \{a, b, c\}$, and that T has a cherry, a, b say, we call T a *triplet* and denote T by $ab|c$ (or, equivalently, $c|ab$).

We denote by $\mathcal{R}(T)$ the set of triplets displayed by T. As is well-known, any X-tree T can display at most $\binom{|X|}{3}$ triplets with equality holding if and only if T is binary. Furthermore, any X-tree T is uniquely determined by the set $\mathcal{R}(T)$ in the sense that if T' is a further X-tree and $\mathcal{R}(T) = \mathcal{R}(T')$ holds then T and T' must be equivalent (see e. g. [25, Chapter 9] and [110, Section 6.4]).

Observe that if T is an X-tree, ω is an equidistant, proper edge-weighting for T, and $a, a', b \in X$ are pairwise distinct elements then either all three pairwise dis-

tances induced on $Z := \{a, a', b\}$ must coincide or two of the distances induced by it must coincide and the third one must be strictly less than that distance. Moreover, $D_{(T,\omega)}(a,a') < D_{(T,\omega)}(a,b) = D_{(T,\omega)}(a',b)$ holds if and only if $aa'|b \in \mathcal{R}(T)$ and we have $D_{(T,\omega)}(a,a') = D_{(T,\omega)}(a,b) = D_{(T,\omega)}(a',b)$ if and only if the Z-tree $T|_Z$ is the star-tree on Z.

The next result is fundamental for this chapter.

Lemma 1. *Suppose that T and T' are two X-trees with equidistant, proper edge-weightings ω and ω', respectively, and let $a, a', b \in X$ denote three pairwise distinct elements such that $D_{(T,\omega)}(a,a') = D_{(T',\omega')}(a,a')$ and $D_{(T,\omega)}(a,b) = D_{(T',\omega')}(a,b)$. Then the following hold:*

(i) If $D_{(T,\omega)}(a,a') < D_{(T,\omega)}(a,b) = D_{(T,\omega)}(a',b)$ then

$$D_{(T',\omega')}(a,a') < D_{(T',\omega')}(a,b) = D_{(T',\omega')}(a',b), \tag{3.1}$$

in particular, $D_{(T',\omega')}(a',b) = D_{(T,\omega)}(a',b)$.

(ii) $aa'|b \in \mathcal{R}(T)$ if and only if $aa'|b \in \mathcal{R}(T')$.

(iii) If $D_{(T,\omega)}(a',b) = D_{(T',\omega')}(a',b)$ then $T|_Z$ is the star-tree on $Z := \{a,a',b\}$ if and only if $T'|_Z$ is the star-tree on Z.

Proof. (i): Note that $D_{(T,\omega)}(a,a') = D_{(T',\omega')}(a,a')$ combined with $D_{(T,\omega)}(a,b) = D_{(T',\omega')}(a,b)$ implies that we cannot have $D_{(T',\omega')}(a,a') = D_{(T',\omega')}(a,b)$ as this would imply that $D_{(T,\omega)}(a,a') = D_{(T,\omega)}(a,b)$, which is impossible. But then our assumptions imply that $D_{(T',\omega')}(a,a') = D_{(T',\omega')}(a',b)$ cannot hold either. Inequality (3.1) now follows from the observation preceding the statement of the lemma. In particular, this implies $D_{(T',\omega')}(a',b) = D_{(T',\omega')}(a,b) = D_{(T,\omega)}(a,b) = D_{(T,\omega)}(a',b)$.

(ii) & (iii): This is an immediate consequence of (i) and the observation preceding the statement of the lemma. □

To be able to state the next result we require a further definition. Suppose $\mathcal{L} \subseteq \binom{X}{2}$ and T is an X-tree. Let $x, y \in X$ be two distinct leaves of T that are contained in the same pseudo-cherry of T. Then we put

$$\mathcal{L}_1(x,y) := \{ab \in \mathcal{L} : x \notin \{a,b\}\} \cup \{ay : ax \in \mathcal{L}\}.$$

Note that $\mathcal{L} = \emptyset$ if and only if $\mathcal{L}_1(x,y) = \emptyset$.

Theorem 2. *Suppose that $\mathcal{L} \subseteq \binom{X}{2}$ is a set of cords and that T is an X-tree. Let $x,y \in X$ denote two distinct leaves of T that are contained in the same pseudo-cherry of T. Then the following hold:*

(i) $\mathcal{L}$ is a topological lasso for T if and only if $\mathcal{L}_1(x,y) \cup \{xy\}$ is a topological lasso for T and $xy \in \mathcal{L}$.

(ii) $\mathcal{L}$ is an equidistant lasso for T if and only if $\mathcal{L}_1(x,y) \cup \{xy\}$ is an equidistant lasso for T and $xy \in \mathcal{L}$.

(ii) If $\neq \emptyset$ then $\mathcal{L}$ is a weak lasso for T if and only if $\mathcal{L}_1(x,y) \cup \{xy\}$ is a weak lasso for T and $xy \in \mathcal{L}$.

Proof. (i): Put $\mathcal{L}_1^+ := \mathcal{L}_1(x,y) \cup \{xy\}$ and assume first that $\mathcal{L}$ is a topological lasso for T. Then $\mathcal{L} \neq \emptyset$. Let T' denote an X-tree and ω and ω' equidistant, proper edge-weightings for T and T', respectively, so that (T,ω) and (T',ω') are $\mathcal{L}_1^+$-isometric. To see that T and T' are equivalent it clearly suffices to show that (T,ω) and (T',ω') are also $\mathcal{L}$-isometric, that is,

$$D_{(T,\omega)}(a,b) = D_{(T',\omega')}(a,b) \tag{3.2}$$

holds for all $ab \in \mathcal{L}$.

Suppose $ab \in \mathcal{L}$. If $x \notin \{a,b\}$ then $ab \in \mathcal{L}_1^+$ and so Equation (3.2) holds as (T,ω) and (T',ω') are $\mathcal{L}_1^+$-isometric. So assume that $x \in \{a,b\}$, say $x = a$. If $y = b$ then $ab = xy \in \mathcal{L}$ and so Equation (3.2) holds by the same argument. If $y \neq b$ then $xb = ab \in \mathcal{L}$ and so $yb \in \mathcal{L}_1^+$. Since $xy \in \mathcal{L}_1^+$ also holds, we have $D_{(T,\omega)}(x,b) = D_{(T',\omega')}(x,b)$ by Lemma 1 and so Equation (3.2) follows in this case, too.

Conversely, suppose that $\mathcal{L}_1^+$ is a topological lasso for T and that $xy \in \mathcal{L}$, then $\mathcal{L}_1^+ \neq \emptyset$. Assume that T' is an X-tree and that ω and ω' are equidistant, proper edge-weightings for T and T', respectively, so that (T,ω) and (T',ω') are $\mathcal{L}$-isometric. To see that T and T' are equivalent it clearly suffices to show that (T,ω) and (T',ω') are also $\mathcal{L}_1^+$-isometric, that is, that

$$D_{(T,\omega)}(a,b) = D_{(T',\omega')}(a,b) \tag{3.3}$$

holds for all $ab \in \mathcal{L}_1^+$.

Suppose $ab \in \mathcal{L}_1^+$. If $x \in \{a,b\}$ then $ab = xy \in \mathcal{L}$ and so Equation (3.3) holds as (T,ω) and (T',ω') are $\mathcal{L}$-isometric. So assume that $x \notin \{a,b\}$, then $ab \in \mathcal{L}_1(x,y)$. If $y \notin \{a,b\}$ holds too then $ab \in \mathcal{L}$ and so Equation (3.3) holds by the same argument. So assume that $y \in \{a,b\}$, say, $y = a$. Then $yb = ab \in \mathcal{L}_1^+$ and so one of $yb \in \mathcal{L}$ and $xb \in \mathcal{L}$ must hold by the definition of $\mathcal{L}_1^+$. If the former holds then $ab = yb \in \mathcal{L}$ and so Equation (3.3) holds by assumption on (T,ω) and (T',ω'). If $xb \in \mathcal{L}$ then $D_{(T,\omega)}(y,b) = D_{(T',\omega')}(y,b)$ follows by Lemma 1 since, by assumption, $xy \in \mathcal{L}$. But then Equation (3.3) holds in this case, too.

(ii) & (iii): These follow using similar arguments as in the proof of (i). □

3.5 The child-edge graph

In this section we first introduce the child-edge graph $G_T(\mathcal{L},v)$ associated with an interior vertex v of an X-tree T and a non-empty set $\mathcal{L} \subseteq \binom{X}{2}$ of cords and then study some of its properties with regards to corraling an X-tree. We start with a definition.

Suppose T is an X-tree, $v \in V^o(T)$, and $\mathcal{L} \subseteq \binom{X}{2}$ is a non-empty set of cords. Then we call the graph $G_T(\mathcal{L},v) = (V_{T,v}, E_{T,v})$, with vertex set $V_{T,v}$ the set of all child edges of v and edge set $E_{T,v}$ the set of all $\{e,e'\} \in \binom{V_{T,v}}{2}$ for which there exist leaves $a,b \in X$ such that e and e' are edges on the path from a to b in T and $ab \in \mathcal{L}$, the *child-edge graph (of v with respect to T and $\mathcal{L}$)*. Note that in case there is no danger of ambiguity with regards to the X-tree T we are referring to, we will write $G(\mathcal{L},v)$ rather than $G_T(\mathcal{L},v)$ and V_v and E_v rather than $V_{T,v}$ and $E_{T,v}$.

For T an X-tree, the next result provides a key insight into the structure of $G(\mathcal{L},v)$, $v \in V^o(T)$, whenever $\mathcal{L}$ corrals T. To state it, we denote by $V^o(T)^-$ the set of all interior vertices of T that are not a parent of a pseudo-cherry of T.

Lemma 3. *Suppose that T is a non-degenerate X-tree, that $\mathcal{L} \subseteq \binom{X}{2}$ is a non-empty set of cords that corrals T, and that $v \in V^o(T)$. Then the following hold:*

(i) If $v \in V^o(T)^-$ and $e \in V_v$ is a child edge of v that is not incident with a leaf of T then $\{e,e'\} \in E_v$, for all $e' \in V_v$ that are incident with a leaf of T. In particular, $G(\mathcal{L},v)$ is connected.

(ii) If $v \in V^o(T) - V^o(T)^-$ then $G(\mathcal{L}, v)$ is connected.

Proof. (i): Let $e \in V_v$ denote a child edge of v that is not incident with a leaf of T and let $u \in V(T)$ denote the child of v that is incident with e. Let $\omega : E(T) \to \mathbb{R}_{\geq 0}$ be an equidistant, proper edge-weighting for T. Assume for contradiction that there exists a child $u' \in V(T)$ of v that is a leaf such that with $e_{u'}$ denoting the child edge of v incident with u', we have that $\{e, e_{u'}\} \not\subseteq E_v$. Note that since ω is equidistant and u' is a leaf of T whereas u is not, we must have $\omega(e_{u'}) > \omega(e)$.

Assume first that $|V_v| \geq 3$. Let T' denote the X-tree obtained from T by deleting the edge $e_{u'}$ and adding the edge $e^* = \{u, u'\}$. Clearly, T' is not a refinement of T. Consider the edge-weighting

$$\omega' : E(T') \to \mathbb{R}_{\geq 0} : f \mapsto \begin{cases} \omega(f) & \text{if } f \neq e^*, \\ \omega(e_{u'}) - \omega(e) & \text{else.} \end{cases}$$

Then it is easy to see that ω' is equidistant and proper. Since, by construction, (T, ω) and (T', ω') must be $\mathcal{L}$-isometric it follows that T' is a refinement of T as $\mathcal{L}$ corrals T; a contradiction.

Now assume that $|V_v| = 2$. If $v \neq \rho_T$ then v must have a parent $w \in V(T)$. Let T' denote the X-tree obtained from T as before except that we now suppress v as this has rendered it a vertex with a single child. Let ω' be the edge-weighting for T' as defined above except that we put $\omega'(\{u, w\}) = \omega(\{u, v\}) + \omega(\{v, w\})$. Then the same arguments as in the previous case yield a contradiction

If $v = \rho_T$ then let T' denote the X-tree obtained from T by collapsing the edge $\{v, u\}$. Clearly, T' is not a refinement of T. Consider the edge-weighting $\omega' : E(T') \to \mathbb{R}_{\geq 0}$ for T' defined by putting $\omega' = \omega|_{E(T')}$. Then the same arguments as in the previous two cases yield a contradiction.

(ii): Assume for contradiction that there exists some $v \in V^o(T) - V^o(T)^-$ such that $G(\mathcal{L}, v)$ is not connected. Then every child of v is a leaf of T and there exist vertices $e_1, e_2 \in V_v$ distinct, such that e_1 and e_2 are not joined by a path in $G(\mathcal{L}, v)$. Let G_1 and G_2 denote the connected components of $G(\mathcal{L}, v)$ containing e_1 and e_2, respectively. For all children $u \in V(T)$ of v, let e_u denote the child edge of v incident with u. Note that since T is non-degenerate there must exist a vertex $w \in V(T)$ that is the parent of v. Let T' denote the X-tree obtained from T via the following process. Let $i = 1, 2$. Then,

for all $u_i \in V(G_i)$, subdivide the edge e_{u_i} by a new vertex p_{u_i} not already contained in $V(T)$. Next, identify all vertices p_{u_i} into the vertex p_i and then delete all copies of the edges $\{v, p_i\}$ from T. Finally, add the edge $\{w, p_i\}$ and suppress p_i if $|V(G_i)| = 1$. If $V_v = V(G_1) \cup V(G_2)$ then also suppress the vertex v. The resulting tree is T' and, in either case, T' is clearly not a refinement of T.

Let ω denote an equidistant, proper edge-weighting for T. Note that $\omega(e_1) = \omega(e)$ must hold for all $e \in V_v$, as $v \in V^o(T) - V^o(T)^-$. For the following, assume first that neither p_1 nor p_2 have been suppressed in the construction of T'. Consider the edge-weighting

$$\omega' : E(T') \to \mathbb{R}_{\geq 0} : e \mapsto \begin{cases} \omega(e) & \text{if } p_1, p_2 \notin e, \\ \omega(\{v, w\}) & \text{if } e \in \{\{w, p_1\}, \{w, p_2\}\} \\ \omega(e_1) & \text{else.} \end{cases}$$

Then, by construction, ω' is equidistant and proper and (T, ω) and (T', ω') are $\mathcal{L}$-isometric. Since, by assumption, $\mathcal{L}$ corrals T it follows that T' is a refinement of T; a contradiction.

In case one of p_1 and p_2 or both of them have been suppressed in the construction of T' the definition of the edge-weighting ω' for T' is similar to the one above thus leading to a contradiction in these cases too. □

The next result is a strengthening of Lemma 3(i). To state it, we require further terminology concerning child-edge graphs. Suppose T is an X-tree, $v \in V^o(T)^-$, and $\mathcal{L} \subseteq \binom{X}{2}$ is a non-empty set of cords. Then we denote by $E_l(v) \subseteq E(T)$ the set of child edges of v that are incident with a leaf of T and by $E_s(v) \subseteq E(T)$ the set of child edges of v that are not contained in $E_l(v)$. Note that $E_s(v)$ is empty if and only if v is the parent of a pseudo-cherry of T. Also note that $E_l(v) = \emptyset$ might hold. Clearly, if neither of them is the empty-set then $\{E_l(v), E_s(v)\}$ is a partition of V_v. For $v \in V^o(T)^-$, we say that $G(\mathcal{L}, v)$ is *rich* if the subgraph $G_T(\mathcal{L}, v)_s$ of $G_T(\mathcal{L}, v)$ induced by $E_s(v)$ is a clique, and, in case $E_l(v) \neq \emptyset$, we have for all $e \in E_l(v)$ and all $e' \in E_s(v)$ that $\{e, e'\} \in E_v$. As before we will write $G(\mathcal{L}, v)_s$ rather than $G_T(\mathcal{L}, v)_s$ if there is no ambiguity with regards to which X-tee T we are referring to. Note that with T and v as above, if $E_l(v) = \emptyset$ then $G(\mathcal{L}, v)$ is rich if and only if $G(\mathcal{L}, v)_s$ is a clique.

To illustrate these concepts consider the set of cords

$$\mathcal{L} = \{ac, ae, ag, bd, be, bh, ce, cg, eh, cd, ef, gh, ai\}$$

on $X = \{a,b,c,d,e,f,g,h,i\}$ and the X-tree T depicted in Figure $2(i)$. For v as indicated in Figure $2(i)$ we depict the child edge graph $G(\mathcal{L}, v)$ in Figure $2(ii)$ which is clearly rich.

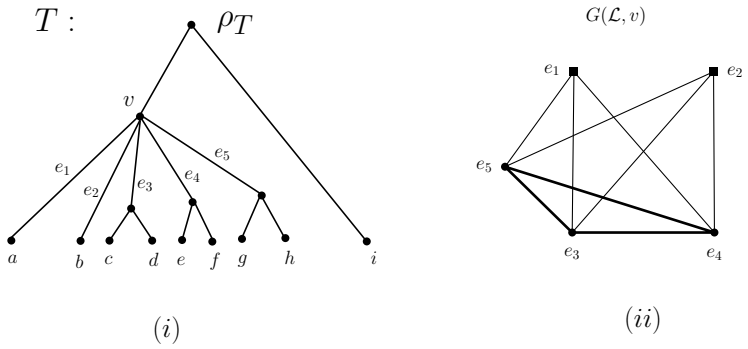


Figure 3.2: (i) The X-tree T for $X = \{a,b,c,d,e,f,g,h,i\}$. (ii) The child-edge graph $G(\mathcal{L}, v)$ for $\mathcal{L}$ as indicated in the text and T and v as in (i). For ease of readability, the vertices in $E_l(v)$ are marked by a square and those in $E_s(v)$ by a dot. The edges of the graph $G(\mathcal{L}, v)_s$ are represented by thick lines.

Proposition 4. *Suppose that T is a non-degenerate X-tree and that $\mathcal{L} \subseteq \binom{X}{2}$ is a non-empty set of cords. If $\mathcal{L}$ corrals T then $G(\mathcal{L}, v)$ must be rich, for all vertices $v \in V^o(T)^-$.*

Proof. Let ω denote an equidistant, proper edge-weighting for T and assume for contradiction that there exists a vertex $v \in V^o(T)^-$ such that $G(\mathcal{L}, v)$ is not rich. We first show that $G(\mathcal{L}, v)_s$ must be a clique. Suppose $G(\mathcal{L}, v)_s$ is not a clique, that is, there exist distinct child edges e and e' of v contained in $V(G(\mathcal{L}, v)_s)$ such that with v_e and $v_{e'}$ denoting the children of v incident with e and e', respectively, we have, for all $a \in L(v_e)$ and all $b \in L(v_{e'})$ that $ab \notin \mathcal{L}$. Note that $|V_v| \geq 3$ must hold. Indeed, if $|V_v| = 2$ then $V_v = \{e, e'\}$ and so $\{e, e'\} \in E_v$ since, by Lemma 3, $G(\mathcal{L}, v)$ is connected. But then $G(\mathcal{L}, v)$ is a clique and so $G(\mathcal{L}, v)_s$ is a clique; a contradiction. Without loss of generality assume that $\omega(e) \geq \omega(e')$.

If $\omega(e) > \omega(e')$ then consider the X-tree T' obtained from T by deleting the edge $\{v, v_e\}$ and attaching v_e to $v_{e'}$ via the edge $e^* = \{v_e, v_{e'}\}$. Clearly, T' is not a refinement of T. Consider the edge-weighting ω' for T' defined by putting

$$\omega' : E(T') \to \mathbb{R}_{\geq 0} : \ f \mapsto \begin{cases} \omega(f) & \text{if } f \neq e^*, \\ \omega(e) - \omega(e') & \text{otherwise.} \end{cases}$$

Clearly, ω' is equidistant and proper and, by construction, (T, ω) and (T', ω') are $\mathcal{L}$-isometric. Since $\mathcal{L}$ corrals T it follows that T' must be a refinement; a contradiction.

If $\omega(e) = \omega(e')$ then consider the X-tree T' obtained from T by first identifying the vertices v_e and $v_{e'}$ (keeping the label v_e) and then deleting one of the edges from v to v_e. Again, T' is clearly not a refinement of T. Consider the edge-weighting $\omega' : E(T') \to \mathbb{R}_{\geq 0}$ defined as $\omega' = \omega|_{E(T')}$. Then the same arguments as before imply that T' is a refinement of T; a contradiction. Thus, $G(\mathcal{L}, v)_s$ is a clique, as required.

Now if $E_l(v) = \emptyset$ then $G(\mathcal{L}, v)$ must be rich. So assume that $E_l(v) \neq \emptyset$. But then Lemma 3(i), implies that $G(\mathcal{L}, v)$ must be rich. □

We conclude this section with a result that will be useful for establishing the aforementioned characterization of weak lassos in terms of child-edge graphs (Theorem 6). Its proof relies on the well-known fact that an X-tree T' is a refinement of an X-tree T if and only if $\mathcal{R}(T) \subseteq \mathcal{R}(T')$ (see e.g. [110, Theorem 6.4.1]).

Lemma 5. *Suppose that T is an X-tree that has a unique cherry x, y and that $\subseteq \binom{X}{2}$ is a set of cords that contains the set $\{xy\} \cup \{az : z \in X - \{x, y\}$ and $a = x$ or $a = y\}$, then $\mathcal{L}$ corrals T.*

Proof. By Theorem 2, it suffices to show that $\mathcal{L}' := \mathcal{L}(x, y) \cup \{xy\}$ corrals T. Suppose there exists an X-tree T' and equidistant, proper edge-weightings ω and ω' of T and T', respectively, such that (T, ω) and (T', ω') are $\mathcal{L}'$-isometric. To see that T' is a refinement of T it suffices to show that $xy|z \in \mathcal{R}(T')$ holds for all $z \in X - \{x, y\}$. Let $z \in X - \{x, y\}$. Then $zy \in \mathcal{L}'$. Combined with the facts that $xy \in \mathcal{L}'$ and that $xy|z \in \mathcal{R}(T)$ it follows, by Lemma 1, that $xy|z \in \mathcal{R}(T')$, as required. □

3.6 A characterization of a weak lasso

In this section, we characterize sets of cords of X that corral an X-tree T in terms of two properties on the child edge-graphs associated with the interior vertex of T. In addition, we present a simple example that illustrates that this characterization does not hold if the equidistance requirement for the two edge-weightings mentioned in the definition of such a lasso is dropped.

Theorem 6. *Suppose that T is a non-degenerate X-tree and that $\mathcal{L} \subseteq \binom{X}{2}$ is a non-empty set of cords. Then $\mathcal{L}$ is a weak lasso for T if and only if the following two properties hold*

(C1) $G(\mathcal{L}, v)$ *is rich, for all* $v \in V^o(T)^-$

(C2) $G(\mathcal{L}, v)$ *is connected, for all* $v \in V^o(T) - V^o(T)^-$.

Proof. Assume first that $\mathcal{L}$ corrals T. Then Properties (C1) and (C2) must hold by Proposition 4 and Lemma 3(ii).

