UNIVERSITY *of* TASMANIA

School of Natural Sciences
Discipline of Mathematics

AN INVESTIGATION INTO RANK BASED APPROACHES FOR INFERRING PHYLOGENIES

by

Joshua Stevenson, BICT/BSc

November 2019

Submitted in partial fulfilment of the requirements for the Degree of
Bachelor of Science with Honours

Supervisors:	Dr Jeremy Sumner
	Assoc. Prof. Michael Charleston
	Prof. Barbara Holland

ABSTRACT

The goal of phylogenetic inference is to find an evolutionary tree which best explains the ancestral history of some set of taxa (species); this is done using some data such as a DNA sequence alignment. *Flattenings*—matrices constructed using site-pattern counts from an alignment—provide a way of identifying 'true' *splits* and hence the true evolutionary tree via the evaluation of their rank. The size of these matrices, exponential in the number of taxa, introduces a computational challenge. This challenge led to the development of so-called *subflattenings*, which exhibit analogous rank properties but have smaller dimensions (quadratic in the number of taxa). The construction of subflattenings involves representation theory and the application of a similarity transformation in which some choices are involved.

We provide some background and a short survey of the literature related to split and rank-based methods for phylogenetic inference. We then explore algebraic concepts involved in the construction of these matrices, providing some proofs and defining (r, c)-subflattenings. We also provide the results of some simulations undertaken to evaluate practical implications of some possible choices in the construction of subflattenings. Finally, we outline some key points of interest for future research.

ACKNOWLEDGEMENTS

I would first like to thank my supervisors Jeremy Sumner, Barbara Holland and Michael Charleston for the guidance and support they provided this year. I am thankful for everything that they—and other teaching staff at UTAS—have taught me throughout my degree.

I would also like to thank my loving partner Bridgette for her endless support and inspiration, my family for giving me confidence to succeed, and my friend Luke for constantly interrupting my work to discuss unrelated mathematics, and putting up with me doing the same in return.

Finally, thanks to the Alexander McAulay Bequest Trust for the funding granted to me this year through the Alexander McAulay Scholarship in Mathematics.

TABLE OF CONTENTS

LIST OF TABLES

LIST OF FIGURES

Introduction

Phylogenetics is an area of biology which incorporates mathematical concepts, from fields like statistics, probability theory and algebra, in the development of models and methods that allow biologists to uncover the evolutionary history of present-day species. This process is called phylogenetic inference. The aim of phylogenetic inference is to effectively and efficiently process biological data—for example, sequences of DNA, amino acids, codons or proteins—and produce a phylogenetic tree, or phylogeny. The resulting phylogenetic trees are graphical representations of the evolutionary relationships between species, in terms of their common ancestors. The textbooks [15] and [25] serve as introductions to many of the concepts arising in phylogenetics.

In this thesis, we will investigate split and rank based methods and tools for phylogenetic inference, provide some new proofs for related results and introduce some new definitions. We aim to provide insight into these tools and methods from both practical and algebraic perspectives, as well as examine them through the use of simulations and analysis on real data sets.

In Chapter 1, we provide some necessary mathematical preliminaries. Chapter 2 is dedicated to a review of the current literature relevant to the topics explored in this thesis. Some definitions, theorems and examples which will be useful in the subsequent chapters are also included. Chapters 3 and 4 are theoretical in focus, covering *flattenings* and *subflattenings*, including details of their construction and their useful properties. Here we provide new proofs and definitions, along with some discussion and examples. Chapter 5 details simulations undertaken to evaluate subflattenings constructed in various different ways. We discuss the results of these simulations, and examine the performance of these subflattenings on some real DNA data sets. The final chapter is a broad discussion of the research outcomes, alongside some points of interest for future research.

Chapter 1

Preliminaries

Throughout this thesis we will assume some knowledge of basic linear algebra, for example the definition of a vector space and the definition of span, as well as basic group theoretic concepts, such as the definition of a group. Included in this chapter are a selection of items which will be of use later, particularly in Chapters 3 and 4. The examples in this chapter also help to motivate subsequent chapters.

1.1 Matrices and the Kronecker Product

Definition 1.1.1 (Matrix Rank)**.** *The* ***rank*** *of a matrix A is the number of linearly independent rows, or equivalently, the number of linearly independent columns. In particular, the rank of A is the dimension of the subspace spanned by its rows (or equivalently by its columns).*

Lemma 1.1.2 (Determinant properties)**.** *The matrix determinant* det *has the following multiplicative property for $n \times n$ matrices A, B:*

$$\det A \det B = \det(AB).$$

We also have

$$\det(A) = \det(A^T),$$

where A^T is the transpose of A, and,

$$\det(D) = d_1 d_2 ... d_n,$$

where D is a diagonal matrix, with diagonal entries $d_1, d_2, ..., d_n$.

Lemma 1.1.3. *Let M be an $n \times n$ matrix with $\det(M) \neq 0$. Then, M is full-rank. That is, $\operatorname{rank}(M) = n$.*

Definition 1.1.4. *Let A be an $m \times n$ matrix and let B be a $p \times q$ matrix, the **Kronecker product** of A and B is defined as the block matrix,*

$$A \otimes B = \begin{bmatrix} A_{1,1}B & \dots & A_{1,n}B \\ \vdots & \ddots & \vdots \\ A_{m,1}B & \dots & A_{m,n}B \end{bmatrix},$$

where $A_{i,j}$ is the (i,j)-th entry of A. The result is a $(mp \times nq)$ matrix, but is $(m \times n)$ when considered as a block matrix. More concisely, we have

$$(A \otimes B)_{I,J} = A_{i_1,j_1} B_{i_2,j_2},$$

where I and J are ordered pairs and i_t and j_t are the t-th entries of I and J respectively. Here we are indexing the rows of $(A \otimes B)$ by lexicographically ordered collection of pairs

$$(I = (a,b) : a \in \{1,...,m\}, b \in \{1,...,p\}),$$

and columns by the ordered collection of pairs

$$(J = (a,b) : a \in \{1,...,n\}, b \in \{1,...,q\}).$$

Associativity of the Kronecker product allows us to extend this indexing and notation to Kronecker products of any number of matrices. We simply include more entries in the tuples I and J.

We will frequently adopt the indexing notation defined in Definition 1.1.4.

For two block matrices, taking the Kronecker product as defined here involves ignoring the block partitioning of each matrix. The following result appearing in [28] and—from a different perspective—in [17], allows us to more easily take Kronecker products of block matrices.

Theorem 1.1.5. *Let A and B be $(m \times n)$ and $(p \times q)$ block matrices, and denote the (i,j)-th block of A or B by $A_{i,j}$ or $B_{i,j}$ respectively. Let M be a matrix with block entries given by*

$$M_{I,J} = A_{i_1,j_1} \otimes B_{i_2,j_2},$$

where I and J are ordered pairs and i_t and j_t are the t-th entries of I and J respectively. Then, we have that:

$$A \otimes B \text{ is permutation similar to } M.$$

That is, $A \otimes B$ can be obtained from M by simultaneous row and column permutations.

Proof. Proofs of this theorem are given in [28] and [17]. ■

Markov matrices are matrices which are often used to represent probabilities of changes between some number of states. We define Markov matrices formally, and discuss their use in phylogenetics in more detail in Chapter 2.

Definition 1.1.6 (Markov Matrix)**.** *A matrix M is a **Markov matrix**, **transition matrix**, or **substitution matrix** if its entries are all values in the range* $[0, 1]$*, and each of its columns sum to 1.*

Note that some authors instead choose to define Markov matrices as having their rows sum to 1, instead of columns. In either case, it is easy to see that the set of Markov matrices are closed under matrix multiplication. Formally, the set of Markov matrices form a semigroup. In fact, if we relax the restriction that matrix entries must lie between 0 and 1, and restrict to non-singular matrices, these matrices form a group [24].

1.2 Groups and Representations

We recall the definition of a group homomorphism, and the definition of the general linear group.

Definition 1.2.1 (Homomorphism)**.** *Let G, H be groups. A **homomorphism** is a map $\rho : G \to H$ which satisfies,*

$$\rho(g_1 g_2) = \rho(g_1)\rho(g_2)$$

for all $g_1, g_2 \in G$.

Definition 1.2.2 (General linear group)**.** *The **general linear group** $\mathrm{GL}(n, \mathbb{R})$ or $\mathrm{GL}(\mathbb{R}^n)$ is the group of $n \times n$ invertible matrices with real entries, under matrix multiplication. More generally, if V is a finite dimensional vector space, $GL(V)$ is the group of $\dim(V) \times \dim(V)$ invertible matrices, isomorphic to the group of bijective linear transformations $V \to V$.*

We also define the affine group, which has relevance in Chapter 4 and will be used within examples below.

Definition 1.2.3 (The affine group)**.** *The **affine group** $\mathrm{Aff}(n)$ is the collection of pairs (M, v), where M is an $n \times n$ matrix and v is an $n \times 1$ column vector. Multiplication in the group is given by*

$$(M_1, v_1)(M_2, v_2) = (M_1M_2, M_1v_2 + v_1),$$

for pairs $(M_1, v_1), (M_2, v_2) \in \mathrm{Aff}(n)$.

The affine group is related to Markov matrices [20]. If we take a matrix S which is invertible and has a constant row, then the below similarity transformation of a Markov matrix M gives the matrix

$$SMS^{-1} = \begin{bmatrix} T & u \\ 0 & 1 \end{bmatrix},$$

where T is an $(n-1) \times (n-1)$ matrix and u is a $(n-1) \times 1$ column vector. The matrix above is obtained by choosing S to have its constant row as the last row, however regardless of the choice, the resulting matrix will be permutation similar to the one shown above. One can verify the above transformation by considering the restrictions on the matrices. Suppose S has a constant final row, then the unit column sums in M mean we have the final row of SM equal to the final row in S. From here, we see that the final row of $(SM)S^{-1}$ is the final row of the identity matrix. The correspondence between the above matrix and the affine group is clear, since $(T, u) \in \mathrm{Aff}(n-1)$. If we again relax the restriction that every entry in our Markov matrices must lie between 0 and 1, we can obtain every element in $\mathrm{Aff}(n-1)$ in this way. Additionally, multiplication of the transformed matrices agrees with the multiplication in $\mathrm{Aff}(n-1)$, since,

$$\begin{bmatrix} T_1 & u_1 \\ 0 & 1 \end{bmatrix} \begin{bmatrix} T_2 & u_2 \\ 0 & 1 \end{bmatrix} = \begin{bmatrix} T_1T_2 & T_1u_2 + u_1 \\ 0 & 1 \end{bmatrix},$$

which, in the same way, corresponds to $(T_1T_2, T_1u_2 + u_1) \in \mathrm{Aff}(n-1)$, which is the product of (T_1, u_1) and (T_2, u_2) in $\mathrm{Aff}(n-1)$. Formally, this correspondence is what is known as a *matrix representation* of the group.

Definition 1.2.4 (Matrix representation)**.** *Given a group G, and a homomorphism*

$$\Phi : G \to \mathrm{GL}(\mathbb{R}^n),$$

*we say that $(\Phi, \mathbb{R}^n)$ is a **representation** of G on $\mathbb{R}^n$. Additionally, we say that the representation is **faithful** if the homomorphism Φ is injective.*

Definition 1.2.5 (Subrepresentation)**.** *Let $(\Phi, \mathbb{R}^n)$ be a representation of G on $\mathbb{R}^n$. A **subrepresentation** of $(\Phi, \mathbb{R}^n)$ is a pair $(\Phi|_V, V)$, such that V is a G-invariant* *subspace of $\mathbb{R}^n$. That is, we have*

$$\Phi(g)(v) \in V,$$

for all $g \in G$ and $v \in V$, where $\Phi(g)(v)$ indicates the group action of $\Phi(g)$ on v. Note that the group action here can be left or right matrix multiplication into column or row vectors respectively. The above property allows us to denote $\Phi|_V$ as the induced map,

$$\Phi|_V : G \to \mathrm{GL}(V),$$

taking group elements to the relevant submatrices of matrices in the image of Φ. One can show that a subrepresentation of a group G is itself a representation.

To provide an example of a subrepresentation of the affine group, we first need to define the direct product; a way to construct larger groups from smaller ones.

Definition 1.2.6 (Direct product)**.** *Let G and H be groups. The* ***direct product*** *of G and H, denoted $G \times H$, is the set,*

$$G \times H = \{(g, h) : g \in G, h \in H\},$$

with multiplication given by

$$(g_1, h_1)(g_2, h_2) = (g_1 g_2, h_1 h_2).$$

One can easily see that $G \times H$ is itself a group. We also define the n-fold direct product of a group G with itself to be the set,

$$\times^n G = \{(g_1, ..., g_n) : g_i \in G\}.$$

along with the analogous definition of multiplication.

Example 1.2.7. *We previously discussed a correspondence between the affine group and the transformed Markov matrices, and mentioned that that correspondence is formally referred to as a representation. Formally, this representation of $\mathrm{Aff}(n)$ is given by the map $\rho : \mathrm{Aff}(n) \to \mathrm{GL}(n+1, \mathbb{R})$, defined via,*

$$\rho((M, u)) := \begin{bmatrix} M & u \\ 0 & 1 \end{bmatrix} \in \mathrm{GL}(n+1, \mathbb{R}).$$

Now, consider the direct product of the affine group with itself,

$$G := \mathrm{Aff}(n) \times \mathrm{Aff}(n).$$

We have a related representation of this group; the map

$$\Phi : \mathrm{Aff}(n) \times \mathrm{Aff}(n) \to \mathrm{GL}((n+1)^2, \mathbb{R}),$$

defined using the Kronecker product,

$$\begin{aligned}\Phi\left(((M_1, u_1), (M_2, u_2))\right) &:= \begin{bmatrix} M_1 & u_1 \\ 0 & 1 \end{bmatrix} \otimes \begin{bmatrix} M_2 & u_2 \\ 0 & 1 \end{bmatrix} \\ &= \begin{bmatrix} M_1 \otimes M_2 & M_1 \otimes u_2 & u_1 \otimes M_2 & u_1 \otimes u_2 \\ 0 & M_1 & 0 & u_1 \\ 0 & 0 & M_2 & u_2 \\ 0 & 0 & 0 & 1 \end{bmatrix}.\end{aligned}$$

One can verify that the map Φ is a homomorphism.

Consider the following subspace of $\mathbb{R}^{(n+1)^2}$ defined by,

$$V := \{(0, ..., 0, \lambda_1, ..., \lambda_{2n+1}) : \lambda_i \in \mathbb{R}\} \cong \mathbb{R}^{2n+1},$$

that is, each vector in V is a row-vector with zeroes in the first n^2 locations. We see that for $g \in G$ and $v = (0, ..., 0, \lambda_1, ..., \lambda_{2n+1}) \in V$, we have

$$\begin{aligned}\Phi(g)(v) &= v\Phi(g) \\ &= (0, ..., 0, \lambda_1, ..., \lambda_{2n+1}) \begin{bmatrix} M_1 \otimes M_2 & M_1 \otimes u_2 & u_1 \otimes M_2 & u_1 \otimes u_2 \\ 0 & M_1 & 0 & u_1 \\ 0 & 0 & M_2 & u_2 \\ 0 & 0 & 0 & 1 \end{bmatrix} \\ &= (0, ..., 0, \gamma_1, ..., \gamma_{2n+1}) \in V,\end{aligned}$$

for some $\gamma_i \in \mathbb{R}$. That is, V is G-invariant, and thus we have a subrepresentation $(\Phi|_V, V)$, with

$$\Phi|_V(g) = \begin{bmatrix} M_1 & 0 & u_1 \\ 0 & M_2 & u_2 \\ 0 & 0 & 1 \end{bmatrix} \in \mathrm{GL}(2n+1, \mathbb{R}).$$

One can easily verify that this class of matrices provides a representation of $\mathrm{Aff}(n) \times \mathrm{Aff}(n)$. It should also be apparent that this representation is faithful.

A non-faithful representation which maps group elements to the matrix with a single entry ‘1’ is obtainable in a similar way, by selecting V to be a particular subspace of $\mathbb{R}^{(n+1)^2}$ isomorphic to $\mathbb{R}$:

Example 1.2.8. *Again, consider the homomorphism $\rho : \mathrm{Aff}(n) \to \mathrm{GL}(n+1, \mathbb{R})$ defined via*

$$\rho((M, u)) := \begin{bmatrix} M & u \\ 0 & 1 \end{bmatrix} \in \mathrm{GL}(n+1, \mathbb{R}),$$

giving a representation of $\mathrm{Aff}(n)$*. Consider the subspace* $U = \mathrm{span}\{(0, ..., 0, 1)\} \cong \mathbb{R}$ *of* $\mathbb{R}^{n+1}$*. We see that*

$$\Phi(g)(v) = v\Phi(g) = (0, ..., 0, \lambda) \begin{bmatrix} M & u \\ 0 & 1 \end{bmatrix} = (0, ..., 0, \lambda) \in U.$$

That is, $U \cong \mathbb{R}$ is an $\mathrm{Aff}(n)$*-invariant subspace and we have a subrepresentation* $(\Phi|_U, \mathbb{R})$*, with*

$$\Phi|_U(g) = (1) \in \mathrm{GL}(1, \mathbb{R}).$$

These representations of direct products of copies of the affine group will appear again in later chapters.