To see the converse, assume that Properties (C1) and (C2) hold. We prove that $\mathcal{L}$ must be a weak lasso for T by induction on the size n of X. Note that the statement clearly holds in case $n = 3$ as then T is binary and a refinement of a binary X-tree is the tree itself.

Assume that the statement holds for all finite sets of size $n \geq 3$ and let X denote a set of size $n+1$. Let T be a non-degenerate X-tree and let $\mathcal{L} \subseteq \binom{X}{2}$ denote a non-empty set of cords such that properties (C1) and (C2) are satisfied for T and $\mathcal{L}$. Note that T must contain at least one pseudo-cherry. To see that $\mathcal{L}$ corrals T, let T' denote an X-tree and ω and ω' equidistant, proper edge-weightings for T and T', respectively, such that (T, ω) and (T', ω') are $\mathcal{L}$-isometric. We distinguish between the cases that (i) every pseudo-cherry of T is in fact a cherry of T and (ii) that T contains a pseudo-cherry that has at least three leaves.

Case (i): Assume that every pseudo-cherry of T is a cherry and let $x, y \in X$ such that x, y is a cherry of T. Note that since $n \geq 4$ and T is non-degenerate, there must exist a vertex $w \in V(T)$ that is the parent of v_x (which is itself the parent of the cherry x, y). Put $X_1 = X - \{x\}$ and $\mathcal{L}_1 = \mathcal{L}_1(x, y)$. Note that since $G(\mathcal{L}, v_x)$ is connected, it immediately follows that $xy \in \mathcal{L}$ and that since $G(\mathcal{L}, w)$ is rich we must have $\mathcal{L}_1 \neq \emptyset$. Let T_1 denote

the X_1-tree obtained from T by deleting x and its incident edge and suppressing v_x as this has rendered it a vertex with a single child. Let

$$\omega_1 : E(T_1) \to \mathbb{R}_{\geq 0} : e \mapsto \begin{cases} \omega(e) & \text{if } e \neq \{w,y\} \\ \omega(\{w,v_x\}) + \omega(\{v_x,y\}) & \text{else.} \end{cases}$$

Clearly, ω_1 is equidistant and proper and T_1 is either the star-tree on X_1 or not. Assume first that T_1 is the star-tree on X_1. Then x,y is the unique cherry of T and all elements $z \in X_1 - \{y\}$ are adjacent with the root ρ_T of T which is w. But then Properties (C1) and (C2) combined with Lemma 5 imply that $\mathcal{L}$ corrals T.

So assume that T_1 is non-degenerate. We claim that Properties (C1) and (C2) hold for T_1 and $\mathcal{L}_1$. We start with establishing Property (C1). Assume for contradiction that there exists some $u \in V^o(T_1)^-$ such that $G_{T_1}(\mathcal{L}_1,u)$ is not rich. We first show that $G_{T_1}(\mathcal{L}_1,u)_s$ must be a clique. Assume for contradiction that $G_{T_1}(\mathcal{L}_1,u)_s$ is not a clique, then $|E_s(u)| \geq 2$ and so for all $a \in L_{T_1}(u_e)$ and all $b \in L_{T_1}(u_{e'})$ we have that $ab \notin \mathcal{L}_1$ where u_e and $u_{e'}$ denote the children of u in T_1 incident with e and e', respectively. Note that $V^o(T_1) \cup \{v_x\} = V^o(T)$ and that $u = w$ must hold. Indeed assume for contradiction that $u \neq w$. Then v_x is not a child of u in T and so u_e and $u_{e'}$ are also children of u in T. Since $G_T(\mathcal{L},u)_s$ is a clique by Property (C1), there must exist $a \in L_T(u_e)$ and $b \in L_T(u_{e'})$ such that $ab \in \mathcal{L}$. If $x \notin \{a,b\}$ then, by the definition of $\mathcal{L}_1$, we have $ab \in \mathcal{L}_1$; a contradiction. Thus, $x \in \{a,b\}$. Without loss of generality assume that $x = a$. Then $y \in L_T(u_e)$ and, again by the definition of $\mathcal{L}_1$, we obtain $yb \in \mathcal{L}_1$; a contradiction. Thus $u = w$, as required. But then $y \in \{u_e, u_{e'}\}$ and so one of e and e' is not a vertex in $G_{T_1}(\mathcal{L}_1,u)_s$; a contradiction. Thus, $G_{T_1}(\mathcal{L}_1,u)_s$ must be a clique, as required.

Since, by assumption, $G_{T_1}(\mathcal{L}_1,u)$ is not rich, there must therefore exist a leaf z of T_1 with $e' = \{u,z\} \in E(T_1)$ holding and some vertex e in $G_{T_1}(\mathcal{L}_1,u)_s$ such that $\{e,e'\}$ is not an edge in $G_{T_1}(\mathcal{L}_1,u)$. Let u_e denote the child of u in T_1 incident with e. If $u \neq w$ then since the children of u in T are precisely the children of u in T_1 and, by Property (C1), $G_T(\mathcal{L},u)$ is rich we obtain a contradiction. Thus, $u = w$. But then $y = z$ must hold. Since $G_T(\mathcal{L},w)$ is rich there must exist some $a \in \{x,y\}$ and some $b \in L(u_e)$ such that $ab \in \mathcal{L}$. But then $yb \in \mathcal{L}_1$ and so $\{e,e'\}$ is an edge in $G_{T_1}(\mathcal{L}_1,u)$, a contradiction.

We next establish that Property (C2) is satisfied by T_1 and $\mathcal{L}_1$ which will conclude the proof of the claim. Let $u \in V^o(T_1) - V^o(T_1)^-$, then u must be the parent of a pseudo-cherry of T_1. If $u \neq w$ then since, by assumption, every pseudo-cherry of T is a cherry of T it follows that u is the parent of a cherry of T_1. But then $G_{T_1}(\mathcal{L}_1, u)$ is connected as $G_{T_1}(\mathcal{L}_1, u) = G_T(\mathcal{L}, u)$ and Property (C2) is satisfied by T and $\mathcal{L}$. So assume that $u = w$. Then u is the parent of v_x in T and all children of u in T but v_x are leaves of T. Since, by Property (C1), $G_T(\mathcal{L}, u)$ is rich, there exists for all children z of u that are leaves of T some $b_z \in \{x, y\}$ such that $b_z z \in \mathcal{L}$. But then $yz \in \mathcal{L}_1$ for all such children z of u and thus $G_{T_1}(\mathcal{L}_1, u)$ is connected, as required. Thus Property (C2) is also satisfied by T_1 and $\mathcal{L}_1$ which completes the proof of the claim. By induction, it follows that $\mathcal{L}_1$ is a weak lasso for T_1.

Let T_1' denote the X_1-tree obtained from T' by deleting x and its incident edge and suppressing the parent vertex of x in T_1' if this has rendered it a vertex with a single child. Let ω_1' denote the edge-weighting of T_1' that is canonically induced by ω_1 on the edges of T_1'. Then, by Lemma 1, combined with the assumption that (T, ω) and (T', ω') are $\mathcal{L}$-isometric, it follows that (T_1, ω_1) and (T_1', ω_1') are $\mathcal{L}_1$-isometric. Since $\mathcal{L}_1$ is a weak lasso for T_1 this implies that T_1' is a refinement for T_1.

To establish that $\mathcal{L}$ corrals T it now suffices to show that $xy|a \in \mathcal{R}(T')$ holds for all $a \in L(w) - \{x, y\}$. To this end, note that since Property (C1) is satisfied by T and $\mathcal{L}$, we have for all children $a \in L(w)$ that are leaves of T that there exists some $b \in L_T(v_x) = \{x, y\}$ such that $ab \in \mathcal{L}$. Combined with Lemma 1 and $xy \in \mathcal{L}$, it follows that $D_{(T,\omega)}(x, a) = D_{(T',\omega')}(x, a) = D_{(T,\omega)}(y, a) = D_{(T',\omega')}(y, a)$. Since $xy|a \in \mathcal{R}(T)$ we obtain $xy|a \in \mathcal{R}(T')$, as required. This completes the proof of the induction step in this case.

Case (ii): Assume that T contains a pseudo-cherry $\mathfrak{c}$ of size three or more. Let $v \in V^o(T)$ denote the parent of $\mathfrak{c}$ in T. Then, by Property (C2), $G(\mathcal{L}, v)$ is connected. Since in any connected graph there exists a vertex whose removal (plus incident edges) leaves the graph connected (see [23, Proposition 1.4.1] where a more general result is established), it follows that we may choose some $x \in X$ such that the graph $G^-(\mathcal{L}, v)$ obtained from $G(\mathcal{L}, v)$ by deleting e_x and its incident edges is connected. Note that since x is a leaf in $\mathfrak{c}$, we have $v = v_x$.

Put $X_1 := X - \{x\}$, choose some $y \in L(v_x)$ such that $xy \in \mathcal{L}$, and put $\mathcal{L}_1 := \mathcal{L}(x, y)$. Consider the X-tree T_1 obtained from T by deleting x and its incident edge. We claim

again that $G_{T_1}(\mathcal{L}_1, u)$ satisfies Properties (C1) and (C2) for all $u \in V^o(T_1)$ as specified in those conditions. To see this, note first that since v_x has at least two children in T_1, we have $V^o(T_1) = V^o(T)$ and so also $V^o(T_1)^- = V^o(T)^-$.

We start with establishing Property (C1). Assume for contradiction that there exists some vertex $u \in V^o(T_1)^-$ such that $G_{T_1}(\mathcal{L}_1, u)$ is not rich. We again show first that $G_{T_1}(\mathcal{L}_1, u)_s$ is a clique. Assume for contradiction that this is not the case and let e, e', u_e, and $u_{e'}$ be as in the corresponding situation in the previous case. Note that since $u \in V^o(T_1)^- = V^o(T)^-$ and $v_x \in V^o(T) - V^o(T)^-$ we have $v_x \neq u$. But then similar arguments as the ones used to show in Case (i) that $w \neq u$ in the context of establishing that $G_{T_1}(\mathcal{L}_1, u)_s$ is a clique yield a contradiction. Thus, $G_{T_1}(\mathcal{L}_1, u)_s$ must be a clique, as required. As in the previous case, there must therefore exist some $z \in X_1$ such that $e = \{z, u\} \in E(T_1)$ and some vertex e' in $G_{T_1}(\mathcal{L}_1, u)_s$ such that $\{e, e'\}$ is not an edge in $G_{T_1}(\mathcal{L}_1, u)$. Note that the same arguments as above imply that $u \neq v_x$. By the definition of $\mathcal{L}_1$, it follows that x must be the unique leaf in $L_T(u_{e'})$ such that $xz \in \mathcal{L}$. Since y and x are leaves in the same pseudo-cherry of T, we obtain $yz \in \mathcal{L}_1$. Consequently, $\{e, e'\}$ is an edge in $G_{T_1}(\mathcal{L}_1, u)$, a contradiction. Thus, Property (C1) is satisfied by T_1 and $\mathcal{L}_1$.

To see that T_1 and $\mathcal{L}_1$ satisfy Property (C2) assume, without loss of generality, that there exists some $u \in V^o(T) - V^o(T)^-$ such that $G_{T_1}(\mathcal{L}_1, u)$ is not connected. Then $u \neq v_x$, by the choice of x. Since u is the parent of a pseudo-cherry in T it follows that u must be the parent of the same pseudo-cherry in T_1. But then $G_{T_1}(\mathcal{L}_1, u) = G_T(\mathcal{L}, u)$ and so $G_{T_1}(\mathcal{L}_1, u)$ must be connected as, by Property (C2), $G_T(\mathcal{L}, u)$ is connected; a contradiction. This concludes the proof of the claim. By induction, it follows that $\mathcal{L}_1$ corrals T_1.

Let T_1', w, and ω_1' be as in the previous case. Then, as in that case, T_1' must be a refinement of T_1. We claim that for all children u of w in T distinct from v_x and all $a \in L_T(u)$ we must have

$$xy|a \in \mathcal{R}(T').$$

To see this, note first that Property (C1) implies for all such children u of w that there must exist some $a_u \in L_T(u)$ and some $z_{v_x} \in L_T(v_x)$ such that $a_u z_{v_x} \in \mathcal{L}$. Let u denote a child of w in T distinct from v_x and put $a = a_u$ and $z = z_{v_x}$. We show first that

$$D_{(T,\omega)}(a, y) = D_{(T',\omega')}(a, y) \text{ and } D_{(T,\omega)}(a, x) = D_{(T',\omega')}(a, x). \tag{3.4}$$

Clearly, if $z \in \{x,y\}$ then one of the two equations in (3.4) must hold. Assume without loss of generality that $z \neq x$. Since $G_T(\mathcal{L}, v_x)$ is connected by Property (C2), it follows that there exists a path $z = z_1, z_2, \ldots, z_k = x$, $k \geq 2$, from z to x in $G_T(\mathcal{L}, v_x)$. But then Lemma 1(i) implies $D_{(T,\omega)}(a,x) = D_{(T',\omega')}(a,x)$ since, for all $1 \leq i \leq k-1$, we have $z_i z_{i+1} \in \mathcal{L}$. If $z \neq y$ then similar arguments imply that $D_{(T,\omega)}(a,y) = D_{(T',\omega')}(a,y)$. If $z = y$ then $ya = za \in \mathcal{L}$ and so $D_{(T,\omega)}(a,y) = D_{(T',\omega')}(a,y)$. Thus both equations in (3.4) must hold, as required. Combined with $xy \in \mathcal{L}$ (which holds by the choice of y) and the fact that $xy|a \in \mathcal{R}(T)$ we obtain

$$xy|a \in \mathcal{R}(T')$$

in view of Lemma 1(ii). Thus the claim follows if $|L_T(u)| = 1$. So assume that $|L_T(u)| \geq 2$. Suppose $a' \in L_T(u) - \{a\}$. Then $y|a'a \in \mathcal{R}(T_1')$ must hold as T_1' is a refinement of T_1 and $y|a'a \in \mathcal{R}(T_1)$. Since the only $\{x,y,a',a\}$-tree that can simultaneously display the triplets $y|a'a$ and $xy|a$ is the tree with cherries x,y and a',a and that tree is equivalent with the tree $T|_{\{x,y,a',a\}}$ it follows that $xy|a' \in \mathcal{R}(T')$, as claimed.

Combined with the fact that T_1' is a refinement of T_1 it follows that T' is a refinement of T. Hence, $\mathcal{L}$ corrals T which concludes the proof of the induction step in this case too and, thus, the proof of the theorem.

□

Note that Theorem 6 immediately implies that for a non-empty set $\mathcal{L}$ of cords of X to be a weak lasso for a non-degenerate X-tree T, it must be a *covering* of X, that is, $X = \bigcup_{A \in \mathcal{L}} A$. Also note that Theorem 6 immediately implies that a minimum size weak lasso for T must have

$$\sum_{v \in V^o(T)^-} \left(\binom{|V(G(\mathcal{L},v)_s)|}{2} + |V(G(\mathcal{L},v)_s)| \times |V(G(\mathcal{L},v)_l)| \right) + \sum_{v \in V^o(T) - V^o(T)^-} |V_v|$$

cords. Thus, such a lasso has at most $|V^o(T)^-|\binom{m}{2} + (m-1)|V^o(T) - V^o(T)^-|$ cords where $m = \max_{v \in V^o(T)^-} |V_v|$ and at least $(l-1)|V^o(T)|$ cords where $l = \min_{v \in V^o(T)^-} |V_v|$. Note that these bounds are sharp in the case that all interior vertices of T have the same number k of children. In the former case T is such that no interior vertex of T that is not a parent of a pseudo-cherry is adjacent with a leaf of T. In the latter case T is the *bearded caterpillar tree* on X, that is, T is a (rooted) path and every vertex of that path

is adjacent with $k-1$ leaves except for the end vertex of T that is not ρ_T which has k children.

Finally, note that as the example presented in Figure 3.3 illustrates, for Theorem 9 to hold the requirement that both proper edge-weightings are equidistant in the definition of a weak lasso cannot be dropped. More precisely for $X = \{a,b,c,d,e,f\}$ and $\mathcal{L} = \{ab, bc, bd, af, ae\}$ the X-tree T pictured on the left of that figure (with the indicated edge-weighting ω ignored for the moment) satisfies Properties (C1) and (C2) and the X-tree T' depicted on the right of that figure (again with the indicated edge-weighting ω' ignored where $0 < \varepsilon < 1$) is clearly not a refinement of T. Also ω and ω' are obviously proper and, in the case of ω, equidistant and (T,ω) and (T',ω') are $\mathcal{L}$-isometric.

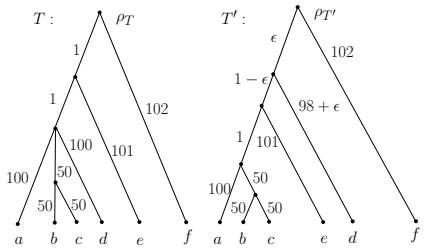


Figure 3.3: An example illustrating that for Theorem 6 to hold the equidistance requirement in the definition of a weak lasso cannot be dropped (see text for details).

3.7 A characterization of an equidistant lasso

In this section, we present a characterization of an equidistant lasso $\mathcal{L} \subseteq \binom{X}{2}$ for an X-tree T in terms of the child-edge graphs associated to the interior vertices of T. To establish it, we require a further notation. Suppose T is an X-tree and v and w are two vertices of T. Then we denote by $E_T(v,w)$ the set of all edges of T on the path from v to w.

Theorem 7. *Suppose T is an X-tree and $\mathcal{L} \subseteq \binom{X}{2}$ is a non-empty set of cords. Then the following are equivalent:*

(i) $\mathcal{L}$ is an equidistant lasso for T.

(ii) for every vertex $v \in V^o(T)$, the graph $G(\mathcal{L}, v)$ contains at least one edge.

Proof. (i) $\Rightarrow$ (ii): Suppose $\mathcal{L}$ is an equidistant lasso for T and assume for contradiction that there exists an interior vertex v of T for which $G(\mathcal{L}, v)$ does not contain an edge. Note first that $v = \rho_T$ cannot hold. Indeed, suppose ω_1 is a equidistant, proper edge-weighting for T and let $\varepsilon > 0$ be a sufficiently small real number. Consider the edge-weighting $\omega_2 : E(T) \to \mathbb{R}_{\geq 0}$ defined by putting, for all $e \in E(T)$, $\omega_2(e) = \omega_1(e) - \varepsilon$ if $\rho_T \in e$ and $\omega_2(e) = \omega_1(e)$ else. Clearly, ω_2 is an equidistant, proper edge-weighting for T distinct from ω_1 and (T, ω_1) and (T, ω_2) are $\mathcal{L}$-isometric. Since, by assumption, $\mathcal{L}$ is an equidistant lasso for T it follows that $\omega_1 = \omega_2$; a contradiction. Thus, $v \neq \rho_T$ and so there must exist a parent $w \in V(T)$ of v in T. Let ω_1 denote again an equidistant, proper edge-weighting for T. Consider the map $\omega_2 : E(T) \to \mathbb{R}_{\geq 0}$ defined by putting

$$\omega_2(e) = \begin{cases} \omega_1(e) & \text{if } v \notin e, \\ \omega_1(e) - \varepsilon & \text{if } e = \{v, w\}, \\ \omega_1(e) + \varepsilon & \text{else,} \end{cases}$$

where $\varepsilon > 0$ is small enough. Clearly, ω_2 is an equidistant, proper edge-weighting for T which is distinct from ω_1. By construction, (T, ω_1) and (T, ω_2) are $\mathcal{L}$-isometric and so $\omega_1 = \omega_2$ must hold as $\mathcal{L}$ is an equidistant lasso for T; a contradiction.

(ii) $\Rightarrow$ (i): Suppose that, for all $v \in V^o(T)$, the graph $G(\mathcal{L}, v)$ has at least one edge and assume for contradiction that $\mathcal{L}$ is not an equidistant lasso for T. Then there exist distinct equidistant, proper edge-weightings ω_1 and ω_2 for T such (T, ω_1) and (T, ω_2) are $\mathcal{L}$-isometric. Thus, there must exist some edge $e \in E(T)$ such that $\omega_1(e) \neq \omega_2(e)$.

Assume, without loss of generality, that $e = \{v, v_0\}$ is such that with v being the parent of v_0 we have that $\omega_1(e') = \omega_2(e')$ holds for all edges e' of T that lie on a path from v_0 to a leaf of T contained in $L(v_0)$. Note that v_0 could be a leaf of T in which case such a path has length zero. Let $v_1, v_2, \ldots, v_l$, $l \geq 1$ denote the other children of v. Then, by assumption, $G(\mathcal{L}, v)$ contains at least one edge and so there exist $i, j \in \{0, l\}$ distinct and leaves $y_i, y_j \in L(v)$ that are descendants of v_i and v_j (or coincide with them), respectively, such that $y_i y_j \in \mathcal{L}$. Let $z \in L(v_0)$ denote a leaf of T.

By Property (E1), we obtain

$$\begin{aligned}\omega_1(e)+\sum_{e'\in E_T(v_0,z)}\omega_1(e') &= D_{(T,\omega_1)}(v,z)=\frac{1}{2}D_{(T,\omega_1)}(y_i,y_j)=\frac{1}{2}D_{(T,\omega_2)}(y_i,y_j)\\ &= D_{(T,\omega_2)}(v,z)=\sum_{e'\in E_T(v_0,z)}\omega_2(e')+\omega_2(e).\end{aligned}$$

and so $\omega_1(e)=\omega_2(e)$ follows by the choice of e; a contradiction. Thus, $\mathcal{L}$ must be an equidistant lasso for T. □

Theorem 7 immediately implies that an equidistant lasso on X need not be a covering of X. Also, it implies that the size of a minimum equidistant lasso for an X-tree T is $|V^\circ(T)|$. For the extreme cases that T is the star tree on X such a lasso has precisely one element and if T is binary such a lasso has $|X|-1$ elements as any such tree is known to have $|X|-1$ interior vertices (see e. g. [110]).

Theorem 6 combined with Theorem 7 immediately implies the following link between equidistant and weak lassos.

Corollary 8. *Suppose T is an X-tree and $\mathcal{L}\subseteq\binom{X}{2}$ is a non-empty set of cords. Then $\mathcal{L}$ is an equidistant lasso for T whenever it is a weak lasso for T.*

We remark in passing that dropping the requirement that the two edge-weightings have to be equidistant in the definition of an equidistant lasso gives rise to the definition of an *edge-weight lasso* for an X-tree. However it is easy to see that Theorem 7 does not hold with equidistant lasso replaced by edge-weight lasso.

3.8 A characterization of a topological lasso

In this section, we prove the companion result for when a set of cords is a topological lasso for an X-tree T in terms of the child-edge graphs associated with the interior vertices of T. We start again with some more notation.

Suppose that T is an X-tree and that $v\in V^o(T)$, but not the root of T. Then we denote the $L(v)$-tree obtained from T by deleting the parent edge of v by $T_{L(v)}$. Now suppose that $Y\subsetneq X$ is such that there exists some $v\in V^o(T)$ such that $Y=L(v)$. Then we denote the root of T_Y by $\rho(T_Y)$.