1.3 Singular Value Decomposition

Definition 1.3.1. *A* ***singular value decomposition*** *of a real-valued $m \times n$ matrix M is the factorisation*

$$M = U\Sigma V^T,$$

where U is an orthogonal $m \times m$ matrix, V is an $n \times n$ orthogonal matrix, and Σ is a rectangular diagonal matrix with non-negative entries. We have,

$$\mathrm{diag}(\Sigma) = (\sigma_1, ..., \sigma_{min\{m,n\}}),$$

and refer to $\sigma_1 \geq ... \geq \sigma_{min\{m,n\}} \geq 1$ as the ***singular values*** *of M.*

Unless otherwise stated, here σ_i will always refer to the i-th largest singular value when the corresponding matrix is apparent from context. We will now define a matrix norm which is related to singular values.

Definition 1.3.2. *The* ***Frobenius norm,*** $\|A\|_F$ *of an $m \times n$ matrix A is defined by,*

$$\|A\|_F = \sqrt{\sum_{j=1}^{n} \sum_{i=1}^{m} A_{i,j}^2} = \sqrt{\sum_{i=1}^{\min\{m,n\}} \sigma_i^2}.$$

The Eckart–Young–Mirsky theorem, provided in [9], relates singular values to the Frobenius distance between any matrix and its closest rank-k matrix.

Theorem 1.3.3 (Eckart–Young–Mirsky Theorem)**.** *Let M be a matrix with* $\operatorname{rank}(M) \geq k$*. The distance to the closest rank-k matrix to M under the Frobenius norm is given by*

$$\min_{A\,:\,\operatorname{rank}(A)=k} \|M - A\|_F = \sqrt{\sum_{i=k+1}^{\min\{m,n\}} \sigma_i^2},$$

where σ_i are the ***singular values*** *of M.*

Proof. The proof is provided in [9]. ■

Chapter 2

Background & Literature Review

2.1 Sequence Data, DNA

Tools used for phylogenetic inference often take as their input, biological data in the form of sequences; for example, sequences of DNA, codons or amino acids. We will frequently give examples in terms of DNA sequence data. DNA sequences can be represented as strings of states from the set of nucleotides $\mathcal{S} = \{A, G, C, T\}$. We refer to $\mathcal{S}$ as a *state space*, and throughout we will denote $k = |\mathcal{S}|$. The states A, G are referred to as *purines* and C, T as *pyrimidines.* Some methods and models in phylogenetics make this distinction (see K2ST in Appendix A), whereas others treat each state equally (see JC69 in Appendix A), or are independent of the number of states (for example the general Markov model, which we discuss in Section 2.3). As a species evolves, parts of its DNA will change. We refer to these changes as *substitutions.* The substitutions $A \leftrightarrow G$ and $C \leftrightarrow T$—those which take *purines* to *purines* or *pyrimidines* to *pyrimidines* respectively—are referred to as *transitions*, and all other state changes are known as *transversions.*

Species which have evolved from a recent common ancestor will share some DNA. Given some number of species (taxa) and corresponding DNA sequences, one can align the sequences into a *DNA sequence alignment.* Note that the process of constructing a sequence alignment from DNA sequences is itself a difficult problem, which we do not discuss here (see [15]). An example of a small portion of such an alignment is given in Table 2.1. We can look at the columns in an alignment (which we refer to as site-patterns) to see how the DNA sequences differ between taxa, for example the highlighted site-pattern 'GTT' in Table 2.1 shows that in this particular place in the alignment, the Gorilla and Chimp DNA sequences share the pyrimidine T. We say that this provides some small amount of support for Chimps and Gorillas sharing a most recent common ancestor, however there are many more site-patterns to consider. A sequence alignment can be summarised by counting the frequency of each of the k^n possible site-patterns (where n is the number of taxa). Dividing each pattern count by the sequence length gives the proportion of each pattern within the alignment (see Table 2.1). One can imagine that these proportions are estimates for the probability of observing each pattern in an alignment with infinite length.

Taxon	Characters							
Human	...	A	A	G	**G**	T	-	...
Gorilla	...	A	A	G	**T**	T	C	...
Chimp	...	A	-	G	**T**	T	A	...

Pattern	Frequency
AAA	12
GGG	19
GTT	9
⋮	⋮

Table 2.1: The sequence alignment **(left)** describes three taxa, each with a sequence of DNA showing six states from the set $\mathcal{S} = \{A, G, C, T, -\}$. Columns, such as the one in bold, are site-patterns. The 'indels', indicated by '-', refer to insertions or deletions. The table of site-pattern frequencies **(right)** is constructed by counting occurrences of each site-pattern in the left table. Site-patterns involving indels are often ignored. This table of frequencies can of course be converted into a table of empirical site-pattern probabilities by dividing by the sequence length—the number of counted site-patterns in the alignment.

2.2 Trees, Splits and Maximum Parsimony

Trees are used in phylogenetics to represent the evolutionary history of a set of taxa. In a phylogenetic tree (Definition 2.2.1), the leaves represent the taxa, and the remaining vertices represent the ancestral history of the taxa.

Definition 2.2.1 (Phylogenetic tree)**.** *A **Phylogenetic X-tree**, $T(X)$, is a tree (V, E) with set of leaves X corresponding to the collection of taxa under consideration. We will often label taxa at the leaves by integers. V is the set of vertices of T, and E the set of edges, sometimes denoted $V(T)$ and $E(T)$ respectively. A tree T is 'rooted' if a single vertex in $V(T)$ is specified as the 'root vertex'.*

The goal of phylogenetic inference is to infer the 'true' tree by looking at the data concerning the taxa at the leaves. By 'true tree', we mean either the tree showing the actual evolutionary history of the species being considered, or—in the case of simulated data—the tree from which the data was generated. We can only know for sure that we have identified the true tree in the latter case. We aim to find the true tree by looking at each site-pattern from the alignment independently, and finding the tree which makes the most sense across all of these site-patterns (see Figure 2.1).

Definition 2.2.2 (character)**.** *A **character** is a map*

$$f : X \to S,$$

where X is the set of leaves of some phylogenetic tree T, and S is set of states.

For example, given a sequence alignment we may consider the set of states $\mathcal{S}$ corresponding to DNA, and think of a character f as corresponding to a site-pattern. In Table 2.1, the highlighted column is the site-pattern given by character f defined by,

$$\begin{aligned} \text{Human} &\overset{f}{\mapsto} G, \\ \text{Gorilla} &\overset{f}{\mapsto} T, \\ \text{and, Chimp} &\overset{f}{\mapsto} T. \end{aligned}$$

The first methods used for phylogenetic inference required only a set of characters, derived from site-patterns from a sequence alignment. One such method is *maximum parsimony.* The intuitive idea behind maximum parsimony was introduced by Edwards and Cavalli-Sforza [10] with the quote:

"The most plausible estimate of the evolutionary tree is that which invokes the minimum net amount of evolution".

The process involves finding the tree which requires the fewest number of changes required along its internal edges for each of the given characters. Methods for achieving this first appeared in [6], and more algorithms for finding the minimum number of changes required for a given character have since been developed (see [25]).

Definition 2.2.3 (Split)**.** *A **split** is a partition of a leaf-set X into two sets A and B, denoted $A|B$, where $A, B \subseteq X$ are non-empty. If for a split $A|B$, we have a tree $T(X)$ such that when a particular edge is be removed, the leaves of the two resultant subtrees correspond to the subsets A and B, we say that the split $A|B$ is **displayed** by $T(X)$. In the context of inferring a tree on X, a split $A|B$ is called a **true** split if it is displayed by the true tree. A split is considered to be **balanced** if $|A| = |B|$ and **unbalanced** otherwise. Finally, if $|A| = 1$ or $|B| = 1$, we say the split $A|B$ is **trivial**.*

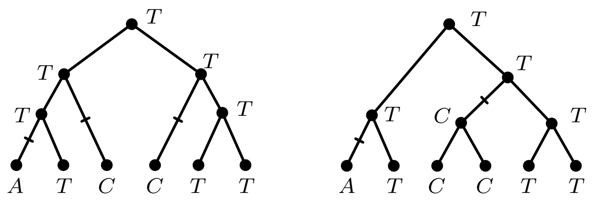


Figure 2.1: Two possible trees on six-taxa for the site-pattern '$ATCCTT$'. The tree on the left requires a change along three different branches to fit the given site-pattern, whereas the tree on the right requires only two. Maximum parsimony (Definition 2.2.5) suggests that for the current site-pattern, the right tree is more likely.

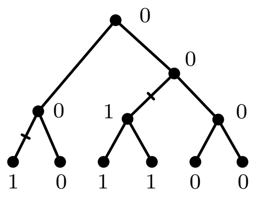
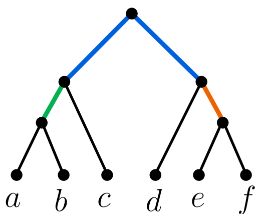


Figure 2.2: An example of the split 134|256 represented as a binary character on a six-taxon tree. The parsimony score for this split is 2, since a change is required along two separate edges



$ab|cdef$, $abc|def$, $abcd|ef$.

Figure 2.3: 'True' splits correspond to edges in a tree which, when removed, partition the leaves of the tree. The tree is uniquely defined by this collection of splits (see Theorem 2.2.4). The leaves in the leaf-set $X = \{a, b, c, d, e, f\}$ represent present-day species. The colour and position of the above splits show their correspondence with the internal edges of the tree. The edges which are directly connected to the root vertex are treated as a single edge, since both edges induce the same bipartition of taxa when removed.

For simplicity of notation, we write $A|B$ as $abc|def$ where $A = \{a, b, c\}$ and $B = \{d, e, f\}$. Also note that since a split is a bipartition (a partition of the leaf-set into two blocks), there is no distinction between splits $A|B$ and $B|A$, and that we often consider only non-trivial splits, since trivial splits are (trivially) true. See Figure 2.3 for an example of a set of all non-trivial splits displayed by a particular 6-taxon tree.

Given a leaf-set X we see that the number of possible splits is given by $2^{|X|-1}-1$. The number of splits displayed by a given X-tree T is simply the number of edges $|E(T)|$. In either case, to count only the non-trivial splits, we can subtract $|X|$. The importance of splits in phylogenetic inference is made apparent in Theorem 2.2.4

Theorem 2.2.4. *A tree with leaf-set X can be reconstructed uniquely from the set of all splits that are displayed by the tree.*

Proof. See Chapter 2 in [25]. See Figure 2.3 for an example. ■

We now formally define a *parsimony score* for both site-patterns and splits.

Definition 2.2.5 (Parsimony score for a character)**.** *The* ***parsimony score*** *for a site-pattern or character f on a given tree T and state space $\mathcal{S}$, is the minimum number of substitutions required along edges of T to obtain the site-pattern f at the leaves, considering all possible labellings of internal vertices by states in $\mathcal{S}$.*

Definition 2.2.6 (Parsimony score for a split)**.** *The* ***parsimony score for a split*** *$A|B$ is the parsimony score for the character that maps taxa in A and B to the states 0 and 1 respectively, with binary state-space $\mathcal{S}$.*

See Figure 2.1 for an example of parsimony scores for site-patterns, and Figure 2.2 for an example of the parsimony score for a split. Note that since true splits correspond to an internal edge, they have a parsimony score of 1.

True splits each tell us something about the true phylogeny—that two subsets of species are somehow separate. In other words, they are separated by an edge in the true evolutionary tree (see Figure 2.3). Fundamentally, we can see that the core problem of phylogenetic inference is identifying these true splits.

2.3 The General Markov Model

In 1978, Felsenstein [13] pointed out that some methods for inferring phylogenetic trees, including maximum parsimony (as described in Section 2.2), compatibility, and distance methods, are not always statistically consistent. That is, it is not

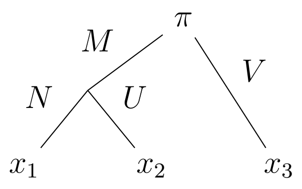


Figure 2.4: A 3-taxon tree with leaf set $L(T) = \{x_1, x_2, x_3\}$. M, N, U and V are substitution matrices and π is the initial distribution of states at the root.

the case that given more data (longer sequences) the tree chosen by these methods will necessarily converge to the true tree. Maximum likelihood methods assume that a sequence of characters have been generated under some stochastic model, and select the tree and model parameters for which the characters have the highest probability of occurring. Under certain conditions, these methods can be shown to be consistent [29, 12]. An example of such a stochastic model is the general Markov model of character evolution. The general Markov model is represented as a class of $k \times k$ substitution matrices with rows and columns indexed by the set of states $\mathcal{S}$, which describe the probability of a substitution from one state to another along a single edge, as defined in Definition 1.1.6. (see Appendix A for examples). Given substitution matrices for the edges of a tree, and an initial distribution of states at the root which describes the probability of beginning in a particular state (often denoted π), we can calculate the probability $P(f)$ of observing a given character f on the leaves of the tree.

For example, if we take the 3-taxon tree shown in Figure 2.4 with some initial distribution π and substitution matrices M, V, N, U on the edges, the probability of the site pattern given by a character f under our model is given by,

$$P(f) = \sum_{i,j \in \mathcal{S}} \pi_i \; M_{j,i} \; N_{f(x_1),j} \; U_{f(x_2),j} \; V_{f(x_3),i},$$

where the pq-th entry of each substitution matrix is the probability of a substitution to p from q along the corresponding edge. The index i can be thought of as ranging over the possible states of the root vertex. Similarly, we think of j as ranging over the possible states of the parent of x_1 and x_2.

The parameters for the general Markov model are the entries of these substitution matrices. The only restriction is that the entries in each column are positive and sum to 1. Note that the convention of having unit row-sums is often used, in which the ij-th entry of each transition matrix is the probability of a substitution from i to j. Throughout, we adopt the unit column-sum convention. Sub-models of the general Markov model are often used, because they have a reduced number of parameters. This is done by adding more constraints to substitution matrices (see Appendix A). Importantly, this means that results concerning the general Markov model are applicable to sub-models as well.

2.4 Pruning Algorithm

We can efficiently calculate site pattern probabilities using the following recursive algorithm first introduced in [12] and described further in [14, 15].

Given the state-set $\mathcal{S} = \{A, C, G, T\}$ and initial distribution vector π, define $L_v(s)$ to be the probability of seeing the observed states following from vertex $v \in V(T)$, given that vertex v has state s. To initialise the algorithm, define for each leaf $x \in L(T)$,

$$L_x(s) = \begin{cases} 0 & \text{if } f(x) = s, \text{ and} \\ 1 & \text{if } f(x) \neq s, \end{cases}$$

where f is the character on the leaves of the tree.

The next step is to calculate $L_v(s)$ for each state s, and for every vertex v in the tree in a post-order traversal. That is, begin with any vertex which has only leaves as children and continue, only calculating $L_x(s)$ for vertices having children c for which $L_c(s)$ has already been calculated.

Suppose that vertex v has the set of child vertices (direct descendants) denoted C_v. Suppose also that a vertex u has an in-branch substitution matrix denoted M_u. Then, $L_v(s)$ is calculated as follows:

$$L_v(s) = \prod_{c \in C_v} \left(\sum_{s' \in S} (M_c)_{s',s} \, L_c(s') \right).$$

The last vertex visited will be the root vertex r, and the probability of observing the site pattern f denoted $P(f)$, is evaluated in the sum:

$$P(f) = \sum_{x \in S} \pi_x \, L_r(x).$$

Calculating $P(f)$ for each possible f allows one to construct a probability distribution P describing the probability of observing each site-pattern on the leaves of the tree. We can think of this probability distribution as an empirical probability distribution obtained from a sequence alignment, but with an infinite sequence length. Choosing the most likely tree—that is, the phylogeny that maximises $P(f)$ for each site-pattern f—requires navigating through tree-space, the set of all trees on X. This is computationally difficult due to the large number of trees. However, from a theoretical point of view, the notation developed above allows us to state the following lemma.