Theorem 9. *Suppose T is an X-tree and $\mathcal{L} \subseteq \binom{X}{2}$ is a non-empty set of cords. Then the following are equivalent:*

(i) $\mathcal{L}$ is a topological lasso for T.

(ii) For every vertex $v \in V^o(T)$, the graph $G(\mathcal{L}, v)$ is a clique.

Proof. (i) $\Rightarrow$ (ii): Suppose that $\mathcal{L}$ is a topological lasso for T and assume for contradiction that there exists some vertex $v \in V^o(T)$ such that $G(\mathcal{L}, v)$ is not a clique. Then there exist child edges $e, e' \in E(T)$ of v such that $\{e, e'\} \notin E_v$. Let v_e and $v_{e'}$ denote the children of v incident with e and e', respectively, and let ω denote an equidistant, proper edge-weighting of T. We distinguish the cases that (i) v is the parent of a pseudo-cherry of T and that (ii) $v \in V^o(T)^-$.

Case (i): Assume that v is the parent of a pseudo-cherry of T. Then since every topological lasso for T is in particular a weak lasso for T, Theorem 6 implies that $G(\mathcal{L}, v)$ is connected and that, in addition to $x := v_e$ and $y := v_{e'}$, there must exist a further child of v (that is a leaf of T). Note that since ω is equidistant, we must have $\omega(e) = \omega(e')$.

Let T' denote the X-tree obtained from T by subdividing the edge e' by a vertex w that is not already contained in $V(T)$, adding the edge $\{x, w\}$ and deleting the edge e. Clearly, T and T' are not equivalent. Let $0 < \varepsilon < \omega(e)$ and consider edge-weighting

$$\omega' : E(T') \to \mathbb{R}_{\geq 0} : f \mapsto \begin{cases} \omega(f) & \text{if } w \notin f, \\ \omega(f) - \varepsilon & \text{if } f \in \{\{w,x\}, \{w,y\}\}, \\ \varepsilon & \text{else.} \end{cases}$$

Clearly, ω' is equidistant and proper and it is straight forward to see that (T', ω') and (T, ω) are $\mathcal{L}$-isometric. Since $\mathcal{L}$ is a topological lasso for T it follows that T and T' must be equivalent, a contradiction.

Case (ii): Assume that $v \in V^o(T)^-$. Then since every topological lasso for T is in particular a weak lasso for T, Theorem 6 implies that $G(\mathcal{L}, v)$ is rich. But then v_e and $v_{e'}$ must be leaves of T. With $x = v_e$ and $y = v_{e'}$ we obtain a contradiction using the same arguments as in Case (i).

(ii) $\Rightarrow$ (i): Suppose that, for every vertex $v \in V^o(T)$, the graph $G(\mathcal{L}, v)$ is a clique and assume that T' is an X-tree and ω and ω' are equidistant, proper edge-weightings

for T and T', respectively, such that (T,ω) and (T,ω') are $\mathcal{L}$-isometric. If T is the star-tree on X then $\mathcal{L}$ must necessarily be a topological lasso for T. So assume that T is non-degenerate. Then, by Theorem 6, $\mathcal{L}$ is a weak lasso for T and so T' must be a refinement of T.

We next show that T and T' are in fact equivalent. Assume for contradiction that T and T' are not equivalent, then there must exist a non-empty subset $Y \subsetneq X$ of leaves of T such that the subtree T_Y of T with leaf set Y is equivalent with the subtree T'_Y of T' with leaf set Y but the subtrees of T and T' with root the parents of $\rho(T_Y)$ and $\rho(T'_Y)$, respectively, are not. Let w denote the parent of $\rho(T_Y)$ in T and w' the parent of $\rho(T'_Y)$ in T'. Then there must exist some $z \in L(w') - L(\rho(T'_Y))$ and distinct $x,y \in L(\rho(T'_Y)) = Y$ such that $xy|z \in \mathcal{R}(T')$ and $xy|z \notin \mathcal{R}(T)$. Hence, $\rho(T_Y)$ must lie on the path from x to y in T. Combined with the fact that T' is a refinement of T, it follows that $T|_{\{x,y,z\}}$ is the star-tree on $Z := \{x,y,z\}$ whose unique interior vertex is $\rho(T_Y)$. Since ω is equidistant, we obtain

$$D_{(T,\omega)}(x,y) = D_{(T,\omega)}(x,z) = D_{(T,\omega)}(z,y). \tag{3.5}$$

Let $e_x, e_y, e_z \in E(T)$ denote the child edges of $\rho(T_Y)$ that are crossed by a path from $\rho(T_Y)$ to x, y, and z, respectively, and let v_{e_s} denote the child of $\rho(T_Y)$ incident with e_s, for all $s \in Z$. Since, by assumption, $G(\mathcal{L}, \rho(T_Y))$ is a clique there must exist leaves $a \in L(v_{e_x})$, $b \in L(v_{e_y})$, and $c \in L(v_{e_z})$ such that $ab, bc, ca \in \mathcal{L}$. By the same reason, it follows in view of Lemma 1 that $D_{(T,\omega)}(p,q) = D_{(T',\omega')}(p,q)$ must hold for any two elements $p,q \in Z$ distinct. Combined with Equality (3.5), we obtain $D_{(T',\omega')}(p,q) = D_{(T',\omega')}(p,q)$, for any two such elements p and q. But then $T'|_Z$ must be the star-tree on Z and so $xy|z \notin \mathcal{R}(T')$, a contradiction. □

Note that Theorem 9 immediately implies that a topological lasso $\mathcal{L} \subseteq \binom{X}{2}$ must be a covering of X. Also note that Theorem 9 implies that if $\mathcal{L}$ is a topological lasso for an X-tree T then $\mathcal{L}$ must contain at least $\sum_{v \in V^o(T)} \binom{|V_v|}{2}$ cords and that $\mathcal{L} = \binom{X}{2}$ must hold in case T is the star-tree on X. Finally, note that as the example presented in Figure 3.4 shows, the requirement that the two proper edge-weightings in the definition of a topological lasso must be equidistant cannot be dropped. More precisely for $X = \{a,b,c,d\}$ and $\mathcal{L} = \{ab, cd, ad\}$ the X-tree T pictured on the left of that figure (with the indicated edge-weighting ω ignored for the moment) satisfies Properties (C1) and (C2)

and is not equivalent with the X-tree T' depicted on the right of that figure (again with the indicated edge-weighting ω' ignored where $0 < \varepsilon < 1$). Also ω and ω' are clearly proper and, in the case of ω, also equidistant and (T, ω) and (T', ω') are $\mathcal{L}$-isometric.

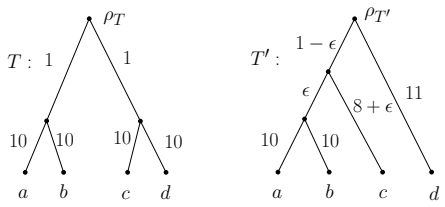


Figure 3.4: An example showing that for Theorem 9 to hold the "equidistant" requirement in the definition of a topological lasso cannot be dropped (see text for details).

Combining Proposition 7 and Theorem 9, we obtain the following corollary.

Corollary 10. *Suppose that T is an X-tree. Then every topological lasso for T must also be an edge-weight lasso for T. Moreover, if T is binary and $\mathcal{L} \subseteq \binom{X}{2}$ then $\mathcal{L}$ is a topological lasso for T if and only if $\mathcal{L}$ is an equidistant lasso for T. In particular, every topological/equidistant lasso for T is also a strong lasso for T in this case.*

Note however that for $\mathcal{L} \subseteq \binom{X}{2}$ a set of cords and T an X-tree it is possible that $\mathcal{L}$ is simultaneously an edge-weight and a topological lasso for T but T is not binary. We present such an example in Section 3.9.

3.9 Examples of lassos

In this section we apply our findings to two types of constructions of sets of cords of X. Both of them were originally introduced in [26] for the case of edge-weighted, unrooted, phylogenetic trees with leaf set X where an edge-weighting for such a tree is defined as in the case of an X-tree.

Assume for the remainder of this section that T is an X-tree. Then the first example relies on the notion of a circular ordering (of the leaf set) of T (see e.g. [110] for further details on such orderings). Following [26] a *circular ordering (of X) of T* is a cyclic permutation σ of X such that the following holds. There exists a planar embedding of T such that, for every $x \in X$, the leaf that is encountered after x when traversing T

in that embedding in say, a counter-clockwise fashion, is the leaf $\sigma(x)$. For example, (a,b,c,d) is a circular ordering of T the $\{a,b,c,d\}$-tree depicted in Figure 3.4(a).

Let $(x_1,x_2,\ldots,x_n)$ denote a circular ordering of the leaves of T where $n := |X|$ and put $x_{n+1} = x_1$. Then since for every interior vertex v of T there exists some $i \in \{1,\ldots,n\}$ such that v lies on the path from x_i to x_{i+1}, Theorem 7 implies that the set

$$\mathcal{L}_c = \{x_i x_{i+1} : 1 \leq i \leq n\}$$

is an equidistant lasso for T. However, $\mathcal{L}_c$ is clearly not an equidistant lasso of minimal size. In view of Theorem 9, $\mathcal{L}_c$ is a topological lasso for T if and only if every interior vertex $v \in V^o(T)$ has degree three (except possibly the root ρ_T which might also have degree two) as in that case $G(\mathcal{L}_c, w)$ is a complete graph for all $w \in V^o(T)$. Thus, $\mathcal{L}_c$ is also a strong lasso for such X-trees. In view of Theorem 6, $\mathcal{L}_c$ is a weak lasso for T if and only if every vertex $v \in V^o(T)^-$ has degree three (except possibly the root ρ_T which might also have degree two) as in that case $G(\mathcal{L}_c, w)$ is rich for all $w \in V^o(T)^-$ and connected for all $w \in V^o(T) - V^o(T)^-$.

Our final construction relies on the notion of a bipartition $\{A,B\}$ of X and was introduced in [26] where it was shown that the set

$$A \vee B := \left\{ ab \in \binom{X}{2} : a \in A \text{ and } b \in B \right\}$$

is a topological lasso[1] for an unrooted phylogenetic tree T' with leaf set X of size 4 or more if and only if, for every 2-subset $\mathfrak{c}$ of X whose elements have that same parent in T', we have that $A \cap \mathfrak{c} \neq \emptyset \neq B \cap \mathfrak{c}$. Note that this implies in particular that for an unrooted phylogenetic tree on X to be topologically lassoed by $A \vee B$, every interior vertex of T can be adjacent with at most two leaves. Thus every pseudo-cherry of T' (defined as in the case of an X-tree) must be a cherry of T.

Defining for an unrooted phylogenetic tree T' on X a set $\mathcal{L} \subseteq \binom{X}{2}$ of cords to be an edge-weight lasso as in the case of an X-tree but again with the requirement "equidistant" on the proper edge-weightings removed, it is not difficult to see that $A \vee B$ is not an edge-weight lasso for T'. Also it is straight forward to see that any two proper

[1]The definition of a topological lasso for an unrooted phylogenetic tree on X is the same as that of a topological lasso for an X-tree but with the requirement dropped that the two proper edge-weightings mentioned in that definition are equidistant.

edge-weightings ω and ω' for T' such that $D_{(T',\omega)}(a,b) = D_{(T',\omega')}(a,b)$ holds for all $ab \in \mathcal{L}$ must coincide on the interior edges of T' where for a proper edge-weighting α of T' we denote the induced distance on $V(T')$ also by $D_{(T',\alpha)}$.

In the case of X-trees the situation changes in so far that if T is non-degenerate and $\{A,B\}$ is such that every pseudo-cherry of T contains elements from both A and B then, in view of Theorem 6, $A \vee B$ must be a weak lasso for T and thus, by Corollary 8, also an equidistant lasso for T. In view of Theorem 9, $A \vee B$ is not a topological lasso for T unless every interior vertex $v \in V^o(T)$ of T is incident with at most two leaves of T and, if v is incident with two leaves, then one is contained in A and the other in B as otherwise $G(A \vee B, v)$ would not be a clique. Since for such X-trees T we have, for all $v \in V^o(T)$, that $G(\mathcal{L}, v)$ contains at least one edge it follows that $A \vee B$ is a strong lasso for T.

If T is the star-tree on X then $A \vee B$ is a weak lasso for T. Also, $A \vee B$ is an equidistant lasso for T as $G(\mathcal{L}, \rho_T)$ contains at least one edge but it is not a topological lasso for T as $A \vee B \neq \binom{X}{2}$.

3.10 Conclusion

In the form of investigating when a set of cords of a finite set X of size at least three is an equidistant/topological/weak/strong lasso for an X-tree, we have addressed the topical problem of when a set of partial distances for a set of individuals within a sample uniquely determines a dendrogram on those individuals. Such structures are commonly constructed as part of a phenetic clustering step within a genomewide association study to better understand the link between phenotypic and a genotypic variation within the sample. For T an X-tree and $\mathcal{L} \subseteq \binom{X}{2}$ a set of cords, we have presented characterizations for when $\mathcal{L}$ is an equidistant/weak/tropological lasso for T in terms of the structure of the child-edge graphs associated to the interior vertices of T. As immediate consequences, our characterizations allow us to not only shed light into the problem of when two of the above types of lassos coincide but also into the size of minimum size equidistant/topological/weak lassos.

Despite these encouraging results a number of open questions remain. For example, the characterizations above require knowledge of the structure of T in the form of the child-edge graphs associated to the interior vertices of T. Thus, is it possible

to characterize or at least understand lassos without this structural insight into T. A potential candidate for this might be the graph $\Gamma(\mathcal{L})$ associated to $\mathcal{L}$ whose vertex set is X and whose edge set is $\mathcal{L}$. The underlying rational for this is that for $|X| \geq 4$, it was shown in [26, Theorem 1] that for $\mathcal{L}$ to be a topological lasso for an unrooted phylogenetic tree T on X the graph $\Gamma(\mathcal{L})$ has to be connected and for $\mathcal{L}$ to be an edge-weight lasso for T it has to be strongly non-bipartite (where a graph G is said to be *strongly non-bipartite* if every connected component of G is not bipartite). Also for $\mathcal{L}$ as constructed in the first example in Section 3.9 the graph $\Gamma(\mathcal{L})$ is connected.

To overcome the potential loss of information in distance based phylogenetic tree reconstruction, [84] proposed using *k-dissimilarities*, $k \geq 3$, on X rather than 2-similarities as is the case when reconstructing edge-weighted phylogenetic trees from distances (see also [35, p.176] and [52, 120, 22] and the references therein for recent work on such objects which are sometimes also called k-way similarities, k-way distances, and k-semimetrics). It would be interesting to know what can be said about lassoing and corraling of X-trees within this more general framework.

Acknowledgement A. -A. Popescu thanks the Norwich Research Park (NRP) for support. The authors thank the referees for their helpful comments.

Chapter 4

ape 3.0: new tools for distance based phylogenetics and evolutionary analysis in R

This chapter is based on

A-A. Popescu, K.T. Huber., E. Paradis ape 3.0: New tools for distance-based phylogenetics and evolutionary analysis in R. *Bioinformatics*, 28:1536–1537, 2012.

A-A. Popescu's contribution is the implementation, testing, comparative benchmarking and writing of the documentation of the newly implemented functions. Also he helped write the first draft of the manuscript.

4.1 Chapter Summary

This chapter focuses on describing our additions to the popular ape package for phylogenetic analysis in R. Our additions comprise a variety of phylogenetic methods for inferring evolutionary scenarios when incomplete data is present, mainly in the form of incomplete distances. We also present tree popping, a splits-based phylogenetic reconstructing algorithm.

Availability ape is distributed through the Comprehensive R Archive Network:
http://cran.r-project.org/web/packages/ape/index.html
Further information may be found at
http://ape.mpl.ird.fr/pegas/

4.2 Introduction

Actively responding to requirements of evolutionary biologists to be able to analyse new types of data as well as larger data sets, ape's features have been improved upon steadily over the years and new functions and object classes have been added by numerous contributors. At the same time, ape has taken a central place in the development of new packages in R so that, to date, about 50 of them depend on it. Consequently, ape now provides, among other things, improved graphical tools for exploring phylogenetic trees as well as for manipulating, comparing and storing them (see [85] for details); new object classes such as `"evonet"` to encode phylogenetic networks; and new features for simulating character evolution, estimating ancestral states [18], and computing sequence alignments by invoking existing programs such as Clustal [12], Muscle [28], or T-Coffee [83]. In parallel, an effort has been made to develop a comprehensive function (`dist.dna`) to compute evolutionary distances from aligned DNA sequences under most published models. At the same time, fast and reliable functions have been incorporated into ape to construct a neighbour-joining tree from a distance matrix [106], and carry out BIONJ [41], and FastME [21] phylogenetic tree estimation from such a matrix; the latter can be done by optimizing either the ordinary least squares or the balanced version of the minimum evolution criterion.

Despite these additions, one of ape's limitations has been its inability to carry out a distance-based phylogenetic analysis in case of incomplete distance information. Such datasets arise, for example, in whole genome studies where there might be incomplete taxonomic coverage or the reliability of distances between some of the taxa under consideration is poor. To allow ape to handle such data, which, for convenience, we refer to as incomplete distances (as opposed to complete distances), we have extended its functionality in three separate but closely interlinked directions: (1) phylogenetic tree building from complete and incomplete distances, (2) estimation of distance values from incomplete distances without explicitly constructing a phylogenetic tree before-

hand, and (3) computation of consensus distance matrices.

Although the methods we have added to ape are already implemented in some specialized computer programs, it is the first time that they can be found together in a single package. In addition to generally running faster than the original versions due to our implementations being based on C (see Section 4.4 for more on this), this makes it possible to directly compare the results from different methods, and to interface them with the many data analysis options provided by ape, thus reducing the possibility of error due to data conversion or compatibility issues.

This chapter is structured as follows. We first describe the new implemented functions and how they are called. Subsequent to this we go into details on how our implementations fare against already existing alternatives, while also presenting the methodology we used to test these methods.

4.3 New Features

In this section we briefly outline all the new methods which have been implemented. Each such outline is followed by details on how the new functions can be called. Unless stated otherwise, in what follows, `dsm` is an incomplete distance matrix and `d` is a complete distance matrix. Also all phylogenetic trees are edge-weighted phylogenetic trees.

An attractive feature of distance-based phylogenetic reconstruction is that a tree can be constructed in a relatively short amount of time. Thus, there has been considerable interest in developing these methods both from complete and incomplete distances. As an alternative to the NJ, BIONJ, and FastME methods mentioned above, which all take complete distances as input, we have added the triangles method [66] as `triangMtd` to ape. Some versions of these methods for incomplete distances exist in the form of BIO-NJ*, NJ* [17], and the triangles method for incomplete distances [49] which we have also added to ape as `bionjs`, `njs`, and `triangMtds`, respectively.

- `bionjs` takes as input an incomplete distance matrix and constructs a phylogenetic tree using the *Bio-NJ** algorithm. The method is invoked as:
  `ptr<-bionjs(dsm)`
  or

`ptr<-bionjs(dsm,i)`
where `i` can be any positive integer, representing the agglomeration parameter of the method. In case `i` < 1 then an error message is returned.

- `njs` takes as input an incomplete distance matrix and constructs a phylogenetic tree using the *NJ** method. It is invoked by putting:
  `ptr<-njs(dsm)`
  or
  `ptr<-njs(dsm,i)`
  where `i` can be any positive integer, representing the agglomeration parameter of the method. In case `i` < 1 then an error message is returned.

- `triangMtd` takes as input a complete distance matrix and returns a phylogenetic tree constructed from that distance matrix using the *triangles method*. It is called by putting:
  `ptr<-triangMtd(d)`

- `triangMtds` is the extension of the triangles method to incomplete distances and takes as input an incomplete distance matrix and returns a phylogenetic tree constructed from that distance matrix. It is called by putting:
  `ptr<-triangMtds(dsm)`

One way to deal with incomplete distances is to first restrict attention to a subset of the data for which complete distance information is available, and then to somehow fit the remaining data into the phylogenetic tree constructed from that subset. This is the philosophy underpinning, for example, the triangles method. An alternative to this is to directly estimate the missing distances from the data without first constructing a phylogenetic tree. Two methods that rely on this idea are the ultrametric and the additive procedure [74], respectively, which we have implemented in `ape` as `ultrametric` and `additive`.

- `additive` takes as input an incomplete distance matrix and attempts to infer missing matrix entries using the *additive procedure*. A call to `additive` is as follows:
  `dsmAdd<-additive(dsm)`

- `ultrametric` takes as input an incomplete distance matrix and infers missing matrix entries using the *ultrametric procedure*. It is called by putting:
  `dsmUltra<-ultrametric(dsm)`

Supertree methods have been used to combine trees from different studies. Essentially they work by combining a collection of trees (e.g. gene trees) into a parental tree which, in a well-defined way, represents the given trees. By contrast to the consensus tree method in `ape`, the taxa sets of the trees do not need to be the same, and may only be overlapping.

As a partial response to the criticism that supertree reconstruction methods only use tertiary data (i.e., phylogenetic trees obtained from sets of distance matrices), consensus distance matrix approaches have been introduced in the literature. Starting from several overlapping taxa sets, each with complete distance information, this boils down to finding ways to compute the distance between any two taxa that are in the union of all taxa sets but not in the same set. A tool that allows one to do this is the superdistance matrix (SDM) [16] method which we have incorporated into `ape` as `SDM`. It should be noted that, as proposed by [16], our implementation returns not only the consensus distance matrix for an input data set, but also the matrix of associated variances. Although any standard tree building method could potentially be used to reconstruct a phylogenetic tree from a consensus distance matrix, some specialized methods have been developed which also take its associated variance matrix into account when building the tree. An example of such a method is minimum variance reduction (MVR) [43] which we have added to `ape` as `mvr`, as well as, in the form of `mvrs`, its extension MVR* to incomplete distances [17].

- `mvr` is an implementation of the *weighted least squares (WLS) minimum variance reduction (MVR)* method. It takes as parameters a complete distance matrix and a matrix `v` of associated variances, and applies the MVR-WLS algorithm [43]. It is called by putting:
  `ptr<-mvr(d,v)`

- `mvrs` takes as input an incomplete distance matrix and a matrix `vsm` of associated variances and constructs a phylogenetic tree using the *MVR** method. It is called as:

`ptr<-mvrs(dsm,vsm)`
or
`ptr<-mvrs(dsm,vsm,i)`
where `i` can be any positive integer, representing the agglomeration parameter of the method. In case `i` < 1 then an error message is returned.