Lemma 2.4.1. *Consider a binary tree $T(X)$, and take two leaves α, β with the same parent vertex. Let the Markov matrices on the edges of these leaves each be the identity. Then, the probability $P(f)$ of observing any site-pattern given by a character f with $f(\alpha) \neq f(\beta)$, is zero. Further, the probability $P(g)$ of observing a site-pattern given by character g with $g(\alpha) = g(\beta)$, is equal to the probability of observing the same site-pattern on the smaller tree $T(X - \{\beta\})$ where the subtree with leaves α, β in $T(X)$ is replaced with a single leaf, labelled α.*

Proof. Let f be such a character. we have the site-pattern probability (calculated in the naïve way described in Section 2.3) given by,

$$P(f) = \sum_{i_1, i_2, \ldots i_{n-1}, i_n \in \mathcal{S}} K(i_1, i_2, ..., i_{n-1}, f) I_{f(\alpha), i_n} I_{f(\beta), i_n},$$

where K is a product of entries of substitution matrices on internal edges, $I_{a,b}$ is the a, b-th entry of the identity matrix, and i_m is ranging over the possible states at the parent vertex of α and β. Since $I_{a,b} = 1$ if $a = b$ and 0 otherwise, we see that the only non-zero terms in the summation occur when $i_m = f(\alpha) = f(\beta)$, so the probability evaluates to zero. Further, we see that if we have a site-pattern given by g with $g(\alpha) = g(\beta)$, we have

$$\begin{aligned}
P(g) &= \sum_{i_1, i_2, \ldots i_{n-1}, i_n \in \mathcal{S}} K(i_1, i_2, ..., i_{n-1}, g) I_{g(\alpha), i_n} I_{g(\beta), i_n} \\
&= \sum_{i_1, i_2, \ldots i_{n-1}, i_n \in \mathcal{S}} K(i_1, i_2, ..., i_{n-1}, g) I_{g(\alpha), i_n} \\
&= \sum_{i_1, i_2, \ldots i_{n-1}, i_n \in \mathcal{S}} K^*(i_1, i_2, ..., i_{n-1}, g) M_{i_n, i_{n-1}} I_{g(\alpha), i_n} \\
&= \sum_{i_1, i_2, \ldots i_{n-1} \in \mathcal{S}} K^*(i_1, i_2, ..., i_{n-1}, g) M_{g(\alpha), i_{n-1}},
\end{aligned}$$

where M is the Markov matrix on the edge leaving the parent of α and β, and the product $K*$ is K with $M_{i_n, i_{n-1}}$ taken out. This is simply an expression for the probability of the same site-pattern g on a smaller tree where α, β are treated as a single leaf which we call α, as required. ■

Given a tree T with leaves in X denoted by integers, and a character f defined by $f(j) = i_j \in \mathcal{S}$, we denote $P(f) = p_{i_1,\ldots,i_{|X|}}$. That is, the order of the states i_j in the subscript of p describe the character f. We also denote the collection of all possible site-pattern probabilities $P(f)$ by P.

2.5 Flattenings

Over the last two decades, significant progress has being made in developing an understanding of algebraic structures underlying phylogenetic models. *Phylogenetic invariants*, introduced in [7, 22], are polynomials which vanish when evaluated at site-pattern probabilities for a particular tree under a given model. For an introduction to phylogenetic invariants, see [2]. These polynomials have been shown to be useful in phylogenetic inference [1]. Methods for phylogenetic inference described so far (Section 2.2 and Section 2.4) require searching through tree-space in order to find a phylogeny which best suits the given alignment. Ideas involving the theory of phylogenetic invariants such as *flattenings* can be used for inference without the need to search through trees [11, 3]. We discuss flattenings without making explicit references to phylogenetic invariants.

Definition 2.5.1 (Flattening)**.** *Consider a phylogenetic tree T with leaf-set X, a state space $\mathcal{S}$, a split $A|B$, and a vector of all possible site-pattern probabilities $P = (p_{i_1,i_2,\ldots,i_{|X|}})_{i_j \in \mathcal{S}}$. A **flattening**, denoted $\mathrm{Flat}_{A|B}(P)$, is an arrangement of these probabilities into a matrix with rows indexed by all $|A|$ length words with alphabet $\mathcal{S}$, and columns indexed by words with length $|B|$, such that probability $p_{i_1,i_2,\ldots,i_{|X|}}$ is located at position $(i_{a_1} \ldots i_{a_{|A|}}, i_{b_1} \ldots i_{b_{|B|}})$, where $a_l \in A$ and $b_l \in B$.*

Flattenings are perhaps best explained via an example:

Example 2.5.2. *Consider a four-taxon tree with a binary state space ($k = 2$) and set of taxa $X = \{0, 1, 2, 3\}$. The flattening produced from the split $03|12$ would be:*

$$\mathrm{Flat}_{03|12}(P) = \begin{array}{c} \\ 00 \\ 01 \\ 10 \\ 11 \end{array} \!\!\begin{array}{c} \begin{array}{cccc} 00 & 01 & 10 & 11 \end{array} \\ \begin{bmatrix} p_{0000} & p_{0010} & p_{0100} & p_{0110} \\ p_{0001} & p_{0011} & p_{0101} & p_{0111} \\ p_{1000} & p_{1010} & p_{1100} & p_{1110} \\ p_{1001} & p_{1011} & p_{1101} & p_{1111} \end{bmatrix} \end{array},$$

with rows and columns indexed by all length-2 words with alphabet $\mathcal{S} = \{0, 1\}$.

The property of flattenings which is useful to phylogenetic inference is related to the rank (Definition 1.1.1). To more easily describe this property though, we need the following definition of *generic rank*.

Definition 2.5.3. *Consider a phylogenetic tree T under a general Markov model. If the initial distribution π at the root has strictly positive entries, and the Markov matrices on each edge of T have full-rank, we say that the model is* ***generic****. For a matrix with entries depending on some generic model, we refer to the rank of the matrix as the* ***generic rank****.*

Throughout, we will enforce the restrictions defined in Definition 2.5.3 on the general Markov model, and thus any results concerning the rank of matrices are results concerning the generic rank.

The applicability of flattening matrices to phylogenetic inference is due to the following theorem provided in [11] and [3].

Theorem 2.5.4. *Given a phylogenetic tree T and state space $\mathcal{S}$, the flattening matrix $\mathrm{Flat}_{A|B}(P)$ computed from a split $A|B$ has $\mathrm{rank}(\mathrm{Flat}_{A|B}(P)) = k$ if the split $A|B$ is displayed by T, and has $\mathrm{rank}(\mathrm{Flat}_{A|B}(P)) \geq k^2$ otherwise.*

Proof. The result is proven in [11]. We provide a new proof in Section 3.1 which is more closely analogous to the proof of the rank properties of subflattenings given in [26]. We will describe subflattenings shortly. ■

Lemma 2.5.5. *Consider a site-pattern probability distribution P as having evolved under a general Markov model on a tree $T(X)$. Let $\tilde{P}$ be the probability distribution corresponding to $T(X)$ with the Markov matrices at the leaf edges replaced by the identity matrix. Then, for a split $A|B$ we have*

$$\mathrm{rank}(\mathrm{Flat_{A|B}(P)}) = \mathrm{rank}(\mathrm{Flat_{A|B}(\tilde{P})})$$

in the generic case.

Proof. We begin with the following relationship relating P and $\tilde{P}$ given in [27]:

$$P = (M_1 \otimes M_2 \otimes ... \otimes M_{|X|}) \cdot \tilde{P},$$

where M_i are Markov matrices on the edges of the leaves in $T(X)$. Now, considering $\mathrm{Flat}_{A|B}(P)$ as a joint-distribution, it is possible to show that

$$\mathrm{Flat}_{A|B}(P) = M_A \mathrm{Flat}_{A|B}(\tilde{P}) M_B^T,$$

where M_A and M_B are the Kronecker products (see Section 1.1) of all the Markov matrices on the edges of the leaves in A and B respectively. These matrices are full-rank, and rank is invariant under multiplication by full-rank square matrices, giving us the result. ■

The sizes of flattening matrices increase exponentially in the number of taxa. Even a 4-taxon flattening with $k = 4$ is cumbersome to write down. *Subflattenings* are smaller matrices which exhibit rank properties similar to those described in Theorem 2.5.4. We now discuss the construction of these subflattening matrices.

2.6 Subflattenings

As mentioned in Section 2.5, flattening matrices have desirable properties but are very large—exponential in the number of taxa. Introduced in [26], *subflattenings* are derived from flattenings, and offer analogous properties with the benefit of reduced dimensions—the number of entries in a subflattening matrix is quadratic in the number of taxa. To define subflattenings, we first need to introduce the following transformation of the flattening.

Definition 2.6.1 (Transformed flattening)**.** *Given a flattening* $\mathrm{Flat}_{A|B}(P)$*, we define the* ***transformed flattening****,*

$$\mathrm{Flat}^{S}_{A|B}(P) := \left(\bigotimes_{i=1}^{|A|} S\right) \mathrm{Flat}_{A|B}(P) \left(\bigotimes_{i=1}^{|B|} S\right)^{T}, \tag{2.1}$$

where S is some $k \times k$ matrix which is invertible and has a constant row. The transformed flattening has the same shape as the original flattening, and it is therefore still indexed by strings of characters in $\mathcal{S}$.

Note that the matrix S in Definition 2.6.1 is chosen because it takes Markov matrices to elements of the affine group via similarity transformation, as discussed in Section 1.2.

Definition 2.6.2 (Subflattening)**.** *Let* $\mathrm{Flat}^{S}_{A|B}(P)$*, be a transformed flattening with rows and columns indexed by strings of characters from $\mathcal{S}$ with length $|A|$ and $|B|$ respectively. Take the state $s \in \mathcal{S}$ which corresponds to the index of the constant row in S (note that by doing this, we are choosing an ordering for the state-space $\mathcal{S}$). Define the* ***subflattening****,* $\mathrm{Subfl}^{S}_{A|B}(P)$*, as the* $(|A|(k-1)+1)\times(|B|(k-1)+1)$ *sub-matrix consisting of entries* $\left(\mathrm{Flat}^{S}_{A|B}(P)\right)_{\alpha\beta}$ *where strings α and β each include at most one state in $\mathcal{S}$ that isn't s.*

For a split $A|B$, the dimensions of the corresponding sub-flattening are

$$(|A|(k-1)+1) \ \times \ (|B|(k-1)+1).$$

That is, if the dimensions of the flattening matrix are $m \times n$ where $m = k^{|A|}$ and $n = k^{|B|}$, then the dimensions of the corresponding sub-flattening are

$$(log_k(m^{k-1})+1) \ \times \ (log_k(n^{k-1})+1).$$

Subflattening matrices have a rank property that is analogous to Theorem 2.5.4 for flattenings [26].

Theorem 2.6.3. *Let T be a tree with site pattern probability distribution P. Let $A|B$ be a split. A subflattening matrix $\mathrm{Subfl}^S_{A|B}(P)$ has rank k if $A|B$ is displayed by T, and rank greater than or equal to $2(k-1)+1$ otherwise.*

Proof. The proof is given in the appendix in [26] ■

The proof of the rank condition for the subflattenings in [26] is in some way similar to the proof of Theorem 2.5.4 for flattenings that we give in Section 3.1. In the subflattenings case, matrix entries are linear combinations of the site-pattern probabilities and so extra care is needed to claim linear independence of rows and columns in the subflattening following a divergence event (a leaf splitting into two new leaves, corresponding to a species diversifying into two new ones). The proof given in [26] appeals to theory of phylogenetic invariants.

We see that flattenings and subflattenings—along with the corresponding results Theorem 2.5.4 and Theorem 2.6.3—can be used to identify true splits:

Algorithm 2.6.4. *Given a DNA sequence alignment ($\mathcal{S} = \{A, C, G, T\}$) on set of taxa X, one can attempt to identify the most likely phylogenetic tree to fit the alignment as follows:*

1. *Count the frequency of each site pattern in the sequence alignment to derive an estimate of the probability distribution $\widehat{P}$.*
2. *Choose a suitable matrix S and for each split $A|B$ of X, construct $\mathrm{Subfl}^S_{A|B}(\widehat{P})$.*
3. *Estimate the rank of $\mathrm{Subfl}^S_{A|B}(\widehat{P})$ (Section 2.7) to decide whether or not $A|B$ is a true split.*
4. *Once all the splits have been evaluated, conclude that the true tree is the one defined by the set of true splits.*

There are a number of issues arising from the process suggested in Algorithm 2.6.4. First, we require a method for estimating the rank of the subflattening matrices.

We also require a course of action if two ‘true’ splits are *incompatible*, that is, they are not displayed by any single true tree; for example, splits 12|34 and 13|24. Finally, another concern is that Algorithm 2.6.4 involves constructing and evaluating a flattening or subflattening matrix for each of the $2^{|X|-1}-1$ possible splits on a given X-tree. Eriksson [11] provides a statistically consistent algorithm, which involves the evaluation of at most $(|X|-1)^2-3$ splits, and suggests singular value decomposition for examining the rank of flattenings. We will discuss singular value decomposition in Section 2.7 and employ a version of Eriksson’s SVD algorithm in Chapter 5.

2.7 Rank Estimation

Importantly, the rank properties described in Theorem 2.5.4 and Theorem 2.6.3 are theoretical results and assume an infinite sequence length. In practice finite sequence lengths mean that flattening matrices are often sparse with rank simply equal to the number of non-zero rows or columns, and subflattening matrices will often be full-rank even for true splits. Identifying true splits through the application of these results therefore requires a measure of how *close* these matrices are to being rank k. The use of singular value decomposition (Definition 1.3.1), along with the Eckart-Young-Mirsky theorem (Theorem 1.3.3), has been suggested by Eriksson [11]. Further work has been done to develop and optimise algorithms involving flattenings and singular values [4, 21, 8, 16]. Optimisations specific to the use of subflattenings have not yet been investigated.

To aid in applying Theorem 1.3.3 to flattenings and subflattenings, [4] provides the following normalised measure of ‘closeness’ to rank k.

Definition 2.7.1 (Split score)**.** *The **split score** for an $m \times n$ flattening or subflattening $F_{A|B}$ corresponding to a split $A|B$ is defined to be*

$$\mathrm{S}(F_{A|B}) = \sqrt{\frac{\sum_{i=k+1}^{\min\{m,n\}} \sigma_i^2}{\sum_{i=1}^{\min\{m,n\}} \sigma_i^2}} = \sqrt{1 - \frac{\sum_{i=1}^{k} \sigma_i^2}{\|F_{A|B}\|_F^2}},$$

*where $\sigma_1 \geq ... \geq \sigma_{min\{m,n\}} \geq 1$ are the **singular values** of $F_{A|B}$.*

The benefits of the above definition, as described in [4], are that only the k largest singular values are needed, and that the split score has values between 0 and 1. A score of 0 indicates the matrix in question is rank k.

2.8 Hadamard Matrices

Hadamard matrices are prominent in phylogenetics, with applications in phylogenetic inference and the study of models and methods [18, 19, 5]. We define Hadamard matrices here.

Definition 2.8.1 (Hadamard matrices)**.** *A $n \times n$ matrix H is a* ***Hadamard matrix*** *if each of its entries are either 1 or -1 and its rows are mutually orthogonal. They satisfy $HH^T = nI$.*

The construction of Hadamard matrices of size $(2^n \times 2^n)$ for a given n, is simple:

Example 2.8.2. *The following is an example of a 2×2 Hadamard matrix,*

$$H_2 = \begin{bmatrix} 1 & -1 \\ 1 & 1 \end{bmatrix},$$

and we can use this matrix and the Kronecker product to obtain a 4×4 Hadamard matrix.

$$H_4 = H_2 \otimes H_2 = \begin{bmatrix} 1 & -1 & -1 & 1 \\ 1 & 1 & -1 & -1 \\ 1 & -1 & 1 & -1 \\ 1 & 1 & 1 & 1 \end{bmatrix}.$$

In general, the Kronecker product of n copies of H_2 will result in a $2^n \times 2^n$ Hadamard matrix.

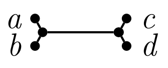
The 4×4 Hadamard matrix is an example of a valid choice for the matrix S in the construction of the subflattening (Definition 2.6.1). We will use Hadamard matrices for this purpose in Chapter 5.

Chapter 3

Flattenings

In this chapter, we provide a new proof for the flattening rank theorem (Theorem 2.5.4). We begin with a small example which illustrates the result and draws attention to some key ideas employed within the proof.

Consider the following four taxon tree under the general Markov model, with binary state-space $\mathcal{S} = \{0, 1\}$. We draw the leaf vertices close to the internal vertices to indicate that the leaf edges have zero-length. The meaning of this is that the Markov matrix on each of these edges is the identity.



From a continuous-time perspective, this corresponds to the idea that there are no substitutions per unit of time along these edges, or equivalently, that no time has passed. For details on interpreting and modelling phylogenetic trees as continuous-time processes, see [25].

Let $P_0 = (p_{ijml})_{i,j,m,l \in \mathcal{S}}$ be the site-pattern probability distribution arising from the model parameters. From Lemma 2.4.1, we observe that,

$$p_{ijml} = 0 \text{ whenever } i \neq j \text{ or } m \neq l,$$

allowing us to simplify the flattenings corresponding to the three possible splits:

(with omitted entries all equal to zero)

$$\mathrm{Flat}_{ab|cd}(P_0) = \begin{array}{c|cccc} & 00 & 01 & 10 & 11 \\ \hline 00 & p_{0000} & & & p_{0011} \\ 01 & & & & \\ 10 & & & & \\ 11 & p_{1100} & & & p_{1111} \end{array},$$

$$\mathrm{Flat}_{ac|bd}(P_0) = \begin{array}{c|cccc} & 00 & 01 & 10 & 11 \\ \hline 00 & p_{0000} & & & \\ 01 & & p_{0011} & & \\ 10 & & & p_{1100} & \\ 11 & & & & p_{1111} \end{array},$$

$$\mathrm{Flat}_{ad|bc}(P_0) = \begin{array}{c|cccc} & 00 & 01 & 10 & 11 \\ \hline 00 & p_{0000} & & & \\ 01 & & p_{0011} & & \\ 10 & & & p_{1100} & \\ 11 & & & & p_{1111} \end{array}.$$

In this form, we can clearly see the effect of Theorem 2.5.4 on the flattenings for this tree. That is, we can see that $\mathrm{Flat}_{ab|cd}(P_0)$ has rank $k = 2$ and $\mathrm{Flat}_{ac|bd}(P_0)$ and $\mathrm{Flat}_{ad|bc}(P_0)$ have rank $k^2 = 4$. Additionally, using Lemma 2.5.5 shows that replacing the identity matrices at the leaves with any other Markov matrices will not change the rank of each of the flattenings.

We now consider the same tree with zero-length leaf edges and corresponding flattenings for general k states, and make similar observations about the ranks in order to prove Theorem 2.5.4. The proof provided is by induction on the number of leaves, and dexamines flattening matrices directly. It is more closely related to the proof for the analogous condition for subflattenings given in [26] than the proof appearing in [11]. The proof provided makes use of the term 'divergence event', which refers to the divergence of a leaf into two new leaves. In terms of evolution, this means a species has diversified into two new ones; this is how evolutionary trees grow. We often re-use the label of the old leaf for one of the new ones arbitrarily, when there is no ambiguity. See Figure 3.1 for an example. We assume a general number of states, k, with state-space $\mathcal{S} = \{\kappa_1, ...\kappa_k\}$.

3.1 Proof of Flattening Rank Theorem

Recall Theorem 2.5.4, concerning the rank properties of flattenings:

Theorem 2.5.4. *Given a phylogenetic tree T and state space $\mathcal{S}$, the flattening matrix* $\mathrm{Flat}_{A|B}(P)$ *computed from a split $A|B$ has* $\mathrm{rank}(\mathrm{Flat}_{A|B}(P)) = k$ *if the split $A|B$ is displayed by T, and has* $\mathrm{rank}(\mathrm{Flat}_{A|B}(P)) \geq k^2$ *otherwise.*

To prove Theorem 2.5.4, we begin by proving the rank conditions hold for the base case: a tree with four leaves, known as a quartet tree. This is the smallest tree for which we have non-trivial splits. We then proceed by induction by assuming the result for an n taxon tree and showing the result holds after a divergence event, recognising that every $n+1$ taxon tree can be grown from an n taxon tree after one such event. This is a common method for proofs of results relating to trees [25]. Note that throughout this proof, statements about rank are referring to the generic rank (see Definition 2.5.3).

Proof. Take the true split $ab|cd$ on the quartet tree T with identity matrices on the leaves and site-pattern distribution P_0. Then, from Lemma 2.4.1,

$$\left(\mathrm{Flat}_{ab|cd}(P_0)\right)_{\kappa_i\kappa_j,\kappa_m\kappa_l} = p_{\kappa_i\kappa_j\kappa_m\kappa_l} = 0 \text{ whenever } \kappa_i \neq \kappa_j \text{ or } \kappa_m \neq \kappa_l.$$

This means that there are only k possible non-zero rows in the flattening (the rows indexed by $\kappa_1\kappa_1$, $\kappa_2\kappa_2$, $\kappa_3\kappa_3$, etc). That is, the rank of the flattening is at most k.

We can see that the rank is *no less than* k by observing the $k \times k$ sub-matrix $\mathrm{Flat}^{(k\times k)}_{ab|cd}(P_0)$ consisting of all the non-zero entries of $\mathrm{Flat}_{ab|cd}(P_0)$. This matrix can be thought of as a flattening for a split $x|y$ on a two-taxon tree with Markov matrices M_x and M_y on the leaves x and y. Then, assuming a root distribution π on this tree, we have,

$$\det(\mathrm{Flat}^{(k\times k)}_{ab|cd}(P_0)) = \det(M_x \mathrm{diag}(\pi) M_y^T) = (\pi_{\kappa_1} \dots \pi_{\kappa_k}) \det(M_x) \det(M_y^T) \neq 0,$$

where $\kappa_i \in \mathcal{S}$ for all i. By Lemma 1.1.3, we see that this submatrix has rank k, and thus the original flattening has rank k.

Now consider a false split on T (without loss of generality, $ac|bd$). We have,

$$\left(\mathrm{Flat}_{ac|bd}(P_0)\right)_{\kappa_i\kappa_j,\kappa_m\kappa_l} = p_{\kappa_i\kappa_m\kappa_j\kappa_l} = 0 \quad \text{ whenever } \kappa_i \neq \kappa_m \text{ or } \kappa_j \neq \kappa_l.$$

That is, the only non-zero entries are precisely the diagonal entries

$$\left(\mathrm{Flat}_{ac|bd}(P_0)\right)_{\kappa_i\kappa_j,\kappa_i\kappa_j},$$

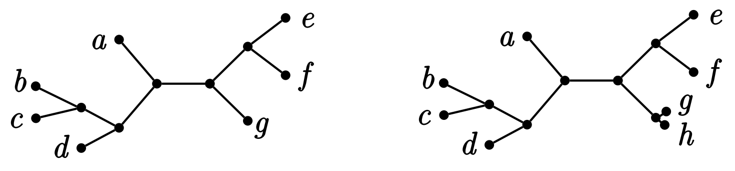


Figure 3.1: An example of a divergence event occurring on a tree, with the initial tree on the left, and the resulting tree on the right. The leaf g splits into two leaves, g and h, each with branch length zero.

and so we see that the flattening has rank k^2.

Now consider a tree with a general n leaves, take a split $A|B$ and assume the flattening satisfies the rank conditions (Theorem 2.5.4). Suppose a divergence event occurs on leaf n and that $n \in B$, giving a new taxon $n{+}1$. Note that the branch lengths on taxa n and $n{+}1$ are zero (see Figure 3.1 as an example).

We will first consider the case where the new taxon $n{+}1$ is assigned to B, giving a new split $A|B'$. We show that in this case the rank conditions still hold for the flattening on $A|B'$. Then, we will consider the case where the new taxon n is instead assigned to A, giving the new split $A'|B$. Since n and $n{+}1$ form a cherry, this new split is false, so we must show the rank of the flattening is greater than or equal to k^2. In this scenario, we consider two subcases: the case where $A|B$ was a true split, and the case where $A|B$ was a false split. For each of these cases, we will show that the flattening rank is greater than or equal to k^2, hence completing the proof.

Suppose taxon $n{+}1$ is assigned to B, and label the new set as B'. Then, in the flattening, we have for all $\kappa_i \in \mathcal{S}$ (using Lemma 2.4.1)

$$\big(\mathrm{Flat}_{A|B'}(P)\big)_{\kappa_1\ldots\kappa_i,\kappa_l\ldots\kappa_x\kappa_y} = \begin{cases} 0 & \text{if } \kappa_x \neq \kappa_y, \text{ or} \\ \big(\mathrm{Flat}_{A|B}(P)\big)_{\kappa_1\ldots\kappa_i,\kappa_l\ldots\kappa_x} & \text{otherwise.} \end{cases}$$

We can see that the flattening on $A|B'$ consists of the the same $k^{|B|}$ rows as the flattening on $A|B$, along with $k^{|B|}(k-1)$ additional zero columns (when $\kappa_x \neq \kappa_y$ for each κ_y as above, see (3.3) in Figure 3.3 for an example). The rank of the $A|B'$ flattening therefore is unchanged from the rank of the $A|B$ flattening following the divergence event: the rank conditions given in Theorem 2.5.4 again hold.

Now suppose we instead add the new taxon $n{+}1$ (which has diverged from $n \in B$) to the subset A. Label this new subset A'. Then, $A'|B$ is a false split regardless of whether or not $A|B$ was. We wish to show that the rank of the flattening is at least k^2.

Similar to the previous situation, we have (again using Lemma 2.4.1),

$$\left(\mathrm{Flat}_{A'|B}(P)\right)_{\kappa_1\ldots\kappa_i\kappa_j,\kappa_l\ldots\kappa_x} = \begin{cases} 0 & \text{if } \kappa_j \neq \kappa_x, \text{ or} \\ \left(\mathrm{Flat}_{A|B}(P)\right)_{\kappa_1\ldots\kappa_i,\kappa_l\ldots\kappa_x} & \text{otherwise.} \end{cases}$$

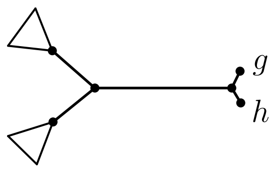
Carefully considering the row and column indexing, notice that each row in the old flattening 'expands' to k rows in the new flattening, with the existing entries 'stretched' down across the k rows, such that in each column, only one of these k rows has a non-zero entry (see (3.4) in Figure 3.3 for an example). We see that the span of the rows in the $A|B$-flattening is a subspace of the span of the rows in the $A'|B$ flattening. Thus from Definition 1.1.1, we have

$$\operatorname{rank}\left(\mathrm{Flat}_{A|B}(P)\right) \leq \operatorname{rank}\left(\mathrm{Flat}_{A'|B}(P)\right).$$

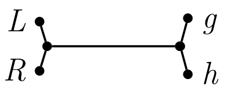
That is, the rank doesn't decrease after the divergence event. At this stage, we have proven by induction the following weaker property:

divergence events do not decrease the rank of the flattenings. (3.1)

This property will allow us to show that the rank of the new $A'|B$ flattening is at least k^2 regardless of whether $A|B$ was true or false. Let g and h be labels for the leaves which the n-th taxon diverges into, so as to align with the example given in Figure 3.2. In the new split $A'|B$, we suppose without loss of generality that $g \in A'$ and $h \in B$. Then, we know the tree looks like:



Then, we consider the quartet tree below, with zero-length leaf branches.



The split $Lg|Rh$ is a false one on this quartet tree, so we see from the base case that the corresponding flattening has rank k^2. We repeatedly deploy both Lemma 2.5.5

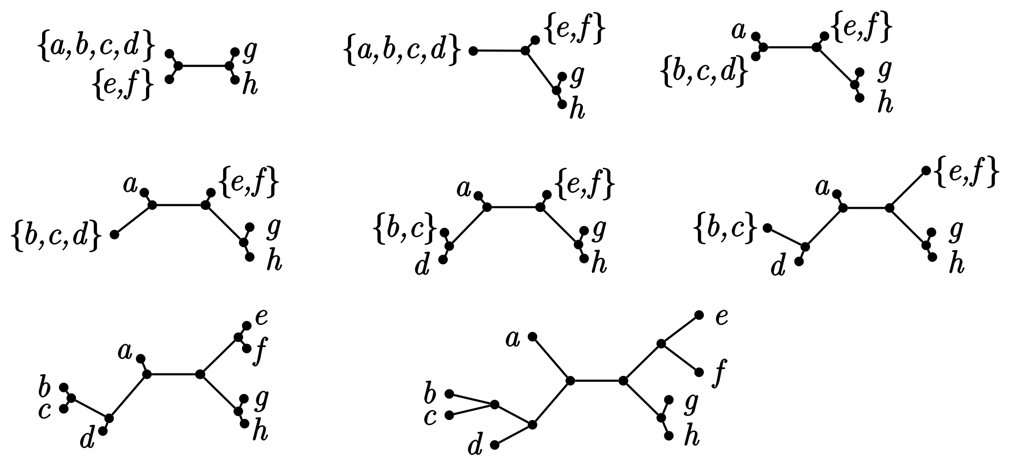


Figure 3.2: The steps showing a quartet tree grow into a larger tree, via divergence events and application of Markov matrices on the leaves. The rank of the flattening on a split $A|B$ with $g \in A, h \in B$ doesn't decrease at any point in this process.

and result (3.1) above, to grow the L and R leaves back into the required subtrees, assigning new leaves to the appropriate side of $Lg|Rh$ in accordance with $A'|B$ (please see Figure 3.2 for an example). We see that after growing the tree the flattening $\mathrm{Flat}_{A'|B}$ is at least rank k^2, since the rank doesn't decrease at any stage during this process. We have now shown that the rank conditions described in Theorem 2.5.4 hold in general. This completes the proof. ∎

$$\mathrm{Flat}_{12|34}(P) = \begin{array}{c|cccc} & 00 & 01 & 10 & 11 \\ \hline 00 & p_{0000} & p_{0001} & p_{0010} & p_{0011} \\ 01 & p_{0100} & p_{0101} & p_{0110} & p_{0111} \\ 10 & p_{1000} & p_{1001} & p_{1010} & p_{1011} \\ 11 & p_{1100} & p_{1101} & p_{1110} & p_{1111} \end{array} \tag{3.2}$$

$$\mathrm{Flat}_{12|345}(P) = \begin{array}{c|cccccccc} & 000 & 001 & 010 & 011 & 100 & 101 & 110 & 111 \\ \hline 00 & p_{0000} & & & p_{0001} & p_{0010} & & & p_{0011} \\ 01 & p_{0100} & & & p_{0101} & p_{0110} & & & p_{0111} \\ 10 & p_{1000} & & & p_{1001} & p_{1010} & & & p_{1011} \\ 11 & p_{1100} & & & p_{1101} & p_{1110} & & & p_{1111} \end{array} \tag{3.3}$$

$$\mathrm{Flat}_{125|34}(P) = \begin{array}{c|cccc} & 00 & 01 & 10 & 11 \\ \hline 000 & p_{0000} & & p_{0010} & \\ 001 & & p_{0001} & & p_{0011} \\ 010 & p_{0100} & & p_{0110} & \\ 011 & & p_{0001} & & p_{0111} \\ 100 & p_{1000} & & p_{1010} & \\ 101 & & p_{1001} & & p_{1011} \\ 110 & p_{1100} & & p_{1110} & \\ 111 & & p_{1101} & & p_{1111} \end{array} \tag{3.4}$$

Figure 3.3: The first of the above matrices is a flattening on a quartet tree. The next two matrices are are examples of flattenings directly following a divergence event, where taxon 4 splits into taxa 4 and 5. The flattening labelled (3.3) corresponds to the case where the new taxon is added to the left side of the split, and (3.4) corresponds to the case where the new taxon is added to the right side of the split.

Chapter 4

Subflattenings

4.1 Subflattening Construction

The construction of subflattenings begins with the similarity transformation S taking Markov matrices to matrices in the affine group $\mathrm{Aff}(k-1)$,

$$M \xmapsto{S} SMS^{-1} = \begin{bmatrix} T & v \\ 0 & 1 \end{bmatrix} \in \mathrm{Aff}(k-1),$$

as previously discussed in Section 1.2. Note that S is the matrix we introduced earlier in Definition 2.6.1. We will discuss options for choosing this matrix in Chapter 5.

The above transformation, applied to the Markov matrices used in the construction of flattenings, leads to the construction of transformed flattenings (again see Definition 2.6.1). The entries of the transformed flattening are given by,

$$q_{i_1 i_2 \dots i_{|X|}} := \sum_{j_1, j_2 \dots j_{|X|} \in \mathcal{S}} S_{i_1 j_1} S_{i_2 j_2} \dots S_{i_{|X|} j_{|X|}} p_{j_1 j_2 \dots j_{|X|}},$$

and are referred to in [26] as 'q-coordinates'. Their location within the transformed flattening depends on the split. The proof of the rank properties of subflattenings involve working directly with these linear combinations of site pattern probabilities, observing changes under divergence events [26]. From here, the identification of a representation of $\times^m \mathrm{Aff}(k-1)$ is what gives rise to subflattenings, obtained by taking the corresponding submatrix of the transformed flattening. We will discuss this more in Section 4.3.

4.2 Subflattening Equivalent Definition

Below is an equivalent definition of the sub-flattening, which can be created from the flattening in a single step. This construction is due to David Bryant (pers. commun.).