- `SDM` takes as input at least two distance matrices d_j, $j \geq 2$, followed by j integers w_j representing the respective matrix weights, and applies the *SDM* method to produce a two-element list. The first element in that list is the consensus distance matrix and the second element is the matrix of associated variances. The method is invoked by putting:
`ret<-SDM(`d_1`,`d_2`,`w_1`,`w_2`)` (in case $j = 2$)
and
`ret<-SDM(`d_1`,`d_2`,`d_3`,`w_1`,`w_2`,`w_3`)` (in case $j = 3$)

To take advantage of a combinatorial description of a phylogenetic tree in terms of a collection of weighted splits, we have developed a new class, `bitsplits`, in order to represent this type of object. The need for such a class arises in the context of, for instance, supertree reconstruction. Due to, among other things, noise in the data, it is in general too much to hope for that the given set of trees will fit together nicely into a supertree. To help with this, we have developed the function `is.compatible`, which allows one to quickly check if a collection of splits is compatible (see also below), and, as `treePop`, a weighted version of Meacham's tree popping method [78] which allows one to construct a phylogenetic tree from a collection of weighted splits (see also below and e.g. [110] for details). Another advantage of this new class is that it will ease the development of further distance-based methods such as the refined Buneman trees method [7] which relies on computing such collections. To interface the new class with other functionalities in `ape`, we wrote the function `as.bitsplits` allowing one to convert from the already existing `prop.part` class present in `phangorn` R package [108].

- `is.compatible` checks whether a given `bitsplits` object `bs` represents a compatible split system. It relies on the novel `arecompatible` function which takes

as input two splits, encoded by bits, and checks whether they are compatible or not. The former is invoked by putting:
`is.compatible(bs)`

- `treePop` is an implementation of *Meacham's tree popping*. It takes as input a set of weighted splits each encoded by a `bitsplits` object `bs`,and applies tree popping to that set to produce a phylogenetic tree. The set of splits induced by that tree is known to coincide with the input set of splits. It is called by putting:
`ptr<-treePop(bs)`

4.4 Testing and Benchmarking

In addition to verifying the correctness of our implementations by means of small hand-worked examples, we employed the testing protocol outlined below. In each case, we used randomly generated input data obtained as follows: Using `ape`'s rtree function, we first generated a random edge-weighted phylogenetic tree T on n leaves. In case the implementation in question required a distance matrix as input, we took as randomly generated distance matrix the induced distance matrix D_T of a randomly generated edge-weighted phylogenetic tree T. For comparison purposes we generated a PHYLIP format representation of such a matrix as all programs against which we tested read in matrices in that format.

The comparison of the generated edge-weighted phylogenetic trees was performed by comparing their induced distance matrices using R's `all.equal` function. To aid with this we always transformed the output generated by an alternative implementation into a suitable `ape` object using `ape`'s `read.tree` function.

We next discuss the testing of the newly added methods in more detail. In each case a phylogenetic tree is an edge-weighted phylogenetic tree.

4.4.1 NJ*, Bio-NJ*, MVR and MVR*

We chose $n = 400$ as the leaf set size of our randomly generated phylogenetic trees T. The induced distance matrix D_T of such a tree T we then used as input to NJ* and Bio-NJ*. In all cases we found the trees output by the methods to be in agreement with T.

In the case of the MVR and MVR* methods we used the distance matrix D_T, induced by a randomly generated phylogenetic tree T, as input for both the distance and associated variance matrix. Again we found the respective trees returned by the two methods to be in agreement with T.

We compared our implementations of NJ*, Bio-NJ*, MVR and MVR* against those of PhyD* [17]. For this, we chose the leafset size of T to be 15 and removed some of the entries from the induced distance matrix D_T. The resulting (incomplete) distance matrix D_T^- we then used as input for the respective implementations of the above methods in PhyD* and `ape`. The phylogenetic trees generated by NJ*, Bio-NJ*, MVR and MVR* we then compared in terms of their induced distance matrices. These we found to be in agreement in all cases. We remark that in the case of the MVR and MVR* methods the matrix D_T^- served as both input distance matrix and matrix of associated variances, with D_T^- being replaced by D_T in the case of MVR.

In order to test MVR in case the input distance matrix did not equal the matrix of associated variances, we carried out four separate experiments using some of the trees in the data set accompanying the original release of SDM (see also Section 4.3). More precisely we choose the first 2,3,4 and 5 phylogenetic trees T_i, $1 \leq i \leq 5$, in one of those datasets and then called SDM on the induced distance matrices D_{T_i} in each case. For this we used i as the weight of D_{T_i} $1 \leq i \leq 5$. The resulting distance matrix and matrix of associated variances we then used as input for both implementations of MVR (when the consensus matrix had no missing entries), and MVR*. Again, we found the results to be in agreement.

We next compare the runtimes of our implementations of NJ*, BioNJ*, and MVR* with that of their counterpart Java-based implementations in PhyD* which were produced by their original authors. Since PhyD* involves reading input matrices from files and writing the output tree to a file whereas in our implementations the input is readily available in R and the output is returned as an R object we only measured the amount of time it took PhyD* to generate the output from an (in)complete distance matrix so as not to give our implementations an unfair advantage.

For three randomly generated phylogenetic trees T on $n = 96, 192, 400$ leaves (where the former two values are taken from a performance study carried out by the original authors of the methods of interest), we generated the respective distance matrices D_T from which we then removed entries. We present our findings in Table 1.

	APE						PhyD*					
method	NJ*		Bio-NJ*		MVR*		NJ*		Bio-NJ*		MVR*	
missing	25%	75%	25%	75%	25%	75%	25%	75%	25%	75%	25%	75%
400 taxa	6.7s	4s	6.7s	4s	6.7s	4s	2.3m	43s	2.3m	43s	2.3m	43s
192 taxa	<1s	<1s	<1s	<1s	<1s	<1s	16.5s	5.5s	16s	5.6s	16.5s	5.6s
96 taxa	<1s	<1s	<1s	<1s	<1s	<1s	1.9s	<1s	1.9s	<1s	1.9s	<1s

Table 4.1: The runtimes of our implementations as compared to those in PhyD*. The considered values for the leafset size of T are presented in the first column and the percentages of missing values from D_T are given in the row entitled "missing".

As can be easily seen from Table 1, the runtime of our implementations is faster than the runtime of the corresponding implementations in PhyD*.

4.4.2 SDM

Following the same strategy as for MVR and MVR*, we tested our implementation of SDM against the Java implementation of that method in the program SDM produced by the original authors of SDM. We remark that SDM was required to infer missing matrix entries in all but the first experiment. Independent of the number of input distance matrices considered, we found the corresponding consensus distance matrices and matrices of associated variances to be in agreement.

4.4.3 Triangles method for complete distances

We chose $n = 1000$ as the leaf set size of our randomly generated phylogenetic tree T and used the induced distance matrix D_T as input to the method. That distance matrix we then compared against the distance matrix of the phylogenetic tree inferred by the method. Both we found to be in agreement.

4.4.4 Triangles method for incomplete distances

We compared our implementation with the implementation of that method in the T-Rex software package [73]. As leafset size of T we chose again $n = 15$, and, as above, we removed entries from the distance matrix D_T. The resulting incomplete distance

matrix we then used as input for both implementations of the method. Again, we found the induced distance matrix of the generated phylogenetic trees to be in agreement.

4.4.5 Additive and ultrametric

We compared our implementations against the respective implementations of both methods in T-Rex. To this end, we again generated a distance matrix D_T, T a randomly generated phylogenetic tree on 15 leaves, from which we then removed entries. The resulting incomplete distance matrix D_T^- we then used as input for the respective implementations. Since T-Rex represents the generated respective distance matrices in the form of a NJ-tree we also represented the distance matrices generated by our respective implementations in the form of such a tree. Again, we found the respective induced distance matrices of the NJ-trees to be in agreement.

4.4.6 Tree popping and is.compatible

We chose $n = 1000$ as leaf set size of the randomly generated edge-weighted phylogenetic tree T. From that tree we then inferred its induced set $\Sigma(T)$ of splits plus the weight w_S of each split $S \in \Sigma(T)$. The set $\Sigma(T)$ we then tested for compatibility using the `is.compatible` function. As expected, the method found $\Sigma(T)$ to be compatible. We then applied the tree popping algorithm to $\Sigma(T)$ (and associated split weights) and found the generated phylogenetic tree to be in agreement with T.

To further test the tree popping algorithm and `is.compatible`, we also used a set of splits that contained non-compatible split as input. As expected `is.compatible` found that set not to be compatible and the tree popping algorithm returned an empty tree.

Acknowledgement

The authors would like to thank Alexis Criscuolo, Olivier Gascuel, and Klaus Schliep for their feedback and suggestions, and the referees for their helpful comments.

II: Population Structure and NGS data

Chapter 5

A Novel and Fast Approach for Population Structure Inference Using Kernel-PCA and Optimization (PSIKO)

This chapter is based on

A-A. Popescu , A.L Harper, M. Trick, I. Bancroft and K.T. Huber. A novel and fast approach for population structure inference using kernel-PCA and optimization. *Genetics*, 198(4):1421–31, 2014. doi: 10.1534/genetics.114.171314

A-A. Popescu's contribution is writing first drafts for the paper, developing the simulation framework and contributing to the development of the main algorithm described in the paper.

5.1 Chapter Summary

In this chapter we introduce a novel algorithm that promises to efficiently handle inferring population structure from NGS data. This is a long-standing and well-studied problem, however current tools tend to struggle with the sheer volume of data pro-

vided by NGS technology. Hence we developed PSIKO, a quick and scalable tool that promises to aid the problem of inferring population structure from NGS data. We start by presenting the algorithm, then proceed to testing it on a variety of simulated and real biological datasets, showing its good performance in practice and its scalability.

5.2 Introduction

Population stratification has been commonly used to investigate the structure of natural populations for some time and is also recognised as a confounding factor in genetic association studies [65, 75]. As a result, programs for detecting population stratification have become a standard tool for genetic analysis. Such approaches generally separate into two classes. Model-based approaches such as STRUCTURE [98] and the closely related ADMIXTURE approach [1] are desirable in that they return a Q-matrix which for each accession of the (marker) dataset indicates the proportion of its genotype that came from one of $K \geq 2$ assumed founder populations. This biological interpretability of Q-matrices conveniently lends itself to a subsequent use in association studies. On the other hand, such approaches often suffer from long runtimes, particularly as dataset size increases. This problem is becoming particularly exacerbated with the increased use of Next Generation Sequencing (NGS) and large SNP chips to develop marker datasets [63, 116]. Conversely, non-model based approaches such as EIGENSTRAT [95] which uses Principal Component Analysis (PCA), tend towards much shorter runtimes making them more convenient when analysing large marker sets. Unfortunately, EIGENSTRAT only returns principal components (PCs) of a dataset and not a Q-matrix. Some non-model based approaches such as the recently introduced sparse-Non-negative-Matrix-Factorization (sNMF) method [39], have made advances regarding these issues, and output a Q-matrix for use in association genetic analysis whilst significantly shortening run-times. Like EIGENSTRAT, sNMF can be thought of as a feature extraction approach aimed at reducing the dimensionality of a high dimensional dataset. However the matrices used by both approaches to achieve this reduction have different mathematical properties [61]. Even so, sNMF still suffers from longer runtimes with increased number of markers.

In this chapter, we propose the novel PSIKO approach which is linear-kernel PCA based. Like EIGENSTRAT, PSIKO returns significant principal components of a

dataset. Contrary to EIGENSTRAT though, it also generates Q-matrices and these are of comparable quality to those produced by STRUCTURE, ADMIXTURE, and sNMF, at the same time greatly reducing runtime. In addition PSIKO's scaling properties are better than sNMF's (and thus STRUCTURE's and ADMIXTURE's) when the dataset size increases, making it particularly attractive for large datasets.

We rigorously tested the performance of PSIKO using simulated datasets, designed to evaluate the effects of inbreeding, noise, missing data, and SNP pruning, whilst enabling us to compare runtime and scaling properties in comparison to leading approaches such as STRUCTURE, ADMIXTURE and sNMF.

Although we simulated a range of biologically motivated scenarios, as a more realistic test, we also assessed the performance of PSIKO for Q-matrix estimation from two biological datasets. The first of these was a relatively small diversity panel comprising 84 *Brassica napus* lines which had been previously used to perform associative transcriptomics of seed traits [51]. This dataset is of particular interest as it could be considered to have a complex evolutionary history. *B. napus* is a relatively recently formed species, having arisen from spontaneous hybridisation between *B. rapa* and *B. oleracea* as little as 10,000 years ago. It exhibits considerable phenotypic variation, includes spring, semi- and winter ecotypes and has been cultivated as both vegetable and oilseed crops. The most intensive breeding occurred over the last 50-60 years to produce the most commonly used 'canola type' oilseed rape cultivars with both low erucic acid and low glucosinolate content in the seed. Many of the lines in this biological dataset will have been included in these breeding programmes and certain groups (such as the winter oilseed rape lines) may have a complex breeding history. Despite this, the wide diversity of accessions in the panel enabled 101,644 SNP markers to be discovered. Originally the population stratification of this set of accessions was analysed using STRUCTURE before using the identified Q-matrix in a mixed linear association model (MLM). We decided to compare the Q-matrices from PSIKO to those of STRUCTURE as well as sNMF and ADMIXTURE, and determine how these Q-matrices affect the results of the MLM for the original seed oil traits.

On its own and in combination with PLINK's (a popular whole genome analysis toolkit, see [99]) sliding window SNP pruning procedure, we also tested the Q-matrices produced by PSIKO and the three other methods under investigation on a subset of the HapMap Phase 3 project dataset [117]. This dataset should provide a more standard

random mating model than the Brassica dataset, whilst providing an excellent real-life example of the very large marker datasets that will become more common with the advances in sequencing technology.

This chapter is structured as follows. We first present the novel PSIKO algorithm, reviewing relevant concepts from biology and data mining in the process. Subsequent to this, we describe the generation of simulated datasets, which we used to test and benchmark PSIKO. We then proceed to describing in more details the biological datasets we used for validating PSIKO under more realistic scenarios. We subsequently discuss the performance of PSIKO for the above simulated and real datasets. We conclude with a discussion of our findings and how PSIKO may be used to gain relevant insight into population structure.

5.3 Materials and Methods

In this section, we first provide an outline of PSIKO in terms of a two step approach and then describe these two steps in detail. This also includes a brief description of kernel-PCA [109] as its main underlying technique. We then present details on the simulation experiments and the real biological datasets that we used to assess the performance of PSIKO, where the former also includes behaviour under noise, missing data, inbreeding, large datasets, and SNP pruning.

We start with remarking that we follow Chapter 2, Section 2.5.2, to obtain a *SNP matrix* that is, a $d \times n$ matrix whose entries are 0,1 and 2 from a dataset given in terms of a sequence of $d \geq 1$ SNPs and $n \geq 1$ accessions.

5.3.1 Method outline

Given a dataset $\mathbf{X}$ in the form of a $d \times n$ SNP matrix, PSIKO aims to infer the number K of founders of $\mathbf{X}$ as well as significant PCs and a Q-matrix. See Chapter 2, Section 2.5.2 for a more detailed description of these quantities. It consists of two main steps: dimensionality reduction (Step I) and population structure inference (Step II). The purpose of Step I is to infer significant principal components of $\mathbf{X}$ and also obtain an estimate for K. For this we use a combination of the Tracy-Widom test [88] with a powerful PCA-based technique called linear-kernel PCA. This technique allows us

to use the fact that SNP data can only attain a small number of numerical values to efficiently compute a PCA projection of $\mathbf{X}$ via bitwise operations on the entries of $\mathbf{X}$. When describing Step I, we show the bit-wise operations that can be used for efficient PCA, and present an outline of kernel-PCA. The purpose of Step II is to quickly find good estimates for the *ancestry coefficients*, that is, the entries of the Q-matrix. For this, we exploit the properties of a PCA-reduced dataset to cast the problem of inferring population structure within a least squares optimization framework.

Step I: Dimensionality reduction PCA is a popular dimensionality reduction method that allows one to reduce the number of variables of the input dataset $\mathbf{X}$ (given in terms of d), at the same time keeping as much variability in the data as possible. It has proven very useful in population genetics and found in [88] and [72] to exhibit desirable properties when applied to datasets containing admixed individuals. Also, one can make use of the fact that SNP data can only attain values $0, 1$ and 2 to efficiently compute the quantities required for PCA using bitwise operations. We first go into details about these bitwise operations, and subsequently outline the kernel-PCA technique.

Suppose $\mathbf{X} = (\mathbf{X}_{ij})_{\substack{1\leq i\leq d\\ 1\leq j\leq n}}$ is an input dataset given in terms of a genotype matrix where n is the number of individuals and d is the number of SNPs. Then PSIKO associates two $d \times n$ matrices, $X^1 = (X^1_{ij})_{\substack{1\leq i\leq d\\ 1\leq j\leq n}}$ and $X^2 = (X^2_{ij})_{\substack{1\leq i\leq d\\ 1\leq j\leq n}}$ of bits to $\mathbf{X}$. To obtain an entry $\mathbf{X}_{ij}$ of $\mathbf{X}$ it proceeds as follows. If $\mathbf{X}_{ij}$ is 0, then it sets $X^1_{ij} = 0$ and $X^2_{ij} = 0$. If $\mathbf{X}_{ij}$ is 1, then it sets $X^1_{ij} = 1$ and $X^2_{ij} = 0$. If $\mathbf{X}_{ij}$ is 2, then it sets $X^1_{ij} = 1$ and $X^2_{ij} = 1$. We summarise these assignments in Table 5.1. This method of storing $\mathbf{X}$ uses much less space than storing its entries as actual numeric values.

$\mathbf{X}_{ij}$	0	1	2
X^1_{ij}	0	1	1
X^2_{ij}	0	0	1

Table 5.1: Numerical values in $\mathbf{X}$ and their corresponding values in X^1 and X^2.

We next outline how the above storage strategy allows for very fast computation of the $n \times n$ Gram matrix $\mathbf{X}^T\mathbf{X}$ between the individuals of $\mathbf{X}$, which is the first step in carrying out linear-kernel PCA. From $\mathbf{X}^T\mathbf{X}$, a kernel-PCA dimension reduction is obtained via an eigen-decomposition. It is easy to check that the dot product between

two individuals i and j of $\mathbf{X}$ (stored as above) is given by

$$bits(X^1_{:i} \wedge X^1_{:j}) + bits(X^1_{:i} \wedge X^2_{:j}) + bits(X^2_{:i} \wedge X^1_{:j}) + bits(X^2_{:i} \wedge X^2_{:j}).$$

Here $\wedge$ represents the logical bit-wise *and* operation, $bits(b)$ represents the number of set bits of a bit string b and $X^k_{:i}$ represents column i of the matrix X^k, $k = 1,2$. By using the available built-in `C++` bitwise operations, the above quantity can be computed extremely fast even for a very large number of SNPs, thus providing a very quick way of obtaining PSIKO's Gram matrix.

One of the challenges faced by PCA when applied to NGS data is the very large number of variables (SNPs). This implies the need to compute a very large $d \times d$ covariance matrix for $\mathbf{X}$ and also requires computing an eigen-decomposition of such a matrix. Several approaches exist in the literature for addressing this problem. Singular Value Decomposition (SVD) [46] is one such approach. SVD is advantageous because of its good runtime complexity and because it does not require the computation of a large covariance matrix. Another approach is exploiting the relationship between the covariance matrix ($\mathbf{X}\mathbf{X}^T$) and the Gram matrix ($\mathbf{X}^T\mathbf{X}$) of the dataset to avoid the need to manipulate a large matrix (this is also used by kernel-PCA, see below). As it turns out, a particularity of NGS data (the fact that variables can attain only the values $0, 1$ and 2) implies that this latter approach is in fact more efficient than SVD (see Chapter 6, Section 6.3.2 for a comparative benchmark). This is due to the fact that, as described above, the Gram matrix can be computed very efficiently using highly optimised bit-wise operations. However, this efficient computation requires that the entries in the matrix $\mathbf{X}$ retain their initial $0,1,2$ values. This is a problem, as then the matrix is no longer centred. To overcome this problem, we make use of kernel-PCA, which does not require the input matrix be centred, but rather performs a pseudo-centring of the computed Gram matrix. We next go into more details about kernel-PCA.

Rather than carrying out a PCA-analysis directly on a given dataset, in kernel-PCA that dataset is first projected to some new higher dimensional (unknown) feature space, and then classic PCA is applied to the resulting projection of the dataset. Due to its centrality to PSIKO, we next describe it within a kernel-PCA setting [82].

For $\mathbf{X}$ as above, we start with remarking that if it is centred as described in [95] then performing PCA on it reduces to finding an eigen-decomposition of the $d \times d$-

dimensional sample covariance matrix $\mathbf{X}\mathbf{X}^T$. Suppose $\mathbf{W}$ is the matrix of eigenvectors and Λ is the diagonal matrix of eigenvalues of such a decomposition. With this in mind, kernel-PCA can be summarised as follows.

On a high level, the data first gets projected via a map $\phi(x_i) = \phi_i$ from the original feature space to some new higher dimension feature space ϕ, and classic PCA is carried out in this feature space. In practice, this feature space is hard to compute, so instead we apply what is known as the *kernel trick* (see below). To do this, we first note that a PCA projection can also be computed by using the matrix $\mathbf{K} = \mathbf{X}^T\mathbf{X}$ of inner products between observations. We can do this as follows. Letting $\mathbf{U}$ denote the matrix containing the eigenvectors of $\mathbf{K}$ and Λ an $l \times l$ diagonal matrix whose diagonal elements correspond to the l eigenvalues, we have that $(\mathbf{X}\mathbf{X}^T)\mathbf{W} = \mathbf{W}\Lambda$. Pre-multiplying by $\mathbf{X}^T$ gives us $(\mathbf{X}^T\mathbf{X})(\mathbf{X}^T\mathbf{W}) = (\mathbf{X}^T\mathbf{W})\Lambda$. But then the eigenvectors $\mathbf{U}$ of $\mathbf{X}^T\mathbf{X}$ are just $\mathbf{X}^T\mathbf{W}$ with eigenvalues the same as $\mathbf{X}\mathbf{X}^T$. These eigenvectors are however not normalised, since their norm is given by $||u_j||^2 = w_j^T\mathbf{X}\mathbf{X}^T w_j = \lambda_j w_j^T w_j = \lambda_j$. Hence, to normalise u_j, we need to divide that vector by $\sqrt{\lambda_j}$. Finally, the normalised eigenvectors of $\mathbf{X}\mathbf{X}^T$ are given by $\mathbf{W} = \mathbf{X}\mathbf{U}\Lambda^{-\frac{1}{2}}$.