Theorem 4.2.1. *Given a flattening* $\mathrm{Flat}_{A|B}(P)$*, and an invertible* $k \times k$ *matrix* $\mathcal{S}$ *with a constant row of ones at the bottom, we have,*

$$\mathrm{Subfl}^{S}_{A|B}(P) = \overbrace{\begin{bmatrix} \hat{S} \otimes \mathbf{1} \otimes \ldots \otimes \mathbf{1} \\ \mathbf{1} \otimes \hat{S} \otimes \ldots \otimes \mathbf{1} \\ \mathbf{1} \otimes \mathbf{1} \otimes \ldots \otimes \mathbf{1} \\ \vdots \\ \mathbf{1} \otimes \mathbf{1} \otimes \ldots \otimes \hat{S} \\ \mathbf{1} \otimes \mathbf{1} \otimes \ldots \otimes \mathbf{1} \end{bmatrix}}^{|A|\ \textit{terms}} \mathrm{Flat}_{A|B}(P) \overbrace{\begin{bmatrix} \hat{S} \otimes \mathbf{1} \otimes \ldots \otimes \mathbf{1} \\ \mathbf{1} \otimes \hat{S} \otimes \ldots \otimes \mathbf{1} \\ \mathbf{1} \otimes \mathbf{1} \otimes \ldots \otimes \mathbf{1} \\ \vdots \\ \mathbf{1} \otimes \mathbf{1} \otimes \ldots \otimes \hat{S} \\ \mathbf{1} \otimes \mathbf{1} \otimes \ldots \otimes \mathbf{1} \end{bmatrix}}^{|B|\ \textit{terms}}{}^{T}, \qquad (4.1)$$

where $\hat{S}$ *is* S *with its row of ones removed, and* $\mathbf{1}$ *is the* $(1 \times k)$ *row vector of ones.*

Proof. We begin with the definition of the transformed flattening (Definition 2.6.1) and note that the matrix given in Equation (2.1) is converted into the subflattening via the removal of rows and columns indexed by strings which include at most one state in $\mathcal{S}$ that isn't some special state s, say. See Table 4.1 for an example.

For the proof, consider a general state-space $\mathcal{S} = \{\kappa_i : i = 1, ..., k\}$, and choose κ_k as the index for the constant row of ones in S and the 'special' state in the subflattening construction.

We notice that removing the required rows and columns from the transformed flattening is equivalent to removing those rows and columns from the matrices on the left and right of Definition 2.6.1 respectively. We examine the matrix on the left:

$$\left(\bigotimes_{i=1}^{|A|} S \right), \qquad (4.2)$$

and show that removing the required rows gives us the matrix on the left in Equation (4.1).

We index the above matrix by a lexicographical ordering of tuples of states, as

$\varkappa_1\varkappa_1\varkappa_1\varkappa_1$	$\varkappa_1\varkappa_2\varkappa_2\varkappa_3$	$\varkappa_2\varkappa_1\varkappa_1\varkappa_2$	$\varkappa_2\varkappa_2\varkappa_3\varkappa_1$	$\varkappa_3\varkappa_1\varkappa_1\varkappa_3$	$\varkappa_3\varkappa_2\varkappa_3\varkappa_2$
$\varkappa_1\varkappa_1\varkappa_1\varkappa_2$	$\varkappa_1\varkappa_2\varkappa_3\varkappa_1$	$\varkappa_2\varkappa_1\varkappa_1\varkappa_3$	$\varkappa_2\varkappa_2\varkappa_3\varkappa_2$	$\varkappa_3\varkappa_1\varkappa_2\varkappa_1$	$\boldsymbol{\varkappa_3\varkappa_2\varkappa_3\varkappa_3}$
$\varkappa_1\varkappa_1\varkappa_1\varkappa_3$	$\varkappa_1\varkappa_2\varkappa_3\varkappa_2$	$\varkappa_2\varkappa_1\varkappa_2\varkappa_1$	$\varkappa_2\varkappa_2\varkappa_3\varkappa_3$	$\varkappa_3\varkappa_1\varkappa_2\varkappa_2$	$\varkappa_3\varkappa_3\varkappa_1\varkappa_1$
$\varkappa_1\varkappa_1\varkappa_2\varkappa_1$	$\varkappa_1\varkappa_2\varkappa_3\varkappa_3$	$\varkappa_2\varkappa_1\varkappa_2\varkappa_2$	$\varkappa_2\varkappa_3\varkappa_1\varkappa_1$	$\varkappa_3\varkappa_1\varkappa_2\varkappa_3$	$\varkappa_3\varkappa_3\varkappa_1\varkappa_2$
$\varkappa_1\varkappa_1\varkappa_2\varkappa_2$	$\varkappa_1\varkappa_3\varkappa_1\varkappa_1$	$\varkappa_2\varkappa_1\varkappa_2\varkappa_3$	$\varkappa_2\varkappa_3\varkappa_1\varkappa_2$	$\varkappa_3\varkappa_1\varkappa_3\varkappa_1$	$\boldsymbol{\varkappa_3\varkappa_3\varkappa_1\varkappa_3}$
$\varkappa_1\varkappa_1\varkappa_2\varkappa_3$	$\varkappa_1\varkappa_3\varkappa_1\varkappa_2$	$\varkappa_2\varkappa_1\varkappa_3\varkappa_1$	$\varkappa_2\varkappa_3\varkappa_1\varkappa_3$	$\varkappa_3\varkappa_1\varkappa_3\varkappa_2$	$\varkappa_3\varkappa_3\varkappa_2\varkappa_1$
$\varkappa_1\varkappa_1\varkappa_3\varkappa_1$	$\varkappa_1\varkappa_3\varkappa_1\varkappa_3$	$\varkappa_2\varkappa_1\varkappa_3\varkappa_2$	$\varkappa_2\varkappa_3\varkappa_2\varkappa_1$	$\boldsymbol{\varkappa_3\varkappa_1\varkappa_3\varkappa_3}$	$\varkappa_3\varkappa_3\varkappa_2\varkappa_2$
$\varkappa_1\varkappa_1\varkappa_3\varkappa_2$	$\varkappa_1\varkappa_3\varkappa_2\varkappa_1$	$\varkappa_2\varkappa_1\varkappa_3\varkappa_3$	$\varkappa_2\varkappa_3\varkappa_2\varkappa_2$	$\varkappa_3\varkappa_2\varkappa_1\varkappa_1$	$\boldsymbol{\varkappa_3\varkappa_3\varkappa_2\varkappa_3}$
$\varkappa_1\varkappa_1\varkappa_3\varkappa_3$	$\varkappa_1\varkappa_3\varkappa_2\varkappa_2$	$\varkappa_2\varkappa_2\varkappa_1\varkappa_1$	$\varkappa_2\varkappa_3\varkappa_2\varkappa_3$	$\varkappa_3\varkappa_2\varkappa_1\varkappa_2$	$\boldsymbol{\varkappa_3\varkappa_3\varkappa_3\varkappa_1}$
$\varkappa_1\varkappa_2\varkappa_1\varkappa_1$	$\varkappa_1\varkappa_3\varkappa_2\varkappa_3$	$\varkappa_2\varkappa_2\varkappa_1\varkappa_2$	$\varkappa_2\varkappa_3\varkappa_3\varkappa_1$	$\varkappa_3\varkappa_2\varkappa_1\varkappa_3$	$\boldsymbol{\varkappa_3\varkappa_3\varkappa_3\varkappa_2}$
$\varkappa_1\varkappa_2\varkappa_1\varkappa_2$	$\varkappa_1\varkappa_3\varkappa_3\varkappa_1$	$\varkappa_2\varkappa_2\varkappa_1\varkappa_3$	$\varkappa_2\varkappa_3\varkappa_3\varkappa_2$	$\varkappa_3\varkappa_2\varkappa_2\varkappa_1$	$\boldsymbol{\varkappa_3\varkappa_3\varkappa_3\varkappa_3}$
$\varkappa_1\varkappa_2\varkappa_1\varkappa_3$	$\varkappa_1\varkappa_3\varkappa_3\varkappa_2$	$\varkappa_2\varkappa_2\varkappa_2\varkappa_1$	$\boldsymbol{\varkappa_2\varkappa_3\varkappa_3\varkappa_3}$	$\varkappa_3\varkappa_2\varkappa_2\varkappa_2$	
$\varkappa_1\varkappa_2\varkappa_2\varkappa_1$	$\boldsymbol{\varkappa_1\varkappa_3\varkappa_3\varkappa_3}$	$\varkappa_2\varkappa_2\varkappa_2\varkappa_2$	$\varkappa_3\varkappa_1\varkappa_1\varkappa_1$	$\varkappa_3\varkappa_2\varkappa_2\varkappa_3$	
$\varkappa_1\varkappa_2\varkappa_2\varkappa_2$	$\varkappa_2\varkappa_1\varkappa_1\varkappa_1$	$\varkappa_2\varkappa_2\varkappa_2\varkappa_3$	$\varkappa_3\varkappa_1\varkappa_1\varkappa_2$	$\varkappa_3\varkappa_2\varkappa_3\varkappa_1$	

Table 4.1: Row indexing of the flattening and transformed flattening on a split $A|B$ with $|A| = 4$, state space $\mathcal{S} = \{\varkappa_1, \varkappa_2, \varkappa_3\}$ and special state $\varkappa_3$. The labels in bold are the labels for the rows remaining in the subflattening.

described in Section 1.1,

$$\left(\bigotimes_{i=1}^{|A|} S \right)_{I,J} = S_{I_1,J_1} \dots S_{I_{|A|},J_{|A|}},$$

where I_i, J_i are the i-th entries of the tuples I and J respectively. We see that the last row of the above matrix, indexed by $I = (\varkappa_k, ...\varkappa_k)$, is a row of ones, since the last row of S (the row indexed by $\varkappa_k$) is a row of ones. This row is the same as the last row in the matrix on the left in Equation (4.1). We now consider a general row of the above matrix.

As the only rows we keep from this matrix are those which involve at most one non-$\varkappa_k$ state, we can denote our row indices (aside from the very last row) by $I^{(n,\varkappa_m)}$, indicating:

- the position of the single non-$\varkappa_k$ state, n, and
- which of the $k-1$ other states is in this position, $\varkappa_m$.

We note that the rows are ordered by the first parameter and then by the second.

Fix a row $I^{(n,\varkappa_m)}$. Since the state $\varkappa_k$ is the index of the row of ones in S, we have,

$$\begin{aligned}\left(\bigotimes_{i=1}^{|A|} S\right)_{I^{(n,\kappa_m)},J} &= S_{\kappa_k,J_1}\dots S_{\kappa_k,J_{n-1}} S_{\kappa_m,J_n} S_{\kappa_k,J_{n+1}}\dots S_{\kappa_k,J_{|A|}} \\ &= S_{\kappa_m,J_n}.\end{aligned}$$

Let $J_n^{(m)}$ denote the n-th entry of the m-th tuple, and let the column indices run. We see that row $I^{(n,\kappa_m)}$ looks like,

$$\left(S_{\kappa_m,J_n^{(1)}},\ \dots\ ,S_{\kappa_m,J_n^{k(|A|)}}\right). \tag{4.3}$$

Observe the matrix on the left in Equation (4.1) as a block-vector, and choose the n-th block. We look at the $(\kappa_m,\ J)$ entry of this block, for some J:

$$\left(\left(\bigotimes_{i=1}^{n-1} 1\right)\otimes S\otimes\left(\bigotimes_{i=n+1}^{|A|} 1\right)\right)_{\kappa_m,J} = 1\dots 1\cdot S_{\kappa_m,J_n}\cdot 1\dots 1 = S_{\kappa_m,J_n}.$$

Letting the row indices $J^{(i)}$ run, we see that the κ_m-th row in the above block is given by

$$\left(S_{\kappa_m,J_n^{(1)}},\ \dots\ ,S_{\kappa_m,J_n^{k(|A|)}}\right).$$

Which is the same as (4.3), and so we see that the two matrices are the same—the $I^{(n,\kappa_m)}$-row in the first matrix is the κ_m-row of the n-th block in the other, with order preserved. The argument for the matrices on the right in both expressions is similar, giving us the result. ■

4.3 Generalised Subflattenings

As previously discussed Section 2.6, the subflattening is a submatrix of the transformed flattening. In Section 4.1 we elaborated on this, describing the matrix S—used in the definition of the transformed flattening—in terms of a similarity transformation which takes Markov matrices to the affine group. We introduce some notation.

Definition 4.3.1. *We denote*

$$\widehat{\mathbf{M}}^{(m)} := \bigotimes_{i=1}^{m} \widehat{M}_i,$$

where

$$\widehat{M_i} = SM_iS^{-1} = \begin{bmatrix} T_i & v_i \\ 0 & 1 \end{bmatrix} \in \mathrm{Aff}(k-1),$$

for some Markov matrices M_i.

The submatrix of the transformed flattening on a split $A|B$ which defines the subflattening corresponds to submatrices of $\widehat{\mathbf{M}}^{(m)}$ and $\widehat{\mathbf{M}}^{(n)}$ which provide faithful representations of $\times^m \mathrm{Aff}(k-1)$ and $\times^n \mathrm{Aff}(k-1)$ respectively [26]. While these are the smallest such submatrices which give faithful representations, we show that they aren't the only ones.

Begin with the following definition:

Definition 4.3.2. *For $r < m$, define $\mathrm{D}(r,m)$ to be the sum of the first r terms in the binomial expansion of $k^m = ((k-1)+1)^m$—or equivalently, the last r terms in the expansion of $k^m = (1+(k-1))^m$. That is,*

$$\mathrm{D}(r,m) := \sum_{i=0}^{r} \binom{m}{i} (k-1)^i.$$

Example 4.3.3. *We recognise*

$$\mathrm{D}(1,m) = \binom{m}{0}(k-1)^0 + \binom{m}{1}(k-1) = m(k-1)+1,$$

as the form of the dimension of the sub-flattening, and

$$\mathrm{D}(m,m) = \sum_{i=0}^{m} \binom{m}{i} (k-1)^i (1)^{m-i} = ((k-1)+1)^m = k^m,$$

as the form of the dimension of the flattening.

This definition allows us to state and prove the following lemma.

Lemma 4.3.4. *For each $r < m$, there are $\mathrm{D}(r,m)$ rows in $\widehat{\mathbf{M}}^{(m)}$ which include only entries which are products of $\leq r$ entries from the blocks T_i and v_i within $\widehat{M_i}$ (that is, rows with entries which are monomials of degree no greater than r). Let $R := \{i_1, ..., i_{\mathrm{D}(r,m)}\}$ be the set of indices for these rows. Then, we have*

$$\widehat{\mathbf{M}}^{(m)}_{i_p,j} = 0,$$

for all $i_p \in R$ and $j \notin R$.

Proof. First note that for this proof, we index each $\widehat{M_i}$ as regular $k \times k$ matrices rather than 2×2 block matrices as they are described in Section 1.2. We can index $\widehat{\mathbf{M}}^{(m)}$ via m-tuples $I = (i_1, ..., i_m)$ and $J = (j_1, ..., j_m)$, where

$$\widehat{\mathbf{M}}^{(m)}_{I,J} = (\widehat{M_1})_{i_1,j_1} \cdot ... \cdot (\widehat{M_m})_{i_m,j_m}.$$

Then we see that generically, $\widehat{\mathbf{M}}^{(m)}_{I,J} = 0$ precisely when we have $(i_p, j_p) = (k, h),\ h \neq k$ for *at least one* $p = 1, ..., m$.

We also observe that the indices of rows in $\widehat{\mathbf{M}}^{(m)}$ which contain only elements with order $\leq r$ are those in the set

$$R = \{I = (i_1, ..., i_m) : i_p \neq k \text{ for at most } r \text{ values of } p = 1, ..., m\}.$$

Then a simple counting argument gives the number of such rows as

$$|R| = \sum_{i=0}^{r} \binom{m}{i} (k-1)^i = \mathrm{D}(r, m).$$

Now, consider the entry $\widehat{\mathbf{M}}^{(m)}_{I,J}$ where $I \in R$ and $J \notin R$. Then, J has more than r entries which are not equal to k. Since $r < m$, we then have some p such that $I_p = i_p = k$ and $J_p = j_p = h \neq k$. That is, $(i_p, j_p) = (k, h),\ h \neq k$ and so $\widehat{\mathbf{M}}^{(m)}_{I,J} = 0$ for all $I \in R$ and $J \notin R$, as required. ∎

This result leads us to the identification of the following class of faithful subrepresentations.

Theorem 4.3.5. *For each $r < m$, the matrix $\widehat{\mathbf{M}}^{(m)}$ has a $\mathrm{D}(r, m) \times \mathrm{D}(r, m)$ submatrix, the form of which provides a faithful representation of $\times^m \mathrm{Aff}(k-1)$.*

Proof. Again, let $R := \{i_1, ..., i_{\mathrm{D}(r,m)}\}$ be the collection of indices of rows in $\widehat{\mathbf{M}}^{(m)}$ which contain products of at most r non-zero entries from M_i matrices. Define the $1 \times k^m$ row vector w by

$$w_i = \mathbf{1}_R(i)\lambda_i = \begin{cases} \lambda_i & \text{if } i \in R, \text{ and,} \\ 0 & \text{otherwise,} \end{cases}$$

for some scalars $\lambda_i \in \mathbb{R}$. We see that w is a general element of the subspace $U := \mathrm{span}\{e_i : i \in R\}$ of $\mathbb{R}^{k^m}$ (where e_i is the $1 \times k^m$ row vector with 0 everywhere except for a 1 in position i).

Then, for $i \notin R$, we use Lemma 4.3.4 to obtain

$$\begin{aligned}
\left(w\widehat{\mathbf{M}}^{(m)}\right)_i &= \sum_{l=1}^{k^m} w_l \cdot \widehat{\mathbf{M}}^{(m)}_{l,i} \\
&= \sum_{l \in R} w_l \cdot \widehat{\mathbf{M}}^{(m)}_{l,i} + \sum_{l \notin R} w_l \cdot \widehat{\mathbf{M}}^{(m)}_{l,i} \\
&= \sum_{l \in R} \lambda_l \cdot 0 + \sum_{l \notin R} 0 \cdot \widehat{\mathbf{M}}^{(m)}_{l,i} \\
&= 0.
\end{aligned}$$

Implying $w\widehat{\mathbf{M}}^{(m)} \in U$. Since w is a general element in U, we see that U is an $\times^m \operatorname{Aff}(k-1)$-invariant subspace of $\mathbb{R}^{k^m}$. Thus the mapping of elements of $\times^m \operatorname{Aff}(k-1)$ to matrices with the form of the $\mathrm{D}(r,m) \times \mathrm{D}(r,m)$ sub-matrix is a subrepresentation of $\times^m \operatorname{Aff}(k–1)$, for all $r < m$. Furthermore, these representations are faithful, since each of them have the matrix corresponding to the subflattening representation as a submatrix. ■

We clarify this with an example,

Example 4.3.6. *Take the following three transformed Markov matrices,*

$$\begin{bmatrix} T_1 & v_1 \\ 0 & 1 \end{bmatrix}, \begin{bmatrix} T_2 & v_2 \\ 0 & 1 \end{bmatrix}, \begin{bmatrix} T_3 & v_3 \\ 0 & 1 \end{bmatrix}.$$

Taking the Kronecker product of the first two matrices, we get:

$$\begin{bmatrix} T_1 & v_1 \\ 0 & 1 \end{bmatrix} \otimes \begin{bmatrix} T_2 & v_2 \\ 0 & 1 \end{bmatrix} \sim \begin{bmatrix} T_1 \otimes T_2 & T_1 \otimes v_2 & v_1 \otimes T_2 & v_1 \otimes v_2 \\ 0 & T_1 & 0 & v_1 \\ 0 & 0 & T_2 & v_2 \\ 0 & 0 & 0 & 1 \end{bmatrix}.$$

First, note that we don't have equality above, but rather the matrix on the RHS is permutation similar to the LHS, as is the case for taking Kronecker products of block matrices (see Section 1.1).

We see that the bottom-right 3×3 block in the above matrix is the 'subflattening representation'. If we take the Kronecker product of the above matrix with the third Markov matrix, we have,

$$\begin{bmatrix}
T_1{\otimes}T_2{\otimes}T_3 & T_1{\otimes}T_2{\otimes}v_3 & T_1{\otimes}v_2{\otimes}T_3 & T_1{\otimes}v_2{\otimes}v_3 & v_1{\otimes}T_2{\otimes}T_3 & v_1{\otimes}T_2{\otimes}v_3 & v_1{\otimes}v_2{\otimes}T_3 & v_1{\otimes}v_2{\otimes}v_3 \\
0 & T_1{\otimes}T_2 & 0 & T_1{\otimes}v_2 & 0 & v_1{\otimes}T_2 & 0 & v_2{\otimes}v_3 \\
0 & 0 & T_1{\otimes}T_3 & T_1{\otimes}v_3 & 0 & 0 & v_1{\otimes}T_3 & v_1{\otimes}v_3 \\
0 & 0 & 0 & T_1 & 0 & 0 & 0 & v_1 \\
0 & 0 & 0 & 0 & T_2{\otimes}T_3 & T_2{\otimes}v_3 & v_2{\otimes}T_3 & v_2{\otimes}v_3 \\
0 & 0 & 0 & 0 & 0 & T_2 & 0 & v_2 \\
0 & 0 & 0 & 0 & 0 & 0 & T_3 & v_3 \\
0 & 0 & 0 & 0 & 0 & 0 & 0 & 1
\end{bmatrix}.$$

After some simultaneous row and column permutations, we get,

$$\begin{bmatrix}
T_1{\otimes}T_2{\otimes}T_3 & T_1{\otimes}T_2{\otimes}v_3 & T_1{\otimes}v_2{\otimes}T_3 & v_1{\otimes}T_2{\otimes}T_3 & T_1{\otimes}v_2{\otimes}v_3 & v_1{\otimes}T_2{\otimes}v_3 & v_1{\otimes}v_2{\otimes}T_3 & v_1{\otimes}v_2{\otimes}v_3 \\
0 & T_1{\otimes}T_2 & 0 & 0 & T_1{\otimes}v_2 & v_1{\otimes}T_2 & 0 & v_2{\otimes}v_3 \\
0 & 0 & T_1{\otimes}T_3 & 0 & T_1{\otimes}v_3 & 0 & v_1{\otimes}T_3 & v_1{\otimes}v_3 \\
0 & 0 & 0 & T_2{\otimes}T_3 & 0 & T_2{\otimes}v_3 & v_2{\otimes}T_3 & v_2{\otimes}v_3 \\
0 & 0 & 0 & 0 & T_1 & 0 & 0 & v_1 \\
0 & 0 & 0 & 0 & 0 & T_2 & 0 & v_2 \\
0 & 0 & 0 & 0 & 0 & 0 & T_3 & v_3 \\
0 & 0 & 0 & 0 & 0 & 0 & 0 & 1
\end{bmatrix}.$$

Again, we can see the 'subflattening representation' appearing as the 4×4 *block matrix—actually a* $3(k-1)+1 \times 3(k-1)+1$ *matrix. We can see also that the bottom-right* 7×7 *block matrix—really a* $(3(k-1)^2 + 3(k-1)+1)$*-row matrix—is another sub-representation. Setting aside the proof of Theorem 4.3.5, one can check that specific cases are representations of* $\times^m \operatorname{Aff}(k-1)$ *by comparing the product of any two matrices to the product defined on* $\times^m \operatorname{Aff}(k-1)$*.*

Take the following two matrices,

$$T = \bigotimes_{i=1}^{3} \begin{bmatrix} T_i & v_i \\ 0 & 1 \end{bmatrix},$$

and

$$U = \bigotimes_{i=1}^{3} \begin{bmatrix} U_i & u_i \\ 0 & 1 \end{bmatrix}.$$

We look at one particular block entry of the matrix TU,

$$\begin{aligned}(TU)_{3,5} &= (T_1 \otimes T_3)(U_1 \otimes u_3) + (T_1 \otimes v_3)U_1 \\ &= (T_1U_1 \otimes T_3u_3) + (T_1U_1 \otimes v_3) \\ &= T_1U_1 \otimes (v_3 + T_3u_3).\end{aligned}$$

Taking the corresponding elements of $\times^3 \operatorname{Aff}(k-1)$,

$$\underline{T} = ((T_1, v_1), (T_2, v_2), (T_3, v_3)),$$

and

$$\underline{U} = ((U_1, u_1), (U_2, u_2), (U_3, u_3)),$$

and using the multiplication in the group, we have

$$\begin{aligned}\underline{TU} &= ((T_1, v_1)(U_1, u_1), (T_2, v_2)(U_2, u_2), (T_3, v_3)(U_3, u_3)) \\ &= ((T_1U_1, v_1 + T_1u_1), (T_2U_2, v_2 + T_2u_2), (T_3U_3, v_3 + T_3u_3)).\end{aligned}$$

Under the transformation from the group to the set of matrices of the above form, the (3,5) entry of the corresponding matrix would be

$$T_1U_1 \otimes (v_3 + T_3u_3).$$

Which is the same as above. In other words, letting ρ be the the map which transforms elements of the group to matrices of the above form, we have

$$(\rho(\underline{TU}))_{3,5} = (\rho(\underline{T})\rho(\underline{U}))_{3,5}.$$

The other entries can be checked, giving the representation we expect from the theorem.

Having identified the sub-representations of $\times^m \operatorname{Aff}(k-1)$ in Theorem 4.3.5, we can construct corresponding matrices from site-pattern distributions in a way that is analogous to the construction of subflattenings.

Recalling Definition 2.6.2, the subflattening is defined as the submatrix of the transformed flattening (Definition 2.6.1) with rows and columns indexed by tuples containing at most one state other than a specially selected state, κ_k, say. That is, the rows or columns in the set:

$$R = \{I = (i_1, ..., i_m) : i_p \neq \kappa_k \text{ for at most 1 value of } p = 1, ..., m\}.$$

Where $m = |A|$ for selecting the rows of the subflattening, or $|B|$ for selecting columns.

This also describes precisely the collection of rows and columns of $\widehat{\mathbf{M}}^{(m)}$ which defines the 'subflattening representation' which we call $D(1, m)$ (where the index k is equivalent to the index κ_k. From the proof of Theorem 4.3.5, we see that

$$R = \{I = (i_1, ..., i_m) : i_p \neq k \text{ for at most } r \text{ values of } p = 1, ..., m\},$$

is the corresponding set of rows for the *general* $D(r, m)$ representation. This leads to the following definition.

Definition 4.3.7 ((r, c)-subflattening)**.** *Let* $\mathrm{Flat}^S_{A|B}(P)$ *be a transformed flattening with rows and columns indexed by string of characters from* $\mathcal{S}$ *with length* $|A|$ *and* $|B|$ *respectively. Choose a state* $\kappa_k \in \mathcal{S}$*, and consider the sets of indices:*

$$R_r := \{I = (i_1, ..., i_{|A|}) : i_p \neq \kappa_k \textit{ for at most } r \textit{ values of } p = 1, ..., |A|\},$$

and,

$$R_c := \{I = (i_1, ..., i_{|B|}) : i_p \neq \kappa_k \textit{ for at most } c \textit{ values of } p = 1, ..., |B|\},$$

with $r \leq |A|$ *and* $c \leq |B|$*. We define the* (r, c)*-subflattening,* $\mathrm{Subfl}^{(S,r,c)}_{A|B}(P)$*, as the* $D(r, |A|) \times D(c, |B|)$ *sub-matrix consisting of entries* $\left(\mathrm{Flat}^S_{A|B}(P)\right)_{\alpha\beta}$ *where* $\alpha \in R_r$ *and* $\beta \in R_c$*. We refer to these* (r, c)*-subflattenings as* ***generalised subflattenings***

We see that the $(1, 1)$-subflattening is the subflattening as defined in Definition 2.6.2, and that the $(|A|, |B|)$-subflattening is the transformed flattening defined in Definition 2.6.1.

We make the following conjecture about the rank of the (r, c)-subflattenings, which is analogous to the corresponding flattening and subflattening rank conditions (Theorem 2.5.4 and Theorem 2.6.3).

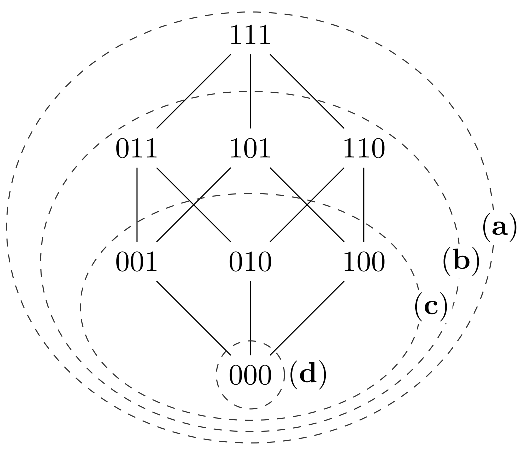


Figure 4.1: The lattice structure of the subflattening row or column indices. Here, 1 is the 'special' state and 0 represents other states. For example if k is the special state, 101 corresponds to words ksk for $s \neq k$. (**a**) highlights the possible indices for the flattening, (**b**) the $(2,2)$-subflattening and (**c**) the $(1,1)$-subflattening. (**d**) corresponds to the non-faithful representation which maps group elements to the 1×1 identity matrix. To visualise indexing in this way for a $(3,2)$-subflattening we would draw the lattice (**a**) to represent the possible column labellings, and (**b**) for the possible row labellings.

Conjecture 4.3.8 ((r, c)-subflattening rank)**.** *Let T be a tree with site pattern probability distribution P. Let $A|B$ be a split. A (r, c)-subflattening,* $\mathrm{Subfl}_{A|B}^{(S,r,c)}(P)$ *has rank k if $A|B$ is displayed by T, and rank greater than or equal to*

$$\min\{\mathrm{D}(r', 2), \mathrm{D}(c', 2)\},$$

otherwise, where $r' = \min\{r, 2\}$ and $c' = \min\{c, 2\}$.

Note that Conjecture 4.3.8 generalises Theorem 2.5.4 and Theorem 2.6.3, since in the case of a transformed flattening on a split $A|B$, we have

$$\min\{\mathrm{D}(r', 2), \mathrm{D}(c', 2)\} = \min\{\mathrm{D}(|A|, 2), \mathrm{D}(|B|, 2)\} = \mathrm{D}(2, 2) = k^2,$$

and for regular subflattenings, we have,

$$\min\{\mathrm{D}(r', 2), \mathrm{D}(c', 2)\} = \min\{\mathrm{D}(1, 2), \mathrm{D}(1, 2)\} = \mathrm{D}(1, 2) = 2(k-1) + 1.$$

We do not currently have a formal proof, but claim that the proof of the subflattening rank condition given in [26] with minor modifications will be applicable. Table 4.2 shows the ranks for some (r, c)-subflattenings on an example tree T_6. We perform some simulations using these generalised subflattenings in Chapter 5.

4.4 The Use of Subflattenings for Inference

Allman et al. [4] discuss a concern for using flattenings and subflattenings for phylogenetic inference—that splits with different *balance* are not comparable [4]. The paper showed that splits which were less balanced tended to have higher scores. That is, they were further from the closest rank k matrix. Simply put, the reason for this is that flattening matrices on more balanced splits are more square, and those on less balanced splits are more rectangular. Flattenings on more balanced splits have a higher maximum possible rank, since the maximum possible rank for an $m \times n$ matrix is $\min\{m, n\}$.

Allman et al. describe this concern as a reason for poor performance of Eriksson SVD algorithm provided in [11]. One may suppose that subflattenings are less impacted by this problem, since their reduced dimensions mean that the difference in shape of the subflattenings on splits with different balance is less dramatic compared to flattenings on the same splits. It is also possible that generalised subflattenings can be used to mitigate some of this bias, by making strategic choices of r and c. Evaluation of such a hypothesis would require simulations on large trees, where differences in split balance is more pronounced. As the code

Split	Pars.	Flat.	(1,1)	(1,2)	(2,1)	(2,2)
0123|45	1	4	4	4	4	4
012|345	1	4	4	4	4	4
01|2345	1	4	4	4	4	4
0124|35	2	16	7	7	7	16
0125|34	2	16	7	7	7	16
0134|25	2	16	7	10	7	16
0135|24	2	16	7	10	7	16
013|245	2	16	7	7	7	16
0145|23	2	16	7	7	7	16
014|235	2	16	7	7	10	16
015|234	2	16	7	7	10	16
0234|15	2	16	7	10	7	16
0235|14	2	16	7	10	7	16
023|145	2	16	7	10	7	16
0245|13	2	16	7	10	7	16
02|1345	2	16	7	7	7	16
0345|12	2	16	7	7	7	16
03|1245	2	16	7	7	10	16
045|123	2	16	7	7	10	16
04|1235	2	16	7	7	10	16
05|1234	2	16	7	7	10	16
024|135	3	64	10	10	10	37
025|134	3	64	10	10	10	37
034|125	3	64	10	10	10	37
035|124	3	64	10	10	10	37

Table 4.2: Ranks of flattenings and subflattenings on a particular 6-taxon tree (T_6) using the calculated site-pattern probability distribution (equivalent to an infinite sequence length). Note that the rank here is calculated based on a rank deficiency threshold, in particular the threshold used in implementations of matrix rank functions in *NumPy* and *MATLAB*. The 'Pars.' column shows parsimony scores for each split (see Definition 2.2.6). This table provides some small amount of evidence of the rank condition given in Conjecture 4.3.8.

used for simulations in this paper is not yet optimised for larger trees, we don't address this here. Regardless, we conjecture that the options for choices of r and c are not fine-grained enough to be of use for this purpose.