Based on this, in order to apply the kernel PCA we need to take the following steps. We replace all elements $\mathbf{K}_{ij} = \langle \mathbf{x}_i, \mathbf{x}_j \rangle$ of $\mathbf{K}$ with $\kappa(\mathbf{x}_i, \mathbf{x}_j)$. Denote this new matrix by $\mathbf{K}_\phi$. We call κ a kernel function, which can be thought of as a proxy for the dot product between observations $\mathbf{x}_i$ and $\mathbf{x}_j$ in the feature space ϕ. Then in order to find the projection of $\mathbf{x}$ in ϕ we need to compute $\phi_*^T\mathbf{W}_\phi = \phi_*^T\Phi\mathbf{U}_\phi\Lambda^{-\frac{1}{2}}$, where $\phi_* = \phi(\mathbf{x})$ and Φ is the matrix $\mathbf{X}$ with all columns $\mathbf{x}_i$ replaced by $\phi(\mathbf{x}_i)$, $1 \leq i \leq n$, and $\mathbf{U}_\phi$ is obtained from the eigen decomposition of $\mathbf{K}_\phi$. Since in practice computing ϕ is difficult, we can instead proceed as follows. The projection of $\mathbf{x}$ can be re-written as $\mathbf{k}\mathbf{U}_\phi\Lambda^{-\frac{1}{2}}$, where $\mathbf{k} = (\langle \phi(\mathbf{x}), \phi(\mathbf{x}_1) \rangle, \langle \phi(\mathbf{x}), \phi(\mathbf{x}_2) \rangle, \ldots, \langle \phi(\mathbf{x}), \phi(\mathbf{x}_n) \rangle)$. Now we replace all entries of $\mathbf{k}$ by their kernel evaluations, obtaining $\mathbf{k}' = (\kappa(\mathbf{x}, \mathbf{x}_1), \kappa(\mathbf{x}, \mathbf{x}_2), \ldots, \kappa(\mathbf{x}, \mathbf{x}_n))$. Then the projection of $\mathbf{x}$ in ϕ is obtained by computing $\mathbf{k}'\mathbf{U}_\phi\Lambda^{-\frac{1}{2}}$. This replacement of all inner products between observations by an appropriately chosen kernel function is called the kernel trick, and allows us to project our data in ϕ without having to compute ϕ explicitly. The gain in speed of kernel-PCA over PCA (and thus the ability to cope with large NGS datasets) is an immediate consequence of the fact that computing $\mathbf{K}_\Phi$ requires $O(n^2 d)$ operations and a further $O(n^3)$ are required for its eigen-decomposition, (as opposed to $O(d^2 n)$ and $O(d^3)$ for PCA for the corresponding tasks) which amounts

to considerably fewer operations for kernel-PCA when d is much larger than n. Additionally, kernel-PCA allows us to start of with a non-centred input matrix $\mathbf{X}$. This is because we can centre the matrix $\mathbf{K}_\Phi$ after it has been computed without having to worry about whether $\mathbf{X}$ was centred or not. This is done by setting

$$K_{centred} = K - \mathbf{1}_n K - K\mathbf{1}_n + \mathbf{1}_n K \mathbf{1}_n$$

where $\mathbf{1}_n$ represents an $n \times n$ matrix where each element has value $\frac{1}{n}$.

With the above in mind, for Step I we proceed as follows. We first perform a linear kernel-PCA for $\mathbf{X}$, that is, we take the kernel function to be the inner product between accessions of $\mathbf{X}$. Subsequent to this we subject the resulting eigenvalues to the Tracy-Widom test to identify significant principal components (see e. g. [90] for a survey of attractive alternative approaches). This test has proven very popular in population genetics and relies on the fact that non-zero eigenvalues of a matrix follow a Tracy-Widom distribution. Checking whether an eigenvector is a significant principal component of that matrix or not then reduces to checking whether its associated eigenvalue passes a certain statistical significance test [88].

Step II: Population Structure Inference Simulation studies indicate that a PCA-reduced dataset $\mathbf{X}$ obtained in Step I can be represented in terms of a $(K-1)$-dimensional simplex $\mathcal{S}_{K-1}$ where $K \geq 2$ (see e. g. Figure 5.2 for examples for the case $K = 3$, and [88] and [72] where this phenomenon has also been observed for general K). The vertices of such a simplex correspond to the putative *founders* of the dataset, that is, its non-admixed accessions. The position of an accession relative to these vertices encodes the admixture proportion of that accession in the sense that it can be uniquely expressed as a convex combination of the vertices of that simplex. Put differently, with $\mathbf{a}_1, \mathbf{a}_2, \ldots, \mathbf{a}_K$ denoting the vertices of the simplex $\mathcal{S}_{K-1}$ representing a dataset $\mathbf{X}$ found in Step I, any of its accessions $\mathbf{x}$ can be expressed as

$$\mathbf{x} = \sum_{i=1}^{K} \lambda_i \mathbf{a}_i,$$

where, for all $1 \leq i \leq K$, the quantity $\lambda_i \geq 0$ is the genetic contribution of founder $\mathbf{a}_i$ to $\mathbf{x}$ and $\sum_{i=1}^{K} \lambda_i = 1$. Thus, the components of the *ancestry vector* $\lambda_\mathbf{x} = (\lambda_l)_{1 \leq l \leq K}$ of $\mathbf{x}$ can be thought of as the admixture coefficients of $\mathbf{x}$ and computing them is straight

forward using standard arguments from linear algebra if the matrix $\mathbf{A} = (\mathbf{a}_1, \mathbf{a}_2, \ldots, \mathbf{a}_K)$ of founders is known (see below). If this is not the case then, by viewing the matrix $\mathbf{A}Q$ as an approximation of $\mathbf{X}$, the matrix $\mathbf{A}$ (and thus, by virtue of the above, the Q-matrix of $\mathbf{X}$) can be inferred using least squares optimisation. This boils down to minimizing, for a PCA-reduced SNP matrix $\mathbf{X}$ found in Step I, the quantity

$$||\mathbf{X} - \mathbf{A}Q||_F^2, \tag{5.1}$$

with respect to $\mathbf{A}$ and $Q = (\lambda_{\mathbf{x}})_{\mathbf{x}\in\mathbf{X}}$, and $||B||_F$ is the Frobenius norm of a matrix (see Chapter 2, Section 2.5.5). We can solve Equation 5.1 by employing an iterative optimisation technique similar to that in [115].

We start by making some observations that are specific to optimising Equation (5.1), and then present in Algorithm 6 an efficient algorithm for minimising it.

Suppose we are given a matrix Q. Finding a matrix $\mathbf{A}$ which minimises Equation (5.1) can easily be achieved via linear least-squares optimisation. More precisely, we have that

$$\mathbf{x} = \sum_{i=1}^{K} q_{\mathbf{x}i}\mathbf{a}_i, \tag{5.2}$$

holds for any accession $\mathbf{x}$ in our data set. In the context of optimising Equation (5.1) we are interested in finding values for $\mathbf{a}_i$, $1 \leq i \leq K$ such that a given accession $\mathbf{x}$ in $\mathbf{X}$ is approximated as closely as possible by Equation (5.2). This can be achieved by using:

Observation 1.

$$\mathbf{A} = (Q^T Q + \Gamma\Gamma^T)^{-1} Q^T \mathbf{X}^T, \tag{5.3}$$

where Q and $\mathbf{X}$ are as before and $\Gamma = \alpha\mathbf{I}$ is a Tikhonov regularisation matrix.

Usually, the parameter α from Equation 5.3 would be inferred using some form of cross-validation. Indeed, a popular approach in the literature called Leave One Out Cross Validation (LOOCV) can help infer α without the computational overhead required for regular cross-validation [104]. However, a brief study using simulated data show that in practice a value of $\alpha = 1$ produces results that are generally very similar to a LOOCV-inferred value of α. This, in combination with the computational overhead of performing additional operations at each iteration of the algorithm has

lead us to choose implementing PSIKO with $\alpha = 1$ rather than following a LOOCV approach.

Now consider the converse problem, i. e. that the matrix $\mathbf{A}$ is known, and that we are interested in finding the matrix Q. For this we once again use Equation (5.2) above. More precisely, utilising the fact that $\sum_{i=1}^{K} q_{\mathbf{x}i} = 1$ holds for all $\mathbf{x} \in \mathbf{X}$, we obtain:

Observation 2. *Let* $\mathbf{B} := \begin{pmatrix} 1 & 1 & \dots & 1 \\ \mathbf{a}_1 & \mathbf{a}_2 & \dots & \mathbf{a}_k \end{pmatrix}$, $\mathbf{q}_\mathbf{x} := \begin{pmatrix} q_{\mathbf{x}1} \\ q_{\mathbf{x}2} \\ \dots \\ q_{\mathbf{x}K} \end{pmatrix}$ *and* $\mathbf{x}' := \begin{pmatrix} 1 \\ \mathbf{x} \end{pmatrix}$.

Then $\mathbf{B}\mathbf{q}_\mathbf{x} = \mathbf{x}'$ *or, equivalently,*

$$\mathbf{q}_\mathbf{x} = \mathbf{B}^{-1}\mathbf{x}' \tag{5.4}$$

We note that the above solution for $\mathbf{q}_\mathbf{x}$ can produce entries which are outside the interval $[0,1]$. To address this we followed the strategy employed in sNMF [39], and first set for all $\mathbf{x} \in \mathbf{X}$ all entries of $\mathbf{q}_\mathbf{x}$ that are negative to zero. We then divide each entry of $\mathbf{q}_\mathbf{x}$ by the sum of entries of $\mathbf{q}_\mathbf{x}$. This ensures that the values of $\mathbf{q}_\mathbf{x}$ lie in the interval $[0,1]$ and that they also sum to one.

Using Observations 1 and 2, we can optimise Equation (5.1) iteratively (see Algorithm 6, with ε set to 10^{-5}). We found that that algorithm returns accurate estimates of the Q matrix across all simulation scenarios as well as for the two biological datasets under investigation in Section 5.3.3. Additionally, since it is working directly on a reduced-dimension dataset, the matrix operations involved in the algorithm are computed very quickly as opposed to [115], where the whole dataset is considered at each operation.

Algorithm 6 Algorithm used to optimise Equation (5.1)

Input: A data matrix $\mathbf{X}$ as returned by Step I of PSIKO

Output: A matrix $\mathbf{A}$ of founders for $\mathbf{X}$ as well as Q-matrix Q for $\mathbf{X}$, minimising Equation (5.1).

Initialise $\mathbf{A}$ and Q randomly.
$prev = 0$
$cur = \mathcal{L}(\mathbf{A}, Q)$
set ε to a small number, say 10^{-5}
while $|prev - cur| < \varepsilon$ **do**
  estimate Q given $\mathbf{A}$ using Equation 5.4
  estimate $\mathbf{A}$ given Q using Equation 5.3
  $prev = cur$
  $cur = \mathcal{L}(\mathbf{A}, Q)$
end while
return $\mathbf{A}, Q$

With the above in mind, a pseudo-code representation of PSIKO is given in Algorithm 7.

Algorithm 7 PSIKO

Input: A dataset in the form of a SNP matrix $\mathbf{X}$ with accession loci encoded as 2's, 1's and 0's.

Output: The number K of founders, the significant principal components (PCs) and a Q-matrix $Q = (q_{\mathbf{cx}})$ for $\mathbf{X}$, where $\mathbf{c}$ is a founder of $\mathbf{X}$ and $\mathbf{x}$ is a an accession of $\mathbf{X}$.

STEP I (Dimensionality Reduction):
1 : first apply linear kernel-PCA to $\mathbf{X}$ to reduce dimensionality of the dataset and then the Tracy-Widom test for non-zero eigenvalues to infer the number $nComp$ of significant principal components. Finally use those components to compute a $nComp$ dimensional dataset X'
2 : normalize $\mathbf{X}'$ to have zero mean and unit variance
STEP II (Population Structure Inference):
3 : find the vertices (and thus the number K of founders) of the $(nComp-1)$-simplex representing $\mathbf{X}'$ by minimising Equation (5.1)
4 : return K and the matrix Q found in Step 3 and the significant PCs found in line 1

5.3.2 Simulated Datasets and performance measure

In this section, we present an outline of how we generated the various types of datasets underpinning our simulation study for assessing PSIKO's performance. In addition, we also briefly review the Root Mean Squared Error measure which we use as assessment criterion. We start with providing details concerning our simulation study.

Simulated datasets generation We used the command line-based coalescent simulator `msms` [32] to first simulate founder allele frequencies and then used them to simulate admixture proportions and genotypes of admixed individuals. More precisely, we simulated $K = 3, 4, \ldots, 10$ independent, randomly mating populations each of which comprised 100 individuals, where by an *individual* we mean a sequence comprising of L loci evolved over a period of 10,000 generations. More precisely, we simulate K independent demes (populations) with no migration between them over a period of 10,000 generations. Each deme is represented by 100 simulated individuals. After 10,000 generations, all K demes are merged and the coalescent process is allowed to terminate. We simulate a fixed number of segregating sites (SNPs in our case) in each case. Specifically, for $K = 3$ and $13,626$ segregating sites, we used the following `msms` command:

```
msms.jar 300 1 -s 13626 -N 1000 -I 3 100 100 100 -ej 2.5 1 2 -ej
2.5 2 3
```

By modifying the -I flag and adding more -ej flags, this command can be used to simulate an arbitrary number of independent populations. The user is referred to the `msms` manual for more details.

Here, the number of generations is biologically inspired and the number of individuals and the value $K = 3$ is based on [1]. The values we chose for L were 13,262 (which is as in [1]) and, to shed light on to the scalability of PSIKO, also 100,000; 250,000 and 2.5 million. We then used these individuals to calculate founder allele frequencies $f_{k1}, f_{k2}, \ldots, f_{kL}$ for all $1 \leq k \leq K$.

Once obtained, we simulated the genotype of an individual on a locus by locus basis using the following two-step process. For a locus l of an individual i, we first simulated the founder z_l of l by sampling from a multinomial distribution with parameter the admixture proportions for individual i. The admixture proportions were sampled from a Dirichlet distribution and represent the contribution of each founder to the dataset.

Subsequent to this, we simulated the genotype of individual i at locus l by sampling from a multinomial distribution with parameter $f_{z_l l}$ the allele frequency of population z_l at locus l (see Figure 5.1 for a summary of this two-step process).

We repeated this process 1,000 times to obtain an admixed dataset containing 1,000 individuals.

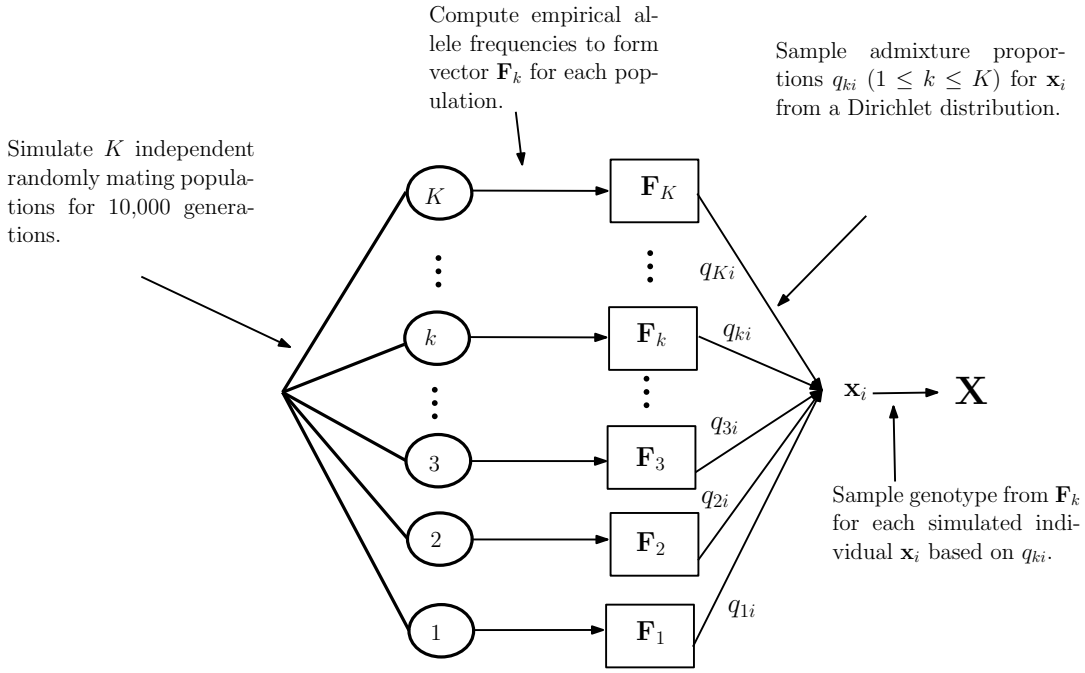


Figure 5.1: A summary of how the datasets underpinning our simulation experiments were generated. Each of the $1,2,\ldots,K$ encircled values indicates a founder population generated with the `msms` software. For all $1 \leq k \leq K$, the vector $\mathbf{F}_k$ represents empirical allele frequencies computed for each of the K founder populations (i. e. $\mathbf{F}_k = (f_{k1}, f_{k2}, \ldots, f_{kL})$) and the values q_{ki} represent the proportion population k contributes to accession $\mathbf{x}_i$ of the dataset $\mathbf{X}$.

Performance measure To assess the performance of the four approaches under consideration with regards to their ability to recover the known Q-matrix underlying a dataset, we used the Root Mean Squared Error *RMSE* between two Q-matrix $\hat{Q}$ and Q', given by:

$$RMSE = \sqrt{\frac{1}{nK}\sum_i \sum_k (\hat{q}_{ik} - q'_{ik})^2} \tag{5.5}$$

where n represents the number of individuals (1,000 in our case) and K represents the number of founders ($K = 3, 4, \ldots, 10$ in our case) and $\hat{q}_{ik}$ and q'_{ik} are the elements of $\hat{Q}$ and Q' respectively, where $1 \leq i \leq n$ and $1 \leq k \leq K$.

Parameter settings For all our simulation experiments we used ADMIXTURE and sNMF with their respective default settings, as suggested by their authors. For STRUCTURE, we used the following settings. We assumed admixed populations with independent allele frequencies. We set the length of the burn-in period to 2,000 iterations and ran the program for an additional 2,000 iterations after the burn-in period. All remaining parameters were used with default values. To ensure fairness in runtime comparison between the above three methods and PSIKO, we only compared their runtimes for the ground truth value of K, thus ensuring that a single run of PSIKO was timed against a single run of all the other methods.

5.3.3 Biological Datasets

To assess the performance of PSIKO with challenging biological datasets, we first performed a comparison of the Q-matrix provided by PSIKO to those estimated using STRUCTURE, ADMIXTURE and sNMF for a set of 84 diverse *Brassica napus* accessions as described in [51]. Over half of these accessions are winter oilseed rape types (OSR; 49), but the rest comprise diverse winter fodder types (5), spring OSR (14), Chinese semi-winter OSR (5), Japanese kale (2), Siberian kale (2) and swede (7). Q-matrix estimations were compared directly and subsequently used to perform linear model association mapping following the method outlined in [51]. Briefly, the Q-matrices were used as covariates in general linear models (GLM), and a mixed linear models (MLM), where a relatedness measure was included as a random effect for two seed oil traits, i. e. erucic acid and glucosinolate content using the program TASSEL [4]. The results of these models were then compared to their P-value expectations. Results were presented as QQ-plots showing observed against expected $\log_{10}$P values for each of the four stratification methods, and each of the seed oil traits and association model types.

To also investigate PSIKO in a human population context, we applied it to a subset of the HapMap Phase 3 dataset [117]. That subset comprised 541 individuals spanning the groupings with the following sampling scenarios. African ancestry in Southwest

USA (ASW), Yoruban in Ibadan, Nigeria, West Africa (YRI), Utah residents with Northern and Western European ancestry from the CEPH collection (CEU) and Mexican ancestry in Los Angeles, California (MEX). Each individual was genotyped over 1,457,897 SNP loci. We remark in passing that the choice of dataset is as in [1] noting though that that paper used an older version of the dataset and that those sequences had been pruned so that each comprised 13,298 genotyped SNP loci [1]. The general understanding of the dataset is that the ASW sample is admixed with ancestries from YRI and CEU and that MEX is admixed with ancestries from CEU and an unsampled founder population [59, 68, 1]. Therefore the number of founders for this dataset is expected to be three.

5.4 Results

Bearing in mind that ADMIXTURE has been shown in [1] to be faster than STRUCTURE, FRAPPE [115] and INSTRUCT [40], and that the recently introduced FASTSTRUCTURE approach [102] has runtime comparable to ADMIXTURE [102], to asses PSIKO's performance we only compared it against ADMIXTURE and SNMF and, due to its popularity, STRUCTURE. For this, we used a computing cluster with Intel Sandybridge Dual processor, 8 core E5-2670 2.6GHz CPU's and 2Gb of DDR3 memory at 1066Mhz, with Intel Hyper-threading disabled. We simulated different scenarios for how populations might have arisen. These simulation studies are similar in spirit to those performed in [1]. Additionally we tested the methods on real biological examples. We start with describing the results of the simulation study which also includes details on the parameters we varied and their ranges. We then present our findings for the biological datasets.

5.4.1 Simulated datasets

We simulated datasets each containing 1,000 individuals, where each individual is assumed to be an admixture of $K \geq 2$ founder populations (see Section 5.3). The parameters we varied were the number K of founders and the respective Dirichlet distribution parameters for them. The values we considered for K ranged from three to ten and our choices for the values for the Dirichlet distribution parameters were inspired by the

values used in [1]. We will explain these choices in more detail below as they depend on the values of K employed and thus necessitate a separate treatment of the cases $K = 3$ and $K \geq 4$. Before detailing these cases though, we remark that low values for the Dirichlet distribution parameters correspond to almost admixture-free populations whereas values close to one correspond to heavily admixed populations. Thus, our simulation study allows us to assess the performances of the methods in question on highly admixed and highly non-admixed populations. We start our discussion with remarking that the value for K was correctly recovered by all tested methods for each of the constructed simulated datasets.

For $K = 3$, and datasets with sequence length 13,626 we chose the same values for the three Dirichlet distribution parameters as in [1], resulting in six different simulation scenarios. Three of these scenarios were asymmetric meaning that in each case at least one Dirichlet distribution parameter was different from the other two and the other three were symmetric meaning that in each case all Dirichlet distribution parameters were the same. For each of the six scenarios we generated 100 datasets, resulting in a total of 600 datasets. These we then analysed with regards to their behaviour under PSIKO (see below), and the average Root Mean Square Error for the Q-matrices found by each of the methods considered, where the average is taken over all 100 datasets of a scenario (see Section 5.3). Furthermore, for each of the three sequence lengths, 100,000; 250,000 and 2.5 million we generated 10 datasets as before using the symmetric Dir(1,1,1) parameter distribution. To assess the effect of SNP-pruning we also generated a further 100 datasets following a similar protocol (see below for details). Additionally, to test PSIKO's robustness to deviations from our simulation model, we also simulate scenarios with noise, missing data and inbreeding present.

Behaviour of a dataset To investigate the behaviour of PSIKO when applied to a dataset generated under each of the six scenarios, we randomly chose one dataset from each. Exploiting the observation that the number of founders of a dataset equals the number of significant principal components found for that dataset in Step I of PSIKO plus one (see e. g. [88]), we depict each chosen dataset in terms of a panel containing a two dimensional coordinate system whose axes are labelled by the two significant principal components found by PSIKO for that dataset (Figure 5.2). For each coordinate system that make up that figure, its footer $Dir(x,y,z)$ encodes the simulation scenario used to generate it in terms of the values x, y, and z for the three Dirichlet

distribution parameters. For example, the footer $Dir(0.2, 0.2, 0.5)$ of the leftmost coordinate system in the bottom row indicates that two out of the three Dirichlet distribution parameters had value 0.2 and that the third one had value 0.5.