Another concern is that that the rank properties of flattenings and subflattenings may be inherently related to parsimony methods, which as discussed are known to be biased. The generic rank conditions for subflattenings given in Theorem 2.6.3 can be stated in a more specific way, referencing the parsimony score of the split in question [26]:

Theorem 4.4.1. *Let T be a tree with site pattern probability distribution P. Let $A|B$ be a split. A subflattening matrix $\mathrm{Subfl}^{S}_{A|B}(P)$ has rank k if $A|B$ is displayed by T, and rank $r(k-1)+1$ otherwise, where r is the parsimony score of the split $A|B$.*

Proof. See [26]. ■

The flattening rank property given in [11] is linked to parsimony score for splits in a similar way, although parsimony score is not explicitly mentioned. Evidence of this connection to parsimony is visible in Table 4.2, as well as in results from our simulations in Chapter 5, where one can see that scores of splits are clearly separated by the parsimony scores for the splits in question. The parsimony score for the split in question has a much larger impact on the score than the balance of the split.

Chapter 5

Simulations and Analysis

In this chapter, we discuss the simulations we have conducted and the analysis we have undertaken in order to evaluate the performance of flattenings and subflattenings. All the simulations and analyses detailed here were performed using the Python programming language. Python packages utilised include *Pandas*, *numpy* and *SciPy*. The python package which implements flattenings, subflattenings and a number of other concepts explored in this thesis has been provided as open source on GitHub[1] along with the code written to perform all the simulations and analyses detailed in this chapter. This package has been in development since 2018, with significant improvements made for its use in the work described here, in terms of speed and efficiency as well as maintainability. It is hoped that the software will further improve through continued development.

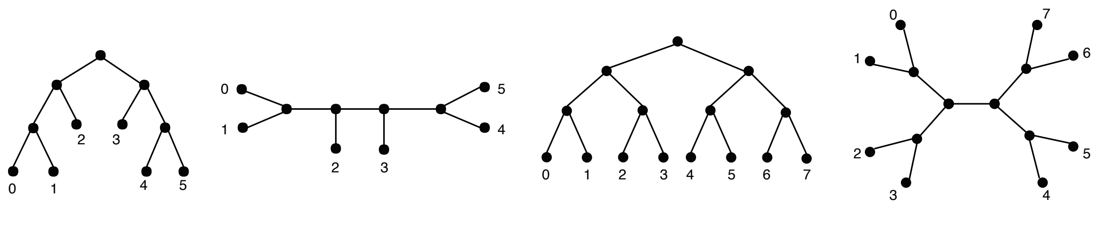


Figure 5.1: Rooted and unrooted versions of the balanced trees used in simulations. We refer to the 6-taxon tree as T_6 and the 8-taxon tree as T_8. True splits are those corresponding to internal edges (see Definition 2.2.3). For example, the true splits on T_6 are 01|2345, 012|345 and 0123|45.

[1]`https://github.com/js51/SplitP`

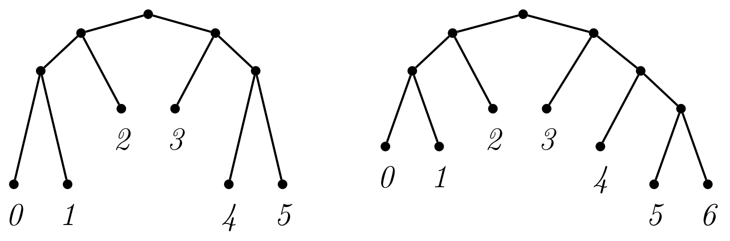


Figure 5.2: Trees used in simulations which are more difficult to infer. We refer to the left tree as T_{6b} and the right as T_7. The difficulty in inferring these trees correctly comes from the short internal branches compared to the branches at the leaves. The difference in these branch lengths is larger in T_{6b}.

5.1 Simulations: Choices of S Matrix

The construction of subflattening matrices is dependent on a choice of matrix S, as detailed in Definition 2.6.2 and in Chapter 4. As mentioned, the only requirement for this matrix is that it transforms Markov matrices to matrices in the affine group,

$$M \overset{S}{\longmapsto} SMS^{-1} = \begin{bmatrix} T & v \\ 0 & 1 \end{bmatrix} \in \mathrm{Aff}(k-1).$$

The only properties required for this are that S is invertible with a constant row. As mentioned in [26], it is not known whether different choices of matrix S will have an effect in practice, or whether any choice of S is optimal. It was however suggested that choosing S to be orthogonal may be appropriate. One possible choice for S is the appropriately sized Hadamard matrix (Section 2.8). We began our investigation into these choices of S with the Hadamard matrix, and scaled versions of it. We include the additional constraint that S had a constant row of ones however, and choose this row to be the final row, so as to align with the assumptions of the alternate definition of the subflattening (Theorem 4.2.1).

For brevity, denote the 4×4 Hadamard matrix

$$H = \begin{bmatrix} 1 & -1 \\ 1 & 1 \end{bmatrix} \otimes \begin{bmatrix} 1 & -1 \\ 1 & 1 \end{bmatrix} = \begin{bmatrix} 1 & -1 & -1 & 1 \\ 1 & 1 & -1 & -1 \\ 1 & -1 & 1 & -1 \\ 1 & 1 & 1 & 1 \end{bmatrix},$$

and our scaled Hadamard matrix,

$$H^{(\lambda)} = \begin{bmatrix} \lambda & -\lambda & -\lambda & \lambda \\ \lambda & \lambda & -\lambda & -\lambda \\ \lambda & -\lambda & \lambda & -\lambda \\ 1 & 1 & 1 & 1 \end{bmatrix}. \tag{5.1}$$

We begin by simply plotting split scores (see Definition 2.7.1) against splits using the flattening, and subflattenings constructed with some different S matrices, on a balanced six-taxon tree (tree T_6, see Figure 5.1). The initial matrices tested included the standard Hadamard matrix, the standard Hadamard matrix scaled by $\frac{1}{2}$ (so as to make it orthogonal), the matrix $H^{(2)}$, and the identity matrix with its last row replaced with a row of ones (which we call "Alt Identity"). The identity matrix was also used as an *invalid* choice of the matrix S, for comparison purposes (this matrix is not valid because of its lack of a constant row). Results of this initial simulation are shown in Figure 5.3. It is clear from these results that different, valid choices of the matrix S could lead to very different results. Keeping in mind that the first three splits in Figure 5.3 are the true ones, the flattening and the Hadamard based subflattening appeared to be the best choices here, since they separated true splits from false ones fairly well. As expected, the invalid identity matrix based subflattening was the worst performer, giving very low (good) scores for some false splits, for example 0145|23.

Next, the same simulation was repeated with only flattenings and some subflattenings using different values of λ in $S = H^{(\lambda)}$. Changing λ primarily affected the higher-scoring splits, increasing their scores (see Figure 5.4). To simulate real data, we used the calculated site-pattern probabilities to generate a table of empirical probabilities by drawing from the corresponding multinomial distribution, as is standard for generating sequence data [15, 26]. A 'sequence length' of 1000 saw similar results for subflattenings, whereas flattening scores for true splits appeared to rise more dramatically (Figure 5.4). Performing the above simulations on different trees gave similar results, with increased variance for the more difficult trees.

The next step was to attempt to quantify the performance of a given matrix S, so that a higher number of possible matrices could be compared more clearly. We proposed that a good choice of matrix S might increase the separation between scores for true splits and scores for false splits.

We considered the following measures:

- **Performance measure 1:** The absolute difference between the highest-scoring

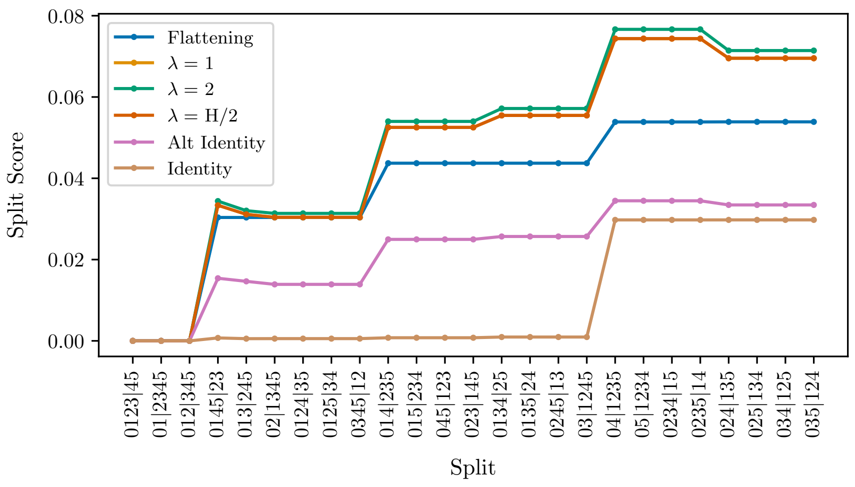


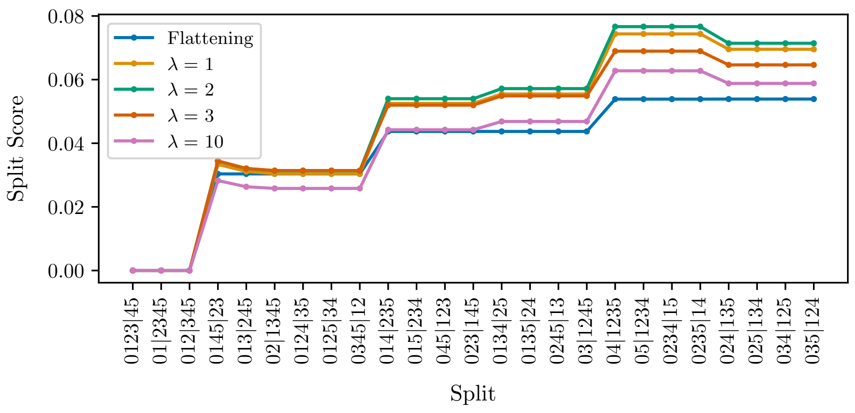
Figure 5.3: Split scores for all splits on tree T_6, varying the matrix S over H ($\lambda = 1$), $H^{(2)}$ ($\lambda = 2$), $\frac{1}{2}H$ (see Equation (5.1)), the identity matrix, and the identity matrix with its bottom row replaced by a row of ones ("Alt Identity").

true-split and the lowest-scoring 'false' split (i.e; the minimum difference in scores between the true splits and the false splits).

- **Performance measure 2:** the above measure divided by the difference in score between the lowest-scoring split and the highest-scoring split.

We then performed simulations calculating the above measures for collections of split-scores on a range of trees, for scaled H matrices $S = H^{(\lambda)}$, $\lambda \in [-5, 5]$, in steps of 0.25. We repeated this for $\lambda \in [-20, 20]$ in steps of 1.

From the results shown in Figure 5.5 we can see that, assuming our performance measures are indicative of real performance, selecting $\lambda \geq 3.5$ or $\lambda \leq 1$ is a poor choice. Looking at the best performing scaling factor across both plots, we suppose that based on these measures, somewhere between 2.75 and 3.5 may be optimal choice for λ. Performing the same simulation on T_8, T_{6b} and T_7 gave similar results. Note that in this simulation, we computed scores for two scaling factors very close to 0, in order to avoid errors appearing in the code at $\lambda = 0$. From the subflattening definition given in Theorem 4.2.1, one can see that setting $\lambda = 0$ gives a subflattening with every entry equal to zero aside from a 1 in the bottom right entry. That is, the rank of the subflattening will be exactly 1, regardless of the split.



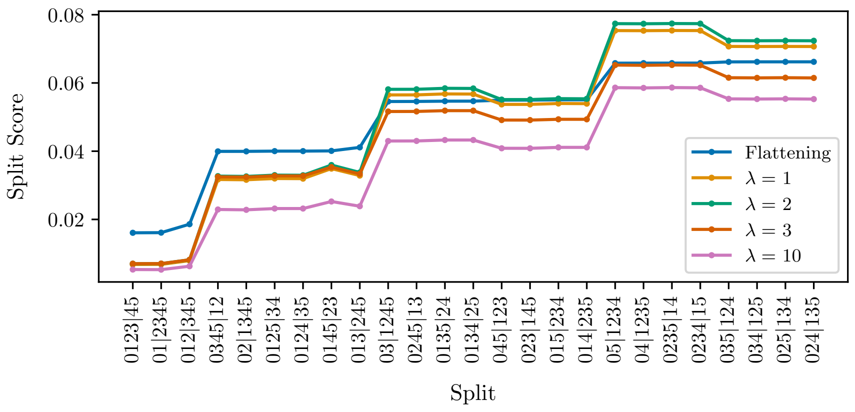


Figure 5.4: **Top:** split scores for all splits on tree T_6 for different scaled Hadamard matrices (Equation (5.1)) using calculated site-pattern probabilities (Section 2.4). **Bottom:** the same as above, but split scores are calculated from empirical site-pattern probabilities drawn from a multinomial distribution (sequence length 1000). The four clear 'steps' correspond to the parsimony scores for splits considered as a binary character on the true tree (see Definition 2.2.6).

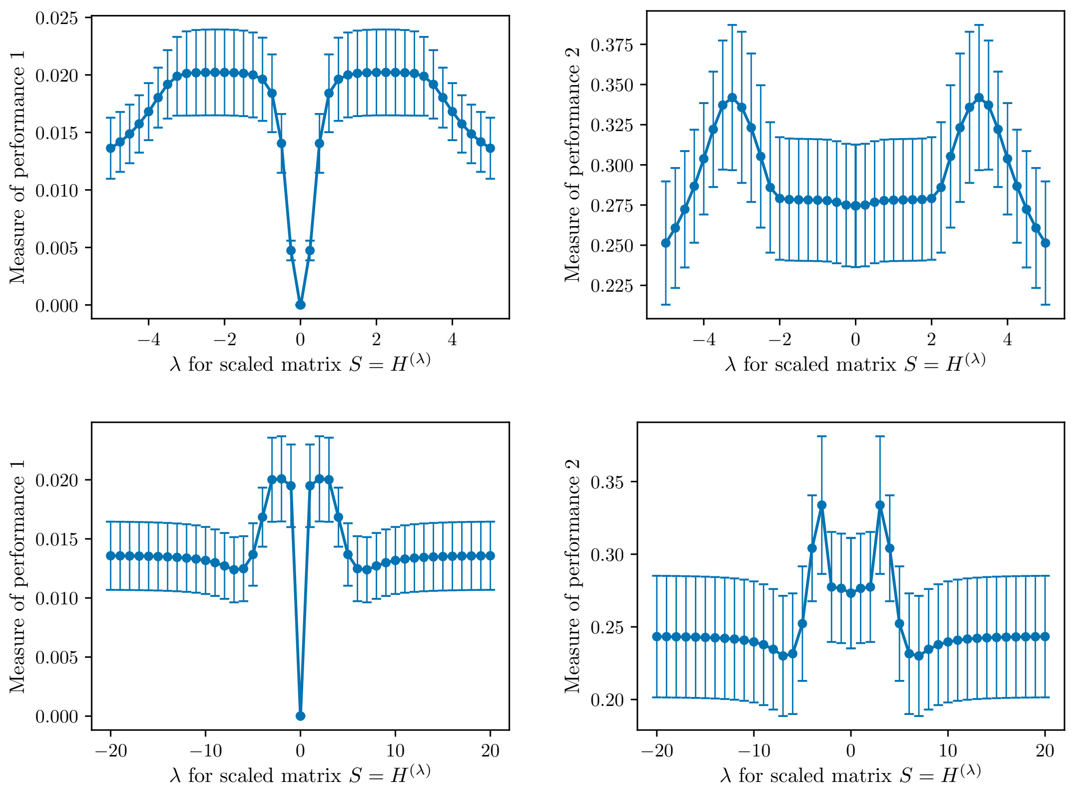


Figure 5.5: Measures of performance on tree T_6. Sequences were drawn from a multinomial distribution (sequence length 1000), 1000 times. For each draw, each one of the 22 subflattenings were constructed, and the average of the split-scores are shown above. The error bars indicate sample variance of the full list of 1000 scores for each scaling factor. The lower two plots are over a larger range of λ values.

The next step was to attempt to determine whether the performance difference we expect from Figure 5.5 is noticeable in practice.