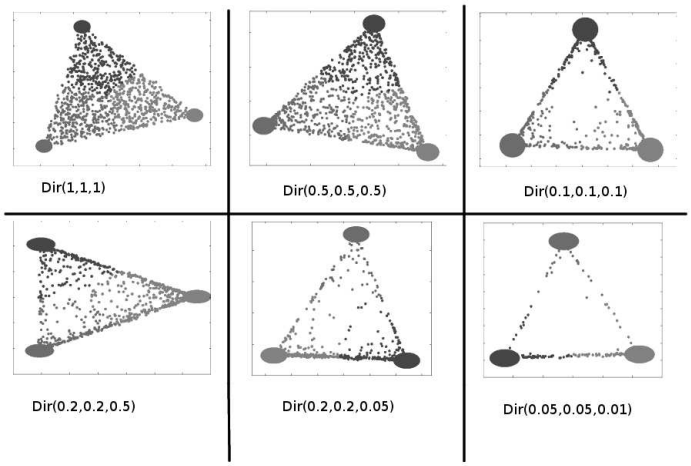


Figure 5.2: PCA reduced dataset under different simulation scenarios, each of which is represented by a separate panel. In each panel, the coordinate axis are the first two significant principal components - see text for details.

As expected (see also [88]), each of the chosen datasets depicted in Figure 5.2 (after having applied PSIKO to them) corresponds to a 2-simplex with the dots inside the simplex representing the dataset's accessions. PSIKO infers three founders for each dataset. We indicated them for each dataset in terms of three ellipses. These are clearly very close to the vertices of the simplex representing that dataset and thus the founders of that dataset. Also the figure suggests that the smaller the values for the Dirichlet distribution parameter are the more the data points get pushed to the simplex's vertices, which is again as expected. This holds not only for the asymmetric scenarios but also for the symmetric ones where the data points get pushed away from the founder with the lowest value. An extreme case in this context is the asymmetric scenario corresponding to $Dir(0.05, 0.05, 0.01)$ as it suggests that one of the founders (i.e. the one corresponding to Dirichlet distribution parameter value 0.01) had very little contribution to the represented dataset.

Average Root Mean Square Error We next turn our attention to assessing the estimated accuracy of PSIKO by measuring the average RMSE between the true and estimated Q-matrices under each one of the six simulation scenarios. For this we used the 600 datasets generated as described above as input to all four methods in question to obtain Q-matrix estimates from each of them. For each method and over all 100 datasets of a scenario we then computed the average RMSE between the true and estimated Q-matrices. A summary of our results in terms of these averages is given in Table 5.2 which consists of six panels each of which corresponds to one of our six simulation scenarios. As can be readily observed, all methods seem to be performing similarly well under all simulation scenarios, with negligible differences between their estimates for the Q-matrices.

	$Dir(0.2,0.2,0.5)$	$Dir(0.2,0.2,0.05)$	$Dir(0.05,0.05,0.01)$
PSIKO	0.008	0.007	0.005
ADMIXTURE	0.008	0.005	0.002
sNMF	0.008	0.005	0.002
STRUCTURE	0.053	0.022	0.021
	$Dir(1,1,1)$	$Dir(0.5,0.5,0.5)$	$Dir(0.1,0.1,0.1)$
PSIKO	0.011	0.009	0.004
ADMIXTURE	0.018	0.01	0.004
sNMF	0.02	0.013	0.005
STRUCTURE	0.015	0.016	0.03

Table 5.2: For $K = 3$, we present the average RMSEs between the true and the estimated Q-matrices for our simulated datasets. -see text for details

Longer Sequences As can be readily observed from Table 5.3 and Figure 5.3, PSIKO is faster than sNMF[1] for each of the three sequence lengths used i. e. 100,000; 250,000 and 2.5 million (see Section 5.3). In fact, as the length of the sequences grows, so too does the difference in run time between PSIKO and sNMF with that difference being significantly in favour of PSIKO. A possible reason for this might be that PSIKO is based on kernel-PCA, which is known to scale very well with the number of variables of a dataset which, in our case, is the number of SNPs i. e. the sequence length (see also Section 5.3). This behaviour seems to suggest that PSIKO

[1]Since it has been shown in [39] that ADMIXTURE is slower than sNMF, we only compared PSIKO against sNMF.

scales better than sNMF with increasing sequence length making it highly attractive for population structure estimation from the very large datasets that are becoming increasingly more common in modern, whole-genome studies.

Sequence Length	100,000	250,000	2,500,000
PSIKO	8s	11s	1m25s
sNMF	55.5s	1m40s	22m28s

Table 5.3: We summarise the relative runtimes of sNMF and PSIKO as averages over all 30 datasets (i. e. 10 datasets for each symmetric Dirichlet distribution parameter setting given in Table 5.2).

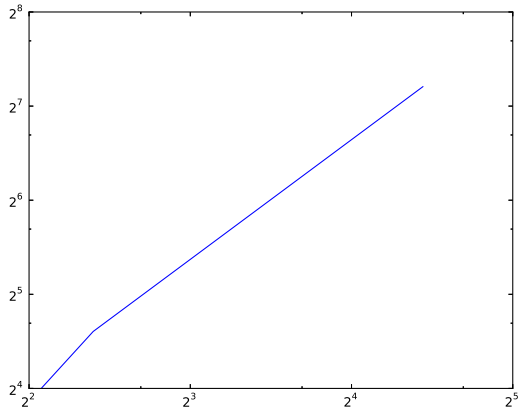


Figure 5.3: Log-log (base e) plot of PSIKO runtimes (x-axis) vs. sNMF runtimes (y-axis), which shows the speed-up provided by PSIKO over sNMF.

SNP-pruning: A popular way to turn a large SNP dataset into a dataset of more manageable size is to employ Linkage Disequilibrium (LD) [99], which is essentially a measure of how frequently SNPs get transmitted together. This technique however has the potential to remove relevant information thus introducing bias to a dataset. To test the robustness of all four methods with regards to this we proceeded as follows. For $K = 3$ we used msms to simulate 100 datasets each comprising 1,000 individuals and 1 million SNPs per individual. From the resulting sequences we then randomly removed

90% of SNPs and then ran PSIKO, ADMIXTURE, and sNMF on the resulting 100 datasets. We found that the average RMSE was below 0.025 for all of PSIKO, sNMF and ADMIXTURE, corresponding to at most a 2.5% error in ancestry estimates. Once again, all of the tested methods correctly inferred K=3. The average runtimes were 3s for PSIKO, 7s for sNMF and 30s for ADMIXTURE.

Larger values of K: Due to the combinatorial explosion caused by asymmetric Dirichlet parameter distributions for increasing values of K, we only considered symmetric Dirichlet distribution parameters for higher values of K, that is, for K ranging between four and ten. For each of these values for K, we chose the same values for the Dirichlet distribution parameters as for the symmetric Dirichlet distribution parameters for $K = 3$ i.e. all 1, all 0.5 and all 0.1.

We found that the performance of each of the methods is comparable for all of the resulting 2,100 datasets (see Table 5.4). It is worth noting though that the runtime of PSIKO is much faster than that of ADMIXTURE (and hence also STRUCTURE), and slightly faster than that of sNMF, with sNMF taking on average 7 seconds to complete processing each dataset, PSIKO taking on average 4s to complete, and ADMIXTURE taking on average 55s to complete.

Dirichlet parameters	1	0.5	0.1
$K = 4$	0.013	0.009	0.007
$K = 5$	0.013	0.01	0.01
$K = 6$	0.015	0.01	0.01
$K = 7$	0.015	0.011	0.011
$K = 8$	0.015	0.012	0.012
$K = 9$	0.016	0.013	0.013
$K = 10$	0.016	0.013	0.01

Table 5.4: Denoting the Dirichlet distribution parameter settings of all 1s, all 0.5s and all 0.1s, by 1, 0.5 and 0.1 respectively and using the latter as column labels, we present the average RMSE between the true and the estimated Q-matrix for PSIKO for the values $K = 4, \ldots, 10$.

Noise Due to the possibility of complex evolutionary processes such as hybridization having confounded the coalescent signal in a dataset, we also tested the robust-

ness of PSIKO for noisy datasets. These we obtained by employing a parameter p that governs the amount of noise that we allowed a dataset's sequences to contain. More precisely, we started with a dataset obtained for $K = 3$ and Dirichlet distribution parameters $Dir(1,1,1)$ (see Section 5.3 for details), and then, for every one of its sequences, flipped on a locus by locus basis the allele of that locus with probability p. Using this modification process we generated 100 noisy datasets for 1,000 accessions at 13,262 loci with noise level p set to 0.01, 0.05, 0.1 and 0.15, corresponding to 1%, 5%, 10% and 15% noise respectively.

As can be readily seen, the difference in the average RMSE between the estimated and true Q-matrix for each approach in question under each of the aforementioned noise level is marginal (Table 5.5) suggesting that all methods are equally robust under the considered simulation scenarios with the observed differences being marginal.

p	0.01	0.05	0.1	0.15
PSIKO	0.011	0.012	0.013	0.015
sNMF	0.016	0.012	0.012	0.02
ADMIXTURE	0.018	0.013	0.013	0.019

Table 5.5: Average RMSE between the true and estimated Q-matrix for $Dir(1,1,1)$ for each approach under each noise level p.

Missing data Reflecting the fact that even with current NGS technology, missing data is still a problem [51], we also assessed the robustness of PSIKO for this type of data. To obtain such datasets, we proceeded as in the previous data experiment only now instead of flipping a locus allele state with probability p, we set it to a missing value character with probability p. More precisely, for $K = 3$ and Dirichlet distribution parameters $Dir(1,1,1)$, we generated 100 datasets for 1,000 accessions each of which 13,262 loci long (see Section 5.3). We set the missing value character probability p to 0.1 and 0.2, corresponding to 10% and 20% missing data, respectively. Using again the average RMSE as assessment criterion, we present our findings in Table 5.6.

As can be readily seen, even with large proportions of data missing all three methods perform equally well with only marginal differences, a fact that was observed for sNMF and ADMIXTURE also in [39].

p	0.1	0.2
PSIKO	0.012	0.012
sNMF	0.013	0.012
ADMIXTURE	0.019	0.021

Table 5.6: Average RMSE between the true and estimated Q-matrix for $Dir(1,1,1)$ for each approach under each missing value probability character p.

Inbreeding The assumption of random mating is frequently violated in natural populations. To test the robustness of PSIKO under these circumstances, we also simulated datasets where inbreeding is present. To do this, we first simulated $K = 3$ independently mating populations as in the noise experiment. For each population $1 \leq k \leq 3$ and each locus l in such a population, we then computed the empirical allele frequencies f_{kl} (see Section 5.3). Subsequent to this and following [39], we used a pre-set value for the *inbreeding coefficient* F_{IS} (i.e. $F_{IS} = 0.25$ and $F_{IS} = 1$) to compute genotype frequencies g_{kl} at locus l in population k. Using the Dirichlet distribution parameters $Dir(1,1,1)$, we then applied the same simulation protocol as above (see Section 5.3.2 for details), with g_{kl} taking the place of f_{kl}. For each value of F_{IS}, we simulated 100 datasets comprising 1,000 individuals each with 13,262 genotyped SNP positions.

As can be seen (Table 5.7), all methods seem to perform well when inbreeding is present in the dataset, although PSIKO seems to be slightly more accurate than sNMF and ADMIXTURE (see also [39] where a similar trend was observed for sNMF and ADMIXTURE).

F_{IS}	0.25	1
PSIKO	0.016	0.017
sNMF	0.026	0.027
ADMIXTURE	0.022	0.026

Table 5.7: Average RMSE between the true and estimated Q-matrix for $Dir(1,1,1)$ for each approach under each value for the inbreeding coefficient F_{IS}.

5.4.2 Biological datasets

In order to further assess PSIKO, we also subjected it to the test of two biological datasets, one of which is an oilseed rape dataset that was originally studied in [51], and the other is from the HapMap 3 project (see Section 5.3 for a brief description of each). We compared our findings with that of ADMIXTURE and sNMF, again using the average RMSE as an assessment measure.

Oilseed rape dataset Two of the four methods tested predicted two population clusters (i. e. $K = 2$). ADMIXTURE predicted three population clusters, while sNMF predicted five clusters. For the purposes of comparing the four models equally, we elected to use the Q-matrices generated for $K = 2$ from each of the programs. Similarly and as recommended by their respective authors, we ran all programs with their default parameter values. Additionally, we ran STRUCTURE with a burn-in period of twenty thousand iterations, followed by another twenty thousand iterations. Direct comparison of the four obtained Q-matrices (Figure 5.4) indicate great similarity, particularly between ADMIXTURE, sNMF and PSIKO.

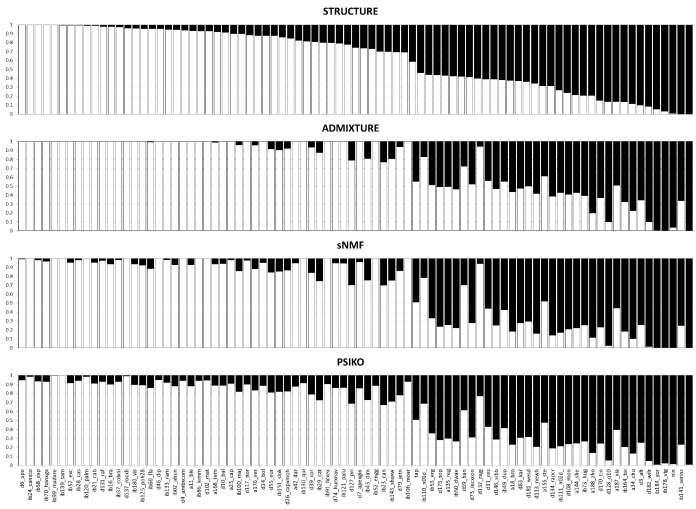


Figure 5.4: Q-matrix plots for the 84 line *Brassica napus* dataset comparing the performance of PSIKO to other leading methods. The proportion of alleles belonging to each of the clusters is shown by respective white bars (cluster 1) or black bars (cluster 2).

The results of the association mapping using each of the four matrices were very similar (Figure 5.5). As expected, incorporating the relatedness matrix as a random effect in a mixed linear model (MLM) reduced the supposed Type I error rate. For the erucic acid trait, the residual error was minimised by the MLM/STRUCTURE model, and for the seed glucosinolates trait the residual error was minimised by the MLM/PSIKO model. It is worth noting, however, that the difference between the Q-matrices was not enough to alter identification of markers in close proximity to the major causative loci (see [51] for details).

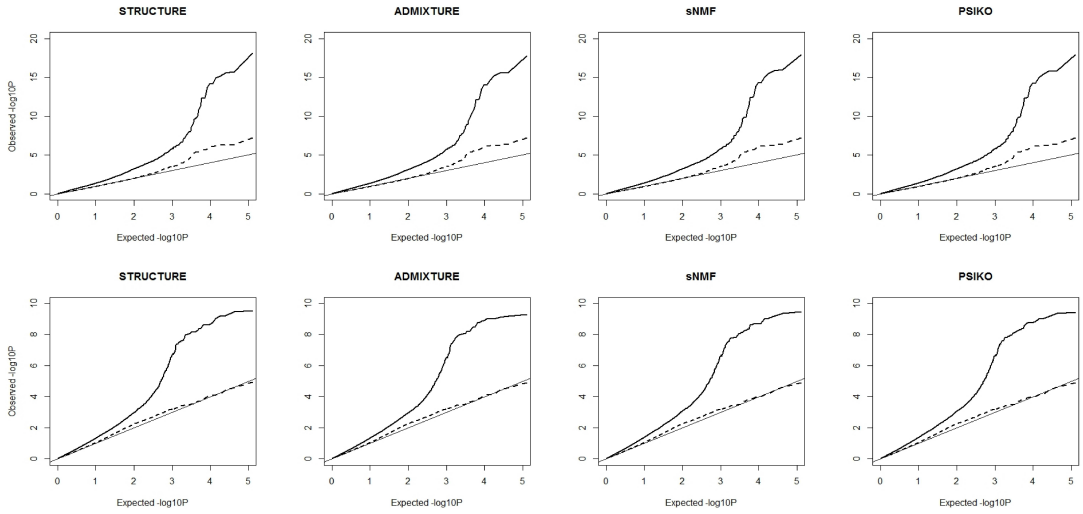


Figure 5.5: QQ-plots illustrating population structure corrections using the four methods in GWAS analysis of two traits in the 84 lines *Brassica napus* panel, erucic acid (A) and seed glucosinolate content (B). The expected $-log_{10}P$ (x-axis) are plotted against those observed (y-axis) from either a general linear model (solid lines) using population structure correction only, and a mixed linear model (dashed lines) with population structure and relatedness corrections. The diagonal line is a guide for the perfect fit to the expected $-log_{10}P$ values.

HapMap3 dataset Given the size of the dataset and thus the prohibitively long runtime of STRUCTURE, we only investigated it with ADMIXTURE, sNMF, and PSIKO (again with all parameter values set to default). Since there is no trait data available, we measured the difference between any two of the three returned Q-matrices

in terms of their RMSE and their R^2 correlation coefficient.

Given the widely accepted fact that the number of founders for this particular dataset is three, all three methods were run with $K = 3$. They all found strikingly similar Q-matrices. More precisely the RMSE between any two matrices was never larger than 0.02 (corresponding to about 2% difference) and the R^2 correlation coefficient was always larger than 0.99 (suggesting that they are almost perfectly correlated). However there was a discrepancy between the methods with regards to estimating the number of founders for the dataset with PSIKO and ADMIXTURE returning $K = 3$ whereas and sNMF returned $K = 4$. The very fast runtime of $48s$ for PSIKO (as compared to ADMIXTURE whose runtime was $5212s$ equating to around $1h$ and $27min$ and sNMF whose runtime was $17min$ and $18s$) is strikingly apparent with this large-scale dataset.

Since mapping information is available for this dataset which can be used for LD based SNP-pruning purposes, we also investigated the performance of PSIKO, sNMF, and ADMIXTURE when the sequences are pruned. More precisely we used the sliding window based SNP-pruning approach implemented in PLINK [99] (with default settings) to obtain a pruning of the HapMap3 dataset. We found that PSIKO, sNMF, and Admixture all correctly infer the widely accepted number of three founders for that dataset, and that the RMSE between any pair of estimated Q-matrices is never greater than 0.02 (i.e. a 2% disagreement), suggesting that all tested methods yield very similar results (data not shown). However, PSIKO took 21 seconds to complete. Using $K = 3$ as input, sNMF took 6 minutes and ADMIXTURE took 36 minutes. Additionally, we found that the SNP pruning took 52 minutes to terminate resulting in a 52 min overhead in the total running time of each method for this experiment. This is in stark contrast to the 48 seconds it took PSIKO to analyse the complete, unpruned dataset.

5.5 Discussion

Population structure is a confounding factor in population association studies, hampering our understanding of how, for example, agronomically important traits have been selected for in crop plants or how diseases might have spread throughout a population [95]. It is therefore important to be able to correct for it and this entails gaining in-

sight into a dataset's Q-matrix as well as the number of its founders. Popular software packages such as STRUCTURE, FRAPPE, INSTRUCT and ADMIXTURE infer both. Many of them are based on sophisticated models and rely on assumptions such as satisfying Hardy Weinberg Equilibrium. However if the dataset in question violates such assumptions or is very large, as would be the case for NGS datasets, these approaches tend to suffer from long runtimes. To address, among others, the issue of scalability the sNMF approach has been proposed [39]. Unlike STRUCTURE and ADMIXTURE, it is not model-based and uses sophisticated algorithmic techniques to ensure fast run-times on large datasets.

Here, we propose the novel and fast PSIKO approach for population structure inference. By combining linear kernel-PCA with a quick-to-solve optimisation problem, it couples the fast runtime and robustness of PCA with the biological interpretability of Q-matrices obtained from model-based approaches such as STRUCTURE and ADMIXTURE. This allows quick estimation of the Q-matrix underpinning a marker dataset as well as the number of founders of that dataset. Due to PCA's few underlying assumptions, PSIKO is widely applicable and generally has a very low run time, at the same time producing results that are comparable in quality with those obtained by ADMIXTURE, STRUCTURE and sNMF.

In order to assess the performance of PSIKO with regards to Q-matrix estimation and inference of founder number, we rigorously tested it on both simulated and real biological datasets. In our simulation studies, we varied the number of founders for a dataset as well as the admixture scenarios for generating a dataset. To help ensure biological relevance, we based our choices for the range of these parameters on those made in [1]. Across a wide range of cases of admixture, we found that PSIKO provides Q-matrix estimates that are very close to the estimates for the respective datasets produced by STRUCTURE, ADMIXTURE and sNMF where closeness is measured in terms of the Root Mean Squared Error between two matrices [1]. Our missing data, noise, and inbreeding experiments suggest that PSIKO as well as ADMIXTURE and sNMF handle these types of data extremely well. However for large datasets PSIKO seems to be superior, even if such a dataset is pruned based on e. g. linkage disequilibrium.

The first of our biological datasets comprises 84 oilseed rape accessions, representing some seven crop types, genotyped over 101,644 SNP loci. The second comprises

541 human samples from differing geographic regions, genotyped at 1,457,897 SNP loci. For each dataset, we found that the Q-matrix estimates generated by PSIKO were very close to those produced by ADMIXTURE and sNMF for that dataset, using the same measure of closeness as in our simulation study. However, it is worth pointing out that independent of whether the dataset had been pruned or not, PSIKO's runtime was only a fraction of that of ADMIXTURE, especially on the human dataset, and was also considerably faster than sNMF.

Although great effort has been put into the development of powerful tools for deriving the number K of founders of a population dataset, inferring that number is still a formidable statistical and computational problem. For example, finding that number using STRUCTURE can be a very computationally intensive task due to the fact that it has to be run on a range of different values for K each of which might require a lot of computational resources. Even for newer methods such as ADMIXTURE or sNMF, finding the optimal value of K relies on running the methods for a range of values of K. In PSIKO, we exploit the behaviour of the eigenvalues returned by linear-kernel PCA for a dataset to infer K. Due to the algorithmic internals of PCA this can be done quickly. We are also motivated by a study in [88] as well as numerous simulation studies which indicate that the number of founders of a dataset equals the number of significant principal components for that dataset plus one. Our simulation studies as well as our two real biological examples suggest that PSIKO holds great promise for this.

The speed of PSIKO is similar to that of sNMF for smaller datasets, and is faster than that of ADMIXTURE. While for small datasets the differences in speed between PSIKO and sNMF are negligible, with increasing sequence length PSIKO proves to be significantly faster than sNMF and implicitly also ADMIXTURE. We therefore argue that PSIKO could be a very attractive tool for analysing the larger datasets that arise from NGS technologies. For smaller datasets ($< 50K$ SNPs), the differences between the three methods are not as clear-cut, and the user should choose whichever method would suit their particular dataset best.

In summary we propose a novel, non-model-based method for inferring population structure. It exploits the advantages of linear kernel-PCA to quickly and accurately describe a SNP dataset's population stratification. It is much (up to 300 times) faster than classical, model-based approaches whilst its outputs match those of state-of-the-

art methods such as sNMF. Its superior speed for large data sets makes it particularly attractive for datasets generated by NGS approaches.

5.6 Acknowledgements

A. -A. Popescu thanks the Norwich Research Park (NRP) for support. The research presented in this chapter was carried out on the High Performance Computing Cluster supported by the Research and Specialist Computing Support service at the University of East Anglia. The authors would like to thank the referees for their helpful suggestions and comments.

Chapter 6

PSIKO2: a fast and versatile tool to infer population stratification on various levels in GWAS

This chapter is based on

A-A. Popescu, K.T. Huber. PSIKO2: a fast and versatile tool to infer population stratification on various levels in GWAS *Bioinformatics*, 31: Epub, 2015.

A-A. Popescu's contribution is the development, implementation, testing and comparative benchmarking of the new PSIKO2 LAI algorithm. He also contributed to the writing of the first draft of the paper.

6.1 Chapter Summary

In this chapter we present an extension of PSIKO. This extension involves enhancing PSIKO to not only infer global-level ancestry (in the form of a Q-matrix), but also localised ancestry, in the form of genomic origins of smaller genomic regions. This is especially important when dealing with recently admixed individuals, for which global ancestry might provide too rough a picture to provide significant insight.

Availability: Source code, binaries, and user manual are freely available at https://www.uea.ac.uk/computing/psiko.

6.2 Introduction

Genome-Wide Association Studies (GWAS) are an invaluable tool for identifying genotypic loci linked with agriculturally important traits or certain diseases. The signal on which such studies rely upon can however be obscured by population stratification, that is reproductive isolation of a sampled population, making it necessary to account for it in some way. A powerful way to do this is to assume that the genotype of each individual (generally called an accession and represented in terms of a SNP sequence) in a study is an admixture of genotypes of $K \geq 2$ (generally unknown) founder (populations). This admixture can then be expressed in terms of a dataset's principal components (PCs) or its population stratification matrix (i. e. its Q-matrix) which indicates for each accession of a study the proportion of its genotype that came from each of the K founders. Contrary to leading tools such as EIGENSTRAT [95] which only infers a dataset's PCs and STRUCTURE [98] (and its extension to FASTSTRUCTURE [102]), ADMIXTURE [1], and SNMF [39] which only infer a dataset's Q-matrix, PSIKO [92] is able to infer both. Furthermore, comparison of PSIKO against competing methods suggest that whilst the quality of its Q-matrices is on par with those generated by them, PSIKO has better scaling properties. However, until now PSIKO could not be used for local ancestry inference (LAI), which is the inference of the ancestry of small regions of the genome. LAI is important for applications ranging from human population studies to identification of disease causative loci [5]. Such information is preferable in cases of recent admixture, since genome-level ancestry can be too coarse to properly account for processes acting on small segments of a genome. Furthermore PSIKO could only be used on a LINUX platform and the efficiency of its PCA-step was not benchmarked. PSIKO2 addresses these shortcomings.

This chapter is structured as follows. We benchmark our implementation against leading alternative approaches to show its speed and correctness. Subsequently we provide details on our LAI algorithm. This includes details about the simulation experiments which we used to assess it and also our findings. To conclude, we briefly talk about implementation and usage of PSIKO2.

6.3 Linear Kernel-PCA Testing and Performance Measure

To ensure the correctness of PSIKO2's predecessor regarding K and Q-matrix estimation, we rigorously tested it in [92] on both real biological and simulated data. This testing did not include scrutinizing the efficiency of our implementation of linear-kernel PCA. We rectify this here by comparing PSIKO2 against an implementation of that strategy in the freely available SKLEARN software [89].

Using datasets generated as in Chapter 5, Section 5.3.2, we next present in Table 6.1 the relative runtimes and memory usage of PSIKO's implementation of linear kernel-PCA as opposed to SKLEARN. Note that SKLEARN uses an SVD-based implementation of PCA.

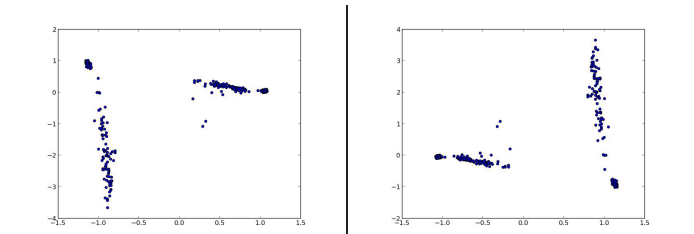
	100K	250K	1M	2.5M
PSIKO	4.0/36	10.5/36	35/833	75/1192
SKLEARN	71.4/2436	337.9/10284	1415/39891	NA / >63000

Table 6.1: Runtime comparison (in seconds)/memory consumption (in megabytes) of PSIKO and SKLEARN's SVD based PCA implementation. These differences show that PSIKO's bit representation strategy provides improved performance over the classical PCA where the input data is represented as numbers.

Using the R^2 correlation coefficient between the PCA-reduced datasets generated by the two kernel-PCA methods as assessment measure, we found R^2 to be larger that 0.999 for all simulated datasets indicating that both methods produce, to all extends and purposes, identical output with differences attributable to floating point arithmetic precision errors. However PSIKO's runtime was a fraction of that of SKLEARN (see Table 6.1), with SKLEARN running out of memory for sequences of length 2.5M.

To further test PSIKO2 we also compared it against SKLEARN on a real biological dataset. For this we used the subset of the HapMap3 dataset already mentioned in Section 6.4.2, and required both approaches to reduce the dataset to two principal components. We found that apart from a rotational difference (attributable to a difference in sign), both methods produce the same output. Indeed the R^2 correlation coefficient for the two resulting reduced datasets is greater than 0.9999, indicating identical output. All runs were realised on a computer having Intel Westmere Dual processor, six

core X5650 2.66GHz CPU, with 48GB of memory and up to 63GB of swap space.



sklearn	PSIKO

Table 6.2: Linear-kernel PCA reduced datasets for the considered HapMap3 dataset as produced by sklearn (left) and PSIKO (right).

6.4 Local Ancestry Inference

In this section, we focus on describing and assessing the performance of LAI as implemented in PSIKO2. We start with describing the LAI algorithm which is based on a sliding window approach and closely related to [5]. We conclude the section by describing the simulation scenario we employed in order to test how PSIKO2 performs.

6.4.1 Sliding window approach

We now describe the sliding window approach that we used for LAI. To explain its underpinning, suppose we are given the number K of founders of a dataset X comprising a set $\mathbf{P}$ of pure (non-admixed) individuals and a set $\mathbf{A}$ of admixed individuals. Let $\mathbf{P}^k$ denote the pure individuals with founder k, $1 \leq k \leq K$. Then application of linear-kernel PCA to X yields the PCs of X, which we store in a matrix $\mathbf{W}$. Consider a window w of length l, spanning a contiguous region of a genotype. We wish to infer the founder of all individuals of X in that window. For this, we denote by P_w and A_w the genotypes of pure and admixed individuals, respectively, at window w. Similarly we denote by $\mathbf{W}_w$ the values of the principal components at window w.

Then the above LAI task boils down to choosing for each window w an integer $1 \leq k \leq K$ as follows. Denote by P_w^k the projection of the pure accessions with founder k onto $\mathbf{W}_w$. Also denote by $\mathbf{A}_w^{'}$ the projection of A_w onto $\mathbf{W}_w$. Let $\mathbf{A}_w^{'}(i)$ denote the projection of individual i onto $\mathbf{W}_w$. Next, we find the mean μ_w^k and variance Σ_w^k of

P_w^k. We then compute the probability of $\mathbf{A}_w^{'}(i)$ under each of the normal distributions, $\mathcal{N}(\mu_w^k, \Sigma_w^k)$, and choose the founder of individual i at window w to be the founder that maximises this probability. More precisely, denoting $o_w(i)$ the founder of window w of individual i, we have:

$$o_w(i) = \underset{k}{\operatorname{argmax}} P(\mathbf{A}_w^{'}(i) | \mu_w^k, \Sigma_w^k) \tag{6.1}$$

Combining this sliding window approach with information about founder genotypes allows us to map, for each individual of a dataset, each such window to one of the K founders. We remark in passing that the window size can be chosen by the user and that the mapping is closely related to that employed by PCADMIX [5]. Contrary to PCADMIX which requires information about founder genotypes as input and thus cannot be used in its absence, this input is optional for PSIKO2. For datasets where this information is not available we infer it from the estimate its Q-matrix.

6.4.2 Assessment of PSIKO2's LAI

In order to assess PSIKO2's ability to infer local ancestry, we generated simulated datasets in a manner similar to [5]. Namely, using some form of K ancestral founders, we simulated a set of individuals whose genotypes, spanning $L \geq 1$ loci, are an admixture of these K pools. These founders were chosen based on [5], comprising three populations from the HapMap3 [117] dataset, namely Utah individuals of European descent (CEU) , Yoruba individuals from Nigeria (YRI) and Han Chinese and Japanese (CHB-JPT). We next describe how we simulated an individual's genotype and then outline our findings.

Genotype Simulation: To simulate an individual's genotype, we first simulate recombination breakpoints by sampling from a Poisson process, in a manner similar to [5]. These breakpoints were then used to split the L loci of an individual into contiguous regions. To each of these regions we first assigned uniformly at random a founder, and then sampled a region's loci from the chosen founder. See Figure 6.1 for a graphical representation. This setup not only provides realistic simulated genotypes, but also readily provides the true ancestry values for each locus for reporting accuracy and true founder genotypes and, thus, to assess PSIKO2's performance when such data is available.

Using the above outline, we generated 100 datasets spanning 20000 loci. Each of the three founders (i. e. CEU, YRI and CHB-JPT) contributed 100 individuals to the dataset, and an additional 300 admixed individuals were simulated per dataset. Ideally we would have liked to compare our results against those of PCADMIX. However for reasons unknown to the authors, PCADMIX would not run on the generated datasets, despite all other methods being considered (sNMF, ADMIXTURE, PSIKO) having no problems with it. [1] Owing to the lack of availability of source code for PCADMIX, we did not investigate this issue further. Instead we rely on the results reported in [5, Figure 5] to compare relative performance.

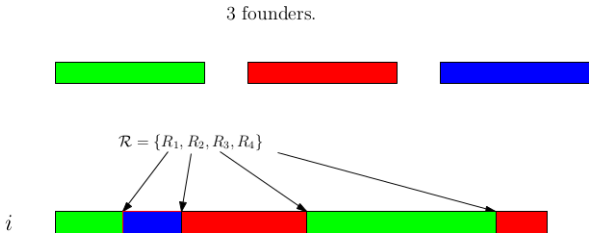


Figure 6.1: For 3 founders indicated by different coloured rectangles and a set $\mathcal{R} = \{R_1, R_2, R_3, R_4\}$ of 4 recombination breakpoints we depict the splitting of an individual i's genome into several regions, each with its own founder as depicted by the colours. The numbers R_1, R_2, R_3, R_4 are drawn from a Poisson process representing the location of the SNPs where breakpoints occur.

Assessment To assess PSIKO2's suitability for LAI, we considered two main scenarios. In the first we provided PSIKO2 with founder and thus our results are directly comparable with those reported for PCADMIX in [5]. In the second, we withheld that information rendering PCADMIX inapplicable as it requires that information as input. Using the Q-matrix estimated by PSIKO2 and taking as proxy for the dataset's founders all accessions which had more than 91% of their genome originating from the same population, this dataset did not pose a problem for PSIKO2.

In both cases, we found the performance of PSIKO2 to be notable with it correctly reporting, within less than a second, the ancestry of 91.2% (first scenario) and 91.1% (second scenario) of the loci under consideration for the input dataset. This is of the same quality as the results that PCADMIX obtained for a dataset with similarly diverged

[1] This was due to it pruning all of the markers under consideration, despite being run with the option of not performing any marker pruning.

founders.

Although conceptually simpler than the approach used by [5], we found PSIKO2 to produce accurate estimates for local ancestry of admixed individuals. Combined with the fact that it does not require the identification of pure individuals whereas e. g. PCAadmix does, this suggests that PSIKO2 holds promise for inferring local ancestry in NGS datasets.

We conclude this chapter with some remarks concerning PSIKO2's implementation and usage.

6.5 Implementation and Usage

Released under a GPL license, PSIKO2 is command-line based and takes as input a genotype matrix in the form of the widely used .geno file format [95]. It is written in C++ and comes with directly linked binary executable files that should work on all modern Linux platforms/Mac environments. Alternatively, users may compile the program themselves if all required libraries are present on their system (see user manual for details). Its output can be used to either inform a study in terms of a dataset's local ancestry (LAI), PCs, K, and Q-matrix (which can then be graphically represented in terms of a barplot using R [100]) or as input to approaches such as STRUCTURE (in the form of e. g. a starting point for estimation of that number), or packages for association mapping such as TASSEL [4], BOLT-LMM [71], and FASTLMM [70].

Chapter 7

Conclusion and Future Work

NGS data promises to revolutionise the field of biology and related areas such as medicine, offering unprecedented amounts of data on which scientists can build our understanding of life. Hand in hand with such data however, come computational challenges that can hinder our ability to make full use of this data. Owing to this, this thesis centres on developing novel methodology that attempts to address some of the many challenges posed by NGS data. The focus lies on phylogenetics and genome-wide association studies, and we first briefly put into context the work that has been carried out in both areas, and then outline our results. As part of this, we also suggest potential avenues of future research that can build on the current work.

Reflecting the two areas of focus, the thesis is in two parts. In the first part we develop novel theory to enable phylogenetic reconstruction methods to better cope with NGS data. Owing to the sheer volume of data that NGS provides, carrying out phylogenetic inference from distance data is attractive due to the speed of such methods. Their input consists of pairwise dissimilarity between these species. While NGS has the advantage of providing large amounts of raw data from which such pairwise distances can be constructed, noise or unreliability of certain reads may render distances values too noisy to consider. Ignoring these distances has been shown to introduce bias [55]. It is therefore important to develop novel approaches that do not suffer from this problem. Motivated by this, in Chapter 3 (which is based on [57]), we develop theoretical foundations for when a set of incomplete distances uniquely determines a phylogenetic tree (see also [60]).

In Chapter 4 (which is based on [91]), we discuss the implementation of several

well known algorithms for phylogenetic inference from incomplete data [17] in ape, a popular R open source phylogenetics package [86].

The second part of the thesis is concerned with novel methodology for inferring population history from NGS data. This is an important problem not only because it can shed light into the evolutionary past of a dataset under consideration, but also because knowledge of such a history can help strengthen methods for finding links between genotype (raw DNA data) and phenotype (observed traits). While algorithms for tackling this problem have been around for quite some time [98], their development pre-dates the advent of NGS data. This in turn causes such methods to struggle when presented with such data, due to e. g. violation of model assumptions or increase in runtime due to the volume of data they are presented with. Therefore, the problem of developing algorithms that can cope with NGS data when correcting for population structure is topical.

In Chapter 5 (which is based on [92]), we introduce PSIKO a novel algorithm which seems to be highly efficient for untangling population structure from NGS datasets. Simulations as well as applications to real biological data suggest it is one of the most scalable and accurate methods for this problem to date, making it very attractive for population genetic studies on NGS data (see also [92]).

In Chapter 6 (which is based on [93]), we present PSIKO2, an extension of PSIKO that has the ability to infer, in addition to global ancestry (population structure), local ancestry from NGS data, which is especially useful when dealing with recently admixed populations.

7.1 Results and Potential Future Directions

In this section, we outline avenues of potential future work. We begin with phylogenetic tree reconstruction from incomplete distances. Subsequent to this, we suggest potential future work pertaining to population genetics.

7.1.1 Incomplete distance construction

In Chapter 3, we presented novel theoretical work for tackling the problem of uniquely determining phylogenetic trees from incomplete distances. While in general this prob-

lem seems to be computationally hard, for a special type of phylogenetic trees, called equidistant trees, we found it to have a polynomial-time solution described in terms of a so-called child-edge graph. We show that the extent to which an equidistant tree is captured by a set of incomplete distances forms a spectrum on the connectedness of the child-edge graph, with well determined trees corresponding to well connected child-edge graphs.

An obvious avenue of future research is the application of our insights in the development of algorithms which can be used to solve practical problems. Indeed, such an avenue is practical, as evidenced in [60]. There, `Lasso`, an algorithm that makes use of the theory developed in Chapter 3 is introduced. That algorithm has been shown to work well on real biological NGS data from a wheat dataset. Despite this dataset's complicated, noisy and cryptic nature, `Lasso` is able to produce strikingly useful results from it, shedding some light into the evolutionary past of this dataset. Similar to `Lasso`, other algorithms that make use of our insights could be put forward, allowing for the development of powerful tools that can deal with NGS data when reconstructing phylogenetic trees. Another avenue would be a more theoretical one, that of addressing the questions in the chapter for rooted trees more general than equidistant ones, namely ones for which the equidistance condition does not hold. Also, another direction of future research is considering phylogenetic trees that also allow for degree two labelled interior vertices. It would be of interest to see how our results would carry over to this class of combinatorial objects.

7.1.2 Population structure inference

PSIKO has been shown to produce promising results on a variety of both real and simulated datasets. Making use of a popular data-mining technique known as linear kernel-PCA and combining a novel iterative least squares optimisation algorithm, PSIKO shows great promise when applied to both real and simulated data, with it being (to the best of our knowledge), one of the fastest tools for carrying out population structure inference to date. We also extend PSIKO to PSIKO2, which is able to return estimates for local ancestry, making use of kernel-PCA and a sliding windows approach. Together with this extension to local ancestry inference, PSIKO2 promises to be a cutting edge tool for population structure inference.

While PSIKO2 returns information regarding both global and local ancestry, it currently does not return information about the actual founders of a dataset. This can be in the form of estimated allele frequencies for each founder or estimated divergence between founders. In population-genetic studies, such information can be very useful. Owing to its remarkable ability to quickly and efficiently deal with large volumes of NGS data, a natural extension of PSIKO would be to adapt it to also return relevant information about the assumed founders of a dataset exhibiting population structure.

Another avenue of future research could be extending the PSIKO algorithm to function on distributed computing clusters. This extension could become very relevant in the future, when NGS data will be such that it becomes infeasible to store on individual commodity computing machines. PSIKO could then be readily applied to such distributed datasets allowing for their efficient analysis.

References

[1] David H. Alexander, John Novembre, and Kenneth Lange. Fast model-based estimation of ancestry in unrelated individuals. *Genome Research*, 19(9):1655–1664, 2009. 46, 48, 49, 95, 105, 108, 109, 119, 123

[2] Thierry Backeljau, Luc De Bruyn, Hans De Wolf, Kurt Jordaens, Stefan Van Dongen, Ron Verhagen, and Birgitta Winnepenninckx. Random amplified polymorphic DNA (rapd) and parsimony methods. *Cladistics*, 11(2):119 – 130, 1995. 2

[3] R. Bernardo. Estimation of coefficient of coancestry using molecular markers in maize. *Theoretical and Applied Genetics*, 85:1055–62, 1993. 5

[4] Peter J. Bradbury, Zhiwu Zhang, Dallas E. Kroon, Terry M. Casstevens, Yogesh Ramdoss, and Edward S. Buckler. Tassel: software for association mapping of complex traits in diverse samples. *Bioinformatics*, 23(19):2633–2635, 2007. 36, 107, 128

[5] Abra Brisbin, Katarzyna Bryc, Jake Byrnes, Fouad Zakharia, Larsson Omberg, Jeremiah Degenhardt, Andrew Reynolds, Harry Ostrer, Jason G Mezey, and Carlos D Bustamante. Pcadmix: principal components-based assignment of ancestry along each chromosome in individuals with admixed ancestry from two or more populations. *Human Biology*, 84(4):343, 2012. 123, 125, 126, 127, 128

[6] TA. Brown. *Genomes. 2nd edition.*, chapter Chapter 16, Molecular Phylogenetics. Oxford, 2002. 1

[7] D. Bryant and V. Moulton. A polynomial time algorithm for constructing the refined Buneman tree. *Applied Mathematics Letters*, 12:51–56, 1999. 88

[8] P. Buneman. *Mathematics in Archaeological and Historical Sciences*, chapter The recovery of trees from measures of dissimilarity, pages 387–395. Edinburgh University Press, 1971. 13

[9] P. Buneman. A note on the metric properties of trees. *Journal of Combinatorial Theory (B)*, 17:48–50, 1974. 16

[10] George Casella and Edward I. George. Explaining the Gibbs Sampler. *The American Statistician*, 46(3):167–174, 1992. 41

[11] Geng Chen and TieLiu Shi. Next-generation sequencing technologies for personalized medicine: promising but challenging. *Science China Life Sciences*, 56(2):101–103, 2013. 2

[12] R. Chenna, H. Sugawara, T. Koike, R. Lopez, T. J. Gibson, D. G. Higgins, and J. D. Thompson. Multiple sequence alignment with the Clustal series of programs. *Nucleic Acids Research*, 31:3497–3500, 2003. 15, 84

[13] B. C. Y. Collard, M. Z. Z. Jahufer, J.B. Brouwer, and E.C.K. Pang. An introduction to markers, quantitative trait loci (QTL) mapping and marker assisted selection for crop improvement: The basic concepts. *Euphytica*, 142:169–196, 2005. 2, 4, 34

[14] 1000 Genomes Project Consortium et al. An integrated map of genetic variation from 1,092 human genomes. *Nature*, 491(7422):56–65, October 2012. 1, 2

[15] ENCODE Project Consortium. The ENCODE (ENCyclopedia of DNA Elements) Project. *Science*, 306(5696):636–640, Oct 2004. 2

[16] A. Criscuolo, V. Berry, EJP Douzery, and O Gascuel. Sdm: a fast distance-based approach for (super)tree building in phylogenomics. *Systematic Biology*, 55:740–755, 2006. 87

[17] A. Criscuolo and O. Gascuel. Fast NJ-like algorithms to deal with incomplete distance matrices. *BMC Bioinformatics*, 9:166, 2008. 3, 17, 24, 31, 32, 85, 87, 90, 130

[18] C. W. Cunningham, K. E. Omland, and T. H. Oakley. Reconstructing ancestral character states: a critical reappraisal. *Trends in Ecology and Evolution*, 13:361–366, 1998. 84

[19] Geert De Soete. Ultrametric tree representations of incomplete dissimilarity data. *Journal of Classification*, 1:235–242, 1984. 22, 32, 57

[20] A. P. Dempster, N. M. Laird, and D. B. Rubin. Maximum likelihood from incomplete data via the EM algorithm. *Journal of the Royal Statistical Society. Series B (Methodological)*, 39(1):1–38, 1977. 39

[21] R. Desper and O. Gascuel. Fast and accurate phylogeny reconstruction algorithms based on the minimum-evolution principle. *Journal of Computational Biology.*, 9:687–705, 2002. 84

[22] M.-M. Deza and I. G. Rosenberg. *n*-semimetrics. *European Journal of Combinatorics*, 21(6):797–806, 2000. 82

[23] R. Diestel. *Graph Theory*. Springer Verlag, Heidelberg, New York, Electronic Edition 2005. 71

[24] Rebecca W. Doerge. Mapping and analysis of quantitative trait loci in experimental populations. *Nature Reviews Genetics*, 3(1):43–52, January 2002. 4

[25] A. Dress, K.T. Huber, J. Koolen, V. Moulton, and A. Spillner. *Basic Phylogenetic Combinatorics*. Cambridge University Press, Cambridge, UK, 2012. 57, 61

[26] Andreas W. M. Dress, K. T. Huber, and M Steel. 'Lassoing' a phylogenetic tree I: basic properties, shellings and covers. *Journal of Mathematical Biology*, 65:77–105, 2012. 25, 26, 27, 28, 29, 31, 57, 58, 59, 79, 80, 82

[27] A.J. Drummond, M.A. Suchard, D. Xie, and A. Rambaut. Bayesian phylogenetics with BEAUti and the BEAST 1.7. *Molecular Biology and Evolution*, 29(8):1969–1973, 2012. 3

[28] R. C. Edgar. MUSCLE: multiple sequence alignment with high accuracy and high throughput. *Nucleic Acids Research*, 32(5):1792–1797, 2004. 15, 84

[29] A.W.F. Edwards and L.L. Cavalli Sforza. The reconstruction of evolution. *Annals of Human Genetics*, 27, 1963. 15

[30] Barbara E. Engelhardt and Matthew Stephens. Analysis of population structure: A unifying framework and novel methods based on sparse factor analysis. *PLoS Genetics*, 6(9):e1001117, 09 2010. 37, 52, 53

[31] G. Evanno, S. Regnaut, and J. Goudet. Detecting the number of clusters of individuals using the software STRUCTURE: a simulation study. *Molecular Ecology*, 14(8):2611–2620, 2005. 46

[32] Gregory Ewing and Joachim Hermisson. MSMS: a coalescent simulation program including recombination, demographic structure and selection at a single locus. *Bioinformatics (Oxford, England)*, 26(16):2064–2065, 2010. 105

[33] Martin Farach, S. Kannan, and Tandy Warnow. A robust model for finding optimal evolutionary trees. *Algorithmica*, 13:155–179, 1996. 4, 31

[34] J.S. Farris, A. G. Kluge, and M. J. Eckardt. A numeric approach to phylogenetic systematics. *Systematic Zoology*, 19:172–189, 1970. 22

[35] F. Felsenstein. *Inferring Phylogenies*. Sinauer Associates, 2003. 82

[36] J Felsenstein. Evolutionary trees from dna sequences: a maximum likelihood approach. *Journal of Molecular Evolution*, 17(6):368–376, 1981. 15

[37] J. Felsenstein. PHYLIP-phylogenetic inference package (version 3.2). *Cladistics*, 5:164–166, 1989. 3

[38] W. Fitch. Networks and viral evolution. *Journal of Molecular Evolution*, 44:65–75, 1997. 9

[39] Eric Fricho, Franois Mathieu, Tho Trouillon, Guillaume Bouchard, and Olivier Franois. Fast inference of admixture coefficients using sparse non-negative matrix factorization algorithms. *Genetics*, 196(4):973–983, 2014. 51, 95, 103, 111, 114, 115, 119, 123

[40] Hong Gao, Scott Williamson, and Carlos D. Bustamante. A Markov chain Monte Carlo approach for joint inference of population structure and inbreeding rates from multilocus genotype data. *Genetics*, 176(3):1635–1651, 2007. 108

[41] 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. 2, 3, 17, 84

[42] O. Gascuel. Concerning the nj algorithm and its unweighted version, unj. *Mathematical Hierarchies and Biology*, page 149170, 1997. 17

[43] O Gascuel. Data model and classification by trees: the minimum variance reduction (mvr) method. *Journal of Classification*, 17:67–99, 2000. 2, 17, 87

[44] O. Gascuel and M. Steel. Neighbor-Joining revealed. *Molecular Biology and Evolution*, 23:1997–2000, 2006. 16, 17

[45] Alan E. Gelfand and Adrian F. M. Smith. Sampling-Based Approaches to Calculating Marginal Densities. *Journal of the American Statistical Association*, 85(410):398–409, 1990. 40, 41

[46] Gene Golub and William Kahan. Calculating the singular values and pseudo-inverse of a matrix. *Journal of the Society for Industrial & Applied Mathematics, Series B: Numerical Analysis*, 2(2):205–224, 1965. 99

[47] Manfred G. Grabherr, Brian J. Haas, Moran Yassour, Joshua Z. Levin, Dawn A. Thompson, Ido Amit, Xian Adiconis, Lin Fan, Raktima Raychowdhury, Qiandong Zeng, Zehua Chen, Evan Mauceli, Nir Hacohen, Andreas Gnirke, Nicholas Rhind, Federica di Palma, Bruce W. Birren, Chad Nusbaum, Kerstin Lindblad-Toh, Nir Friedman, and Aviv Regev. Full-length transcriptome assembly from RNA-Seq data without a reference genome. *Nature Biotechnology*, 29(7):644–652, July 2011. 2

[48] S. Grunewald and K. T. Huber. *Reconstructing Evolution, new mathematical and computational advances*, chapter Identifying and defining trees, pages 217–241. Oxford University Press, 2007. 13

[49] A. Guénoche and B. Leclerc. The triangles method to build X-trees from incomplete distance matrices. *RAIRO Operations Research*, 35:283–300, 2001. 17, 19, 32, 85

[50] Dean H Hamer. Beware the chopsticks gene. *Molecular Psychiatry*, 5(1):11–13, 2000. 35

[51] Andrea L. Harper, Martin Trick, Janet Higgins, Fiona Fraser, Leah Clissold, Rachel Wells, Chie Hattori, Peter Werner, and Ian Bancroft. Associative transcriptomics of traits in the polyploid crop species Brassica *napus*. *Nature Biotechnology*, 30(8):798–802, July 2012. 3, 6, 21, 57, 96, 107, 114, 116, 117

[52] S. Herrmann, K. T. Huber, V. Moulton, and A. Spillner. Recognizing treelike k-dissimilarities. *Journal of Classification*, 29:321–340, 2012. 82

[53] Brendan P Hodkinson and Elizabeth A Grice. Next-generation sequencing: a review of technologies and tools for wound microbiome research. *Advances in Wound Care*, 4(1):50–58, 2015. 2

[54] P Hogeweg and B Hesper. The alignment of sets of sequences and the construction of phyletic trees: an integrated method. *Journal of Molecular Evolution*, 20(2):175–186, 1984. 14

[55] Huateng Huang and L Lacey Knowles. Unforeseen consequences of excluding missing data from next-generation sequences: simulation study of rad sequences. *Systematic Biology*, page syu046, 2014. 3, 129

[56] K. T. Huber and M Steel. Reconstructing fully-resolved trees from triplet cover distances. *arXiv preprint*, 2012. 4, 7, 25, 30, 31, 59

[57] K.T. Huber and A-A. Popescu. Lassoing and corralling rooted phylogenetic trees. *Bulletin of Mathematical Biology*, 75:444–465, 2013. 4, 5, 129

[58] J. P. Huelsenbeck and F. Ronquist. MrBayes: Bayesian inference of phylogenetic trees. *Bioinformatics*, 17(8):754–755, August 2001. 3

[59] Mattias Jakobsson, Sonja W. Scholz, Paul Scheet, J. Raphael Gibbs, Jenna M. VanLiere, Hon-Chung Fung, Zachary A. Szpiech, James H. Degnan, Kai Wang, Rita Guerreiro, Jose M. Bras, Jennifer C. Schymick, Dena G. Hernandez, Bryan J. Traynor, Javier Simon-Sanchez, Mar Matarin, Angela Britton, Joyce van de Leemput, Ian Rafferty, Maja Bucan, Howard M. Cann, John A. Hardy, Noah A. Rosenberg, and Andrew B. Singleton. Genotype, haplotype and copy-number variation in worldwide human populations. *Nature*, 451(7181):998–1003, 2008. 108

[60] 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*, 32:1628–1642, 2015. 3, 129, 131

[61] H. Kim and H. Park. Sparse non-negative matrix factorizations via alternating non-negativity-constrained least squares for microarray data analysis. *Bioinformatics*, 319:1495–1502, 2007. 95

[62] Jingu Kim and Haesun Park. Fast nonnegative matrix factorization: An active-set-like method and comparisons. *SIAM Journal on Scientific Computing*, 33(6):3261–3281, 2011. 52

[63] Sung Kim, Vincent Plagnol, Tina T. Hu, Christopher Toomajian, Richard M. Clark, Stephan Ossowski, Joseph R. Ecker, Detlef Weigel, and Magnus Nordborg. Recombination and linkage disequilibrium in Arabidopsis *thaliana*. *Nature Genetics*, 39(9):1151–1155, 2007. 95

[64] Lynn C Klotz, Ned Komar, Roger L Blanken, and Ralph M Mitchell. Calculation of evolutionary trees from sequence data. *Proceedings of the National Academy of Sciences*, 76(9):4516–4520, 1979. 22

[65] W.C. Knowler, R.C. Williams, D.J. Pettitt, and A.G. Steinberg. Gm3;5,13,14 and type 2 diabetes mellitus: an association in American Indians with genetic admixture. *American Journal of Human Genetics*, 43:520526, 1988. 95

[66] Bruno Leclerc and Vladimir Makarenkov. On some relations between 2-trees and tree metrics. *Discrete Mathematics*, 192(1):223–249, 1998. 85

[67] Rasko Leinonen, Ruth Akhtar, Ewan Birney, Lawrence Bower, Ana Cerdeno-Tárraga, Ying Cheng, Iain Cleland, Nadeem Faruque, Neil Goodgame, Richard Gibson, Gemma Hoad, Mikyung Jang, Nima Pakseresht, Sheila Plaister, Rajesh Radhakrishnan, Kethi Reddy, Siamak Sobhany, Petra T. Hoopen, Robert Vaughan, Vadim Zalunin, and Guy Cochrane. The European Nucleotide Archive. *Nucleic Acids Research*, 39(suppl 1):D28–D31, January 2011. 2

[68] J Z Li, D M Absher, H Tang, A M Southwick, A M Casto, S Ramachandran, H M Cann, G S Barsh, M Feldman, L L Cavalli-Sforza, and R M Myers. Worldwide human relationships inferred from genome-wide patterns of variation. *Science*, 319(5866):1100–1104, 2008. 108

[69] Wen-Hsiung Li. Simple method for constructing phylogenetic trees from distance matrices. *Proceedings of the National Academy of Sciences*, 78(2):1085–1089, 1981. 22

[70] Christoph Lippert, Jennifer Listgarten, Ying Liu, Carl M Kadie, Robert I Davidson, and David Heckerman. FaST linear mixed models for genome-wide association studies. *Nature Methods*, 8:833–835, 2011. 5, 128

[71] Po-Ru Loh, George Tucker, Brendan K Bulik-Sulivan, Bjarni J Vilhjlmsson, Hillary K Finucane, Rany M Salem, Daniel I Chasman, Paul M Ridker, Benjamin Neal, Bonnie Berger, Nick Patterson, and Alkes L Price. Efficient Baysian mixed-model analysis increases association power in large cohorts. *Nature Genetics*, 47:284–290, 2015. 5, 128

[72] Jianzhong Ma and Christopher I. Amos. Principal components analysis of population admixture. *PLoS ONE*, 7(7):e40115, 07 2012. 98, 101

[73] V. Makarenkov. Reconstructing and visualizing phylogenetic trees and reticulation networks. *Bioinformatics*, 17:664–668, 2001. 3, 91

[74] Vladimir Makarenkov and François-Joseph Lapointe. A weighted least-squares approach for inferring phylogenies from incomplete distance matrices. *Bioinformatics*, 20(13):2113–2121, September 2004. 32, 86

[75] J. Marchini, L. Cardon, M. Phillips, and P. Donnelly. The effects of human population structure on large genetic association studies. *Nature Genetics*, 36:512–517, 2004. 95

[76] G. T. Marth, I. Korf, M. D. Yandell, R. T. Yeh, Z. Gu, H. Zakeri, N. O. Stitziel, L. Hillier, P. Y. Kwok, and W. R. Gish. A general approach to single-nucleotide polymorphism discovery. *Nature Genetics*, 23(4):452–456, December 1999. 2

[77] B. Mau, MA. Newton, and B. Larget. Bayesian phylogenetic inference via Markov Chain Monte Carlo methods. *Biometrics*, 55:1–12, 1999. 15

[78] C. A. Meacham. A manual method for character compatibility analysis. *Taxon*, 30:591–600, 1981. 88

[79] C.A. Meacham. *Numerical Taxonomy*, chapter Theoretical and computational considerations of the compatibilty of qualitative taxonomic characters, pages 304–314. Berlin Heidelberg, 1983. 13

[80] Nicholas Metropolis, Arianna W. Rosenbluth, Marshall N. Rosenbluth, Augusta H. Teller, and Edward Teller. Equation of State Calculations by Fast Computing Machines. *The Journal of Chemical Physics*, 21(6):1087–1092, June 1953. 41

[81] William M. Muir, Gane Ka-Shu Wong, Yong Zhang, Jun Wang, Martien A. M. Groenen, Richard P. M. A. Crooijmans, Hendrik-Jan Megens, Huanmin Zhang, Ron Okimoto, Addie Vereijken, Annemieke Jungerius, Gerard A. A. Albers, Cindy Taylor Lawley, Mary E. Delany, Sean MacEachern, and Hans H. Cheng. Genome-wide assessment of worldwide chicken SNP genetic diversity indicates significant absence of rare alleles in commercial breeds. *Proceedings of the National Academy of Sciences*, 105(45):17312–17317, 2008. 57

[82] Kevin P. Murphy. *Machine Learning: A Probabilistic Perspective*. The MIT Press, Cambridge, Massachusetts, 2012. 40, 48, 49, 50, 99

[83] C. Notredame, D. Higgins, and J. Heringa. T-Coffee: a novel method for multiple sequence alignments. *Journal of Molecular Biology*, 302:205–217, 2000. 14, 15, 84

[84] L. Pachter and D. Speyer. Reconstructing trees from subtree weights. *Applied Mathematics Letters*, 17:615–621, 2004. 82

[85] E. Paradis. *Analysis of Phylogenetics and Evolution with R*. Springer, second edition, 2012. 84

[86] E. Paradis, J. Claude, and K. Strimmer. Ape: Analysis of phylogenetics and evolution in R language. *Bioinformatics*, 20:289–290, 2004. 13, 130

[87] A. H. Paterson. *Genome Mapping in Plants*, chapter Mapping genes responsible for differences in phenotype, pages 41–54. Academic Press, 1996. 4

[88] Nick Patterson, Alkes L Price, and David Reich. Population structure and eigenanalysis. *PLoS Genetics*, 2(12), 2006. 97, 98, 101, 109, 110, 120

[89] F. Pedregosa, G. Varoquaux, A. Gramfort, V. Michel, B. Thirion, O. Grisel, M. Blondel, P. Prettenhofer, R. Weiss, V. Dubourg, J. Vanderplas, A. Passos, D. Cournapeau, M. Brucher, M. Perrot, and E. Duchesnay. Scikit-learn: Machine learning in Python. *Journal of Machine Learning Research*, 12:2825–2830, 2011. 124

[90] Pedro R. Peres-Neto, Donald A. Jackson, and Keith M. Somers. How many principal components? stopping rules for determining the number of non-trivial axes revisited. *Computational Statistics and Data Analysis*, 49(4):974 – 997, 2005. 101

[91] A-A. Popescu, K.T. Huber, and E. Paradis. ape 3.0: New tools for distance-based phylogenetics and evolutionary analysis in r. *Bioinformatics*, 28:1536–1537, 2012. 6, 20, 129

[92] Andrei-Alin Popescu, Andrea L. Harper, Martin Trick, Ian Bancroft, and Katharina T. Huber. A novel and fast approach for population structure inference using kernel-pca and optimisation (PSIKO). *Genetics*, 198(4):1421–1431, 2014. 6, 123, 124, 130

[93] Andrei-Alin Popescu and Katharina T. Huber. PSIKO2: a fast and versatile tool to infer population stratification on various levels in GWAS. *Bioinformatics*, July 2015. 6, 130

[94] Adam H. Price. Believe it or not, QTLs are accurate! *Trends in Plant Science*, 11(5):213–216, May 2006. 4

[95] Alkes L. Price, Nick J Patterson, Robert M Plenge, E Weinblatt Michael, Nancy A Shadick, and David Reich. Principal components analysis corrects for stratification in genome-wide association studies. *Nature Genetics*, 38(8):904–909, 2006. 54, 95, 99, 118, 123, 128

[96] R. C. Prim. Shortest connection networks and some generalisations. *Bell System Tech J.*, 36:389–401, 1957. 18

[97] J. K. Pritchard and N. A. Rosenberg. Use of unlinked genetic markers to detect population stratification in association studies. *American Journal of Human Genetics*, 65:220–228, July 1999. 37

[98] Jonathan K. Pritchard, Matthew Stephens, and Peter Donnelly. Inference of population structure using multilocus genotype data. *Genetics*, 155(2):945–959, 2000. 2, 4, 39, 41, 44, 45, 46, 95, 123, 130

[99] Shaun Purcell, Benjamin Neale, Kathe Todd-Brown, Lori Thomas, Manuel A. Ferreira, David Bender, Julian Maller, Pamela Sklar, Paul I. de Bakker, Mark J. Daly, and Pak C. Sham. PLINK: a tool set for whole-genome association and population-based linkage analyses. *American Journal of Human Genetics*, 81(3):559–575, September 2007. 96, 112, 118

[100] R Core Team. *R: A Language and Environment for Statistical Computing*. R Foundation for Statistical Computing, Vienna, Austria, 2014. 128

[101] J. Rafalski and S. Tingey. Genetic diagnostics in plant breed: RAPDs, microsatellites and machines. *Trends in Genetics*, 9:275–280, 1993. 33

[102] Anil Raj, Matthew Stephens, and Jonathan K. Pritchard. fastSTRUCTURE: Variational inference of population structure in large SNP datasets. *Genetics*, 197(2):573–589, 2014. 108, 123

[103] Neil Risch and Kathleen Merikangas. The Future of Genetic Studies of Complex Human Diseases. *Science*, 273(5281):1516–1517, September 1996. 4

[104] Kamel Saadi, Nicola LC Talbot, and Gavin C Cawley. Optimally regularised kernel fisher discriminant classification. *Neural Networks*, 20(7):832–841, 2007. 102

[105] R. K. Saiki, S. Scharf, F. Faloona, K. B. Mullis, G. T. Horn, H. A. Erlich, and N. Arnheim. Enzymatic amplification of beta-globin genomic sequences and restriction site analysis for diagnosis of sickle cell anemia. *Science*, 230(4732):1350–1354, December 1985. 2

[106] N. Saitou and M. Nei. The neighbor-joining method: a new method for reconstructing phylogenetic trees. *Molecular Biology and Evolution*, 4, 1987. 3, 16, 17, 84

[107] F Sanger, S Nicklen, and AR Coulson. DNA sequencing with chain-terminating inhibitors. *Proceedings of The National Academy of Sciences of The United States Of America*, 74(12):5463–5467, 1977. 2

[108] K.P. Schliep. Phangorn: phylogenetic analysis in R. *Bioinformatics*, 27(4):592–593, 2011. 88

[109] Bernhard Scholkopf, Alexander Smola, and Klaus-Robert Mller. Kernel principal component analysis. In *Advances in Kernel Methods - Support Vector Learning*, pages 327–352. MIT Press, 1999. 97

[110] Charles Semple and Mike Steel. *Phylogenetics*, volume 24 of *Oxford Lecture Series in Mathematics and its Applications*. Oxford University Press, Oxford, UK, 2003. 10, 13, 25, 57, 59, 61, 68, 76, 79, 88

[111] Fabian Sievers, Andreas Wilm, David Dineen, Toby J Gibson, Kevin Karplus, Weizhong Li, Rodrigo Lopez, Hamish McWilliam, Michael Remmert, Johannes Söding, Julie D Thompson, and Desmond G Higgins. Fast, scalable generation of high-quality protein multiple sequence alignments using clustal omega. *Molecular Systems Biology*, 7(1), 2011. 15

[112] Robert L Smith. Efficient monte carlo procedures for generating points uniformly distributed over bounded regions. *Operations Research*, 32(6):1296–1308, 1984. 39

[113] C. Stern. The HardyWeinberg law. *Science*, 97:137–138, 1943. 39, 52

[114] D. L. Swofford, G. L. Olson, P. J. Waddell, and D. M. Hills. *Molecular Systematics*, chapter Phylogenetic inference, pages 407–514. Sinauer Associates, 1996. 15

[115] Hua Tang, Jie Peng, Pei Wang, and Neil J. Risch. Estimation of individual admixture: Analytical and study design considerations. *Genetic Epidemiology*, 28(4):289–301, 2005. 48, 102, 103, 108

[116] The International HapMap Consortium. A second generation human haplotype map of over 3.1 million SNPs. *Nature*, 449(7164):851–861, 2007. 95

[117] The International HapMap Consortium. Integrating common and rare genetic variation in diverse human populations. *Nature*, 467(7311):52–58, 2010. 2, 96, 107, 126

[118] Pieter Vos, Rene Hogers, Marjo Bleeker, Martin Reijans, Theo van de Lee, Miranda Hornes, Adrie Friters, Jerina Pot, Johan Paleman, Martin Kuiper, and Marc Zabeau. AFLP: a new technique for DNA fingerprinting. *Nucleic Acids Research*, 23(21):4407–4414, January 1995. 2

[119] Joe H. Ward. Hierarchical grouping to optimize an objective function. *Journal of the American Statistical Association*, 58(301):236–244, 1963. 23

[120] Matthijs J. Warrens. *n*-way metrics. *Journal of Classification*, 27(2):173–190, 2010. 82

[121] CF Jeff Wu. On the convergence properties of the EM algorithm. *The Annals of Statistics*, pages 95–103, 1983. 48

[122] J. Yu, M. Arbelbide, and R. Bernardo. Power of in silico QTL mapping from phenotypic, pedigree, and marker data in a hybrid breeding program. *TAG Theoretical and Applied Genetics*, 110(6):1061–1067, April 2005. 4, 5

[123] Jianming Yu, Gael Pressoir, William H. Briggs, Irie Vroh Bi, Masanori Yamasaki, John F. Doebley, Michael D. McMullen, Brandon S. Gaut, Dahlia M. Nielsen, James B. Holland, Stephen Kresovich, and Edward S. Buckler. A unified mixed-model method for association mapping that accounts for multiple levels of relatedness. *Nature Genetics*, 38(2):203–208, 2006. 5, 35, 36, 57

[124] Chengsong Zhu, Michael Gore, Edward S. Buckler, and Jianming Yu. Status and prospects of association mapping in plants. *The Plant Genome*, 1:5–20, 2008. 1, 2, 4, 35