5.2 Simulations: Eriksson's SVD Algorithm

The next simulation performed involved running each of the subflattenings tested so far through Eriksson's SVD algorithm on simulated pattern probabilities, in order to determine whether the previous results can be seen in practice. That is, we wish to determine whether the improvement from scaling Hadamard matrices is noticeable when using subflattenings for inference. We used the SVD algorithm

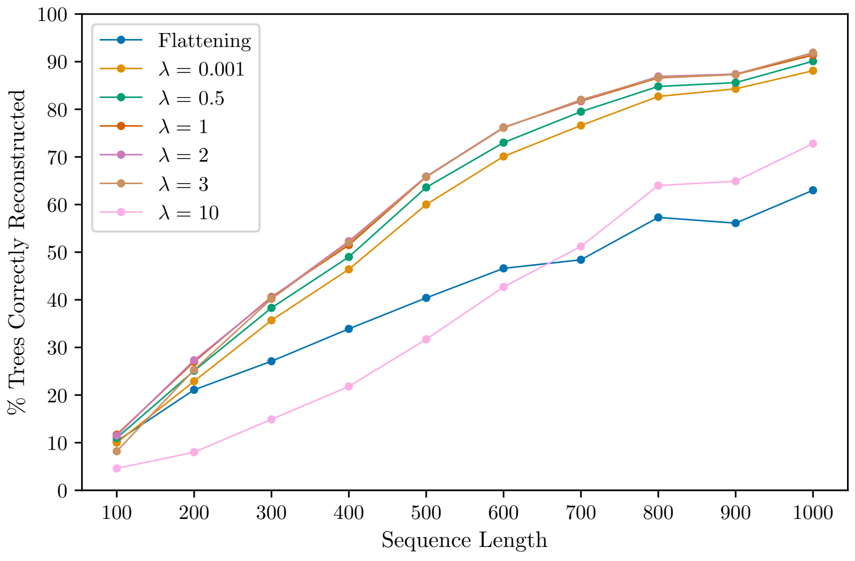


Figure 5.6: Reconstructing tree T_7 using subflattenings with differently scaled S matrices. The percentage of correctly reconstructed trees is almost the same for subflattenings with $\lambda = 1, 2$ and 3.

as it is described in [11], but used the split score defined in [4], instead of using singular values directly.

We selected a more 'difficult' tree (tree T_7) as well as some particular values of λ to test using the SVD algorithm. For each sequence length between 100 and 1000, we drew empirical pattern probabilities 1000 times, each time recording which of the subflattenings were able to correctly reconstruct the tree using the SVD algorithm. The percentage of times the tree was correctly reconstructed by each of the subflattenings, for each of the sequence lengths, is recorded in Figure 5.6.

The results in Figure 5.6 show that, using the SVD algorithm on tree T_7, subflattenings performed much better than flattenings at sequence lengths higher above 200, except when a high value of 10 was chosen for λ. Choosing λ to be less than 1 gave worse performing subflattenings, and the best choices appeared to be $\lambda = 1, 2$ or 3, with no noticeable performance difference between them.

We re-ran the simulation on a more difficult tree (tree T_{6b}) with similar results (see Figure 5.7).

Our next step was to run this simulation again on some more difficult trees. We coded the four quartet trees with varying branch lengths shown in Figure 5.8. The 'shorter' branches were given substitution matrices with a substitution probability

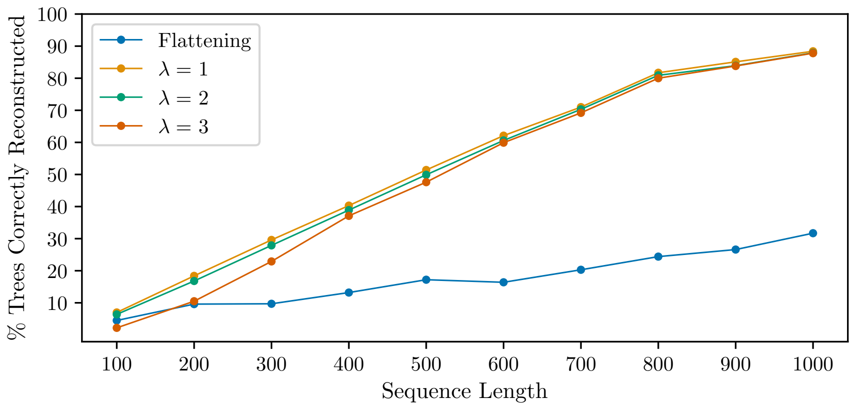


Figure 5.7: Reconstructing tree T_{6b} using subflattenings with differently scaled S matrices. The percentage of correctly reconstructed trees is almost the same for subflattenings with $\lambda = 1, 2$ and 3.

of 0.05, the substitution matrices for the remaining branches were the product of several copies of the previous, giving a substitution probability of approx. 0.33 along those edges. The results of this simulation are shown in Figure 5.9.

From Figure 5.9, we see that subflattenings performed far better than flattenings on quartet tree T_{4b}, and that flattenings performed slightly better on tree T_{4d}. This indicates that compared to flattenings, subflattenings are somehow less susceptible to being biased by long branch attraction, a term used to describe the tendency of some inference methods mistakenly grouping together leaves with long branches, despite the fact they are separated by an internal edge. The subflattening with $\lambda = 10$ again performed poorly across all the quartet trees, and again we saw the remaining subflattenings show similar performance. This time, the figure for T_{4d} showed some more separation between $\lambda = 1, 2$ and 3, with $\lambda = 1$ performing slightly better.

5.3 Simulations: Generalised Subflattenings

We performed some simulations to help to verify the rank conditions of generalised subflattenings, and to check their performance compared to flattenings and standard $(1, 1)$-subflattenings.

First, we computed exact site-pattern probabilities on tree T_6 and plotted split scores for various generalised subflattenings for each split, in the same way we did for subflattenings using scaled S matrices in Section 5.1. Results for T_6 are shown

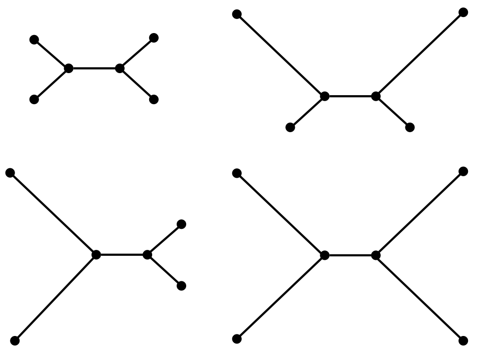


Figure 5.8: Quartet Trees which we label T_{4a}, T_{4b}, T_{4c} and T_{4d} left-to-right. The 'shorter' branches have a substitution probability of 0.05, the remaining branches have substitution probabilities of approximately 0.33.

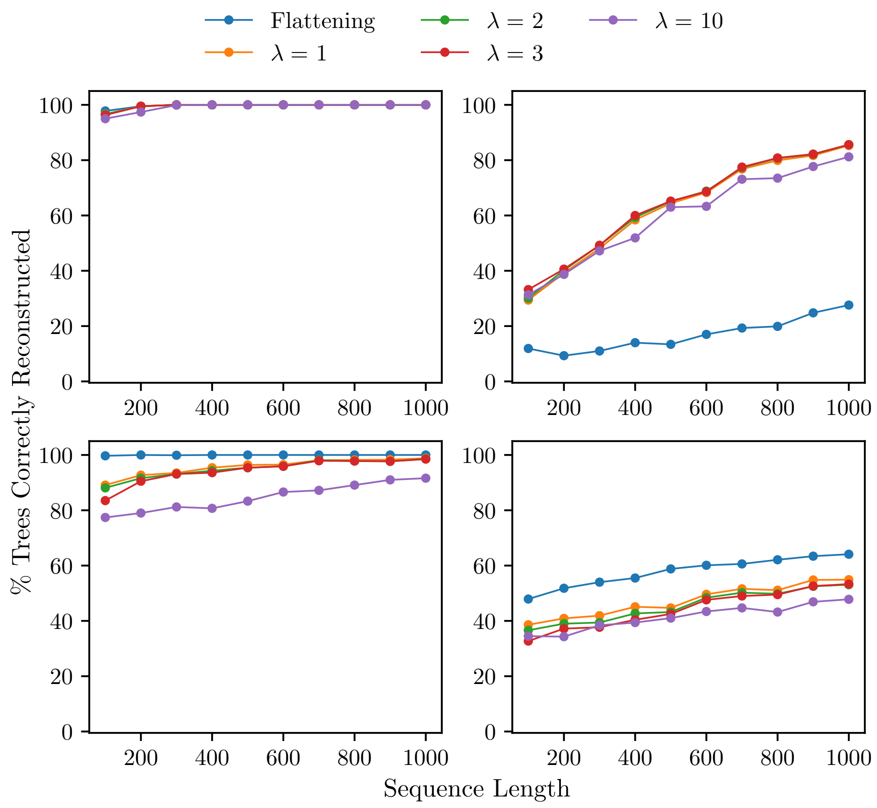


Figure 5.9: Reconstructing quartet trees shown in Figure 5.8 with subflattenings created using differently scaled S matrices. The figures correspond to trees T_{4a}, T_{4b}, T_{4c} and T_{4d} respectively (reading left to right). Eriksson's SVD algorithm is used.

in Figure 5.10. We saw that all the generalised subflattenings correctly gave low scores for true splits, and higher scores for false splits. This is further evidence for the validity of Conjecture 4.3.8. When empirical site-pattern probabilities were used (sequence length 1000) the scores for true splits were ordered as one might expect after looking at Figure 5.4, that is, the larger (r, c)-subflattenings gave higher scores, but lower than the flattening. Testing generalised subflattenings on other trees gave similar results.

Next, we repeated the simulation performed on quartet trees in Section 5.2, but with generalised subflattenings instead of subflattenings with different S matrices. Interestingly, we saw that the $(2, 1)$-subflattenings were negatively impacted by long branch attraction to a degree similar to the flattenings, whereas the $(1, 2)$-subflattenings were less impacted, correctly reconstructing the correct tree almost as often as the standard subflattening. The results of this simulation are given in Figure 5.11.

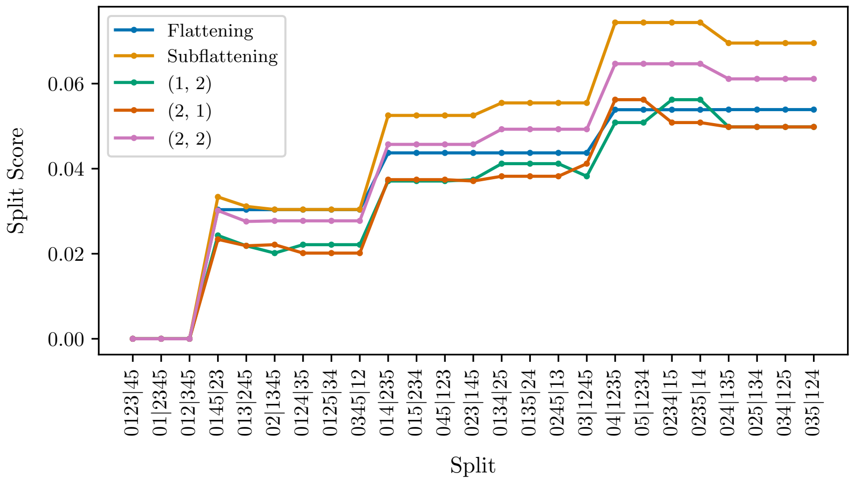
The next step was to compare split scores for different subflattenings and generalised subflattenings on some real data sets.

5.4 Analysis: Real Data Sets

The analysis of flattenings and subflattenings concluded with the evaluation of split scores from two data sets. On each of these data sets, we looked at the scores for flattenings, $(1, 2)$, $(2, 1)$ and $(2, 2)$ subflattenings, as well as subflattenings constructed using matrices $S = H^{(\lambda)}$, with $\lambda \in \{0.5, 3, 15\}$.

The first data set consisted of an alignment of 5 primate DNA sequences, with a sequence length of 1 100 949. All the options we tested seemed to perform similarly, identifying the same four splits as being more likely than others. Since the four lowest scoring splits are incompatible, all we can say from this is that taxa 0 and 1 are clearly separated from the other taxa by an internal edge. In particular, $(1, 2)$ and $(2, 1)$ subflattenings performed similarly to each other, as did the $(2, 2)$ subflattening and the flattening, which is expected since at only 5 taxa the $(2, 2)$ subflattening is almost the same size as the flattening. Both of these pairs were distinctly different from the $(1, 1)$ subflattening, with scores roughly in the middle of the other options. For scaled subflattenings, scaling by $\lambda = 3$ didn't change scores much from the subflattening, as we saw in previous simulations. Scaling by $\lambda = 0.5$ or by $\lambda = 15$ seemed to yield similar results.

The next data set we looked at was a 4 taxa mosquito DNA sequence alignment, with a sequence length of 24 141 782. Interestingly, this time the $(1, 2)$ and $(1, 1)$



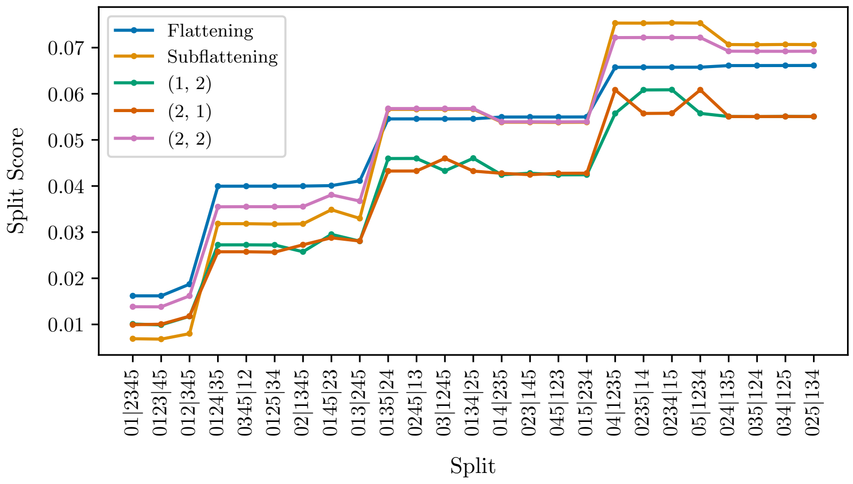


Figure 5.10: **Top:** split scores for all splits on tree T_6 using different generalised subflattenings and exact site-pattern probabilities. **Bottom:** the same as above, but split scores are calculated from empirical site-pattern probabilities drawn from a multinomial distribution (sequence length 1000).

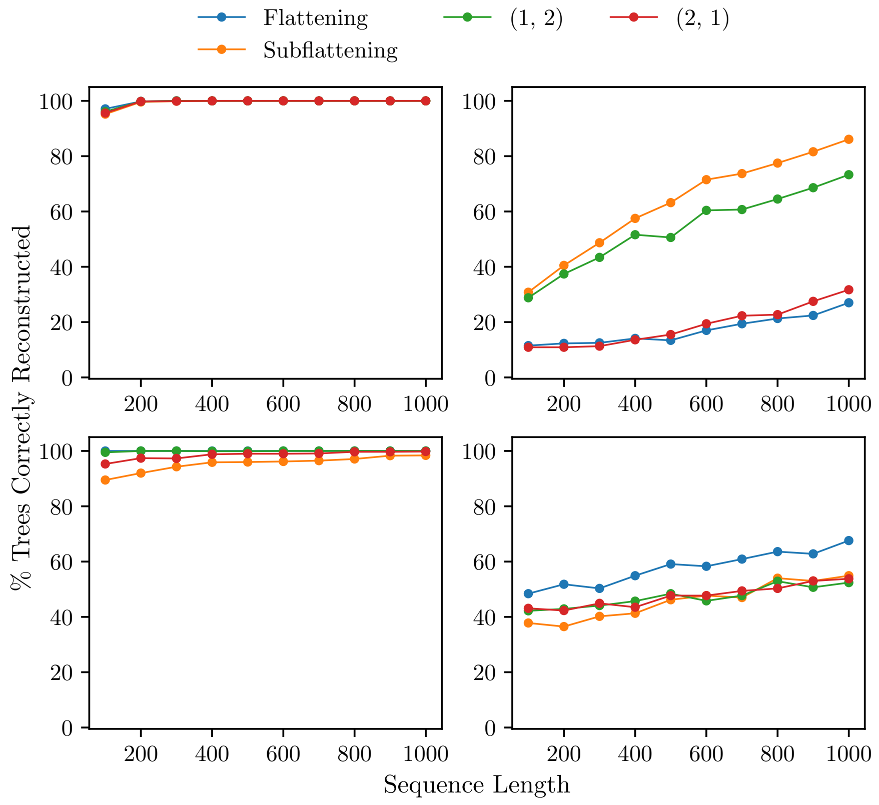


Figure 5.11: Reconstructing quartet trees shown in Figure 5.8 with generalised subflattenings. The figures correspond to trees T_{4a}, T_{4b}, T_{4c} and T_{4d} respectively (reading left to right). Eriksson's SVD algorithm is used.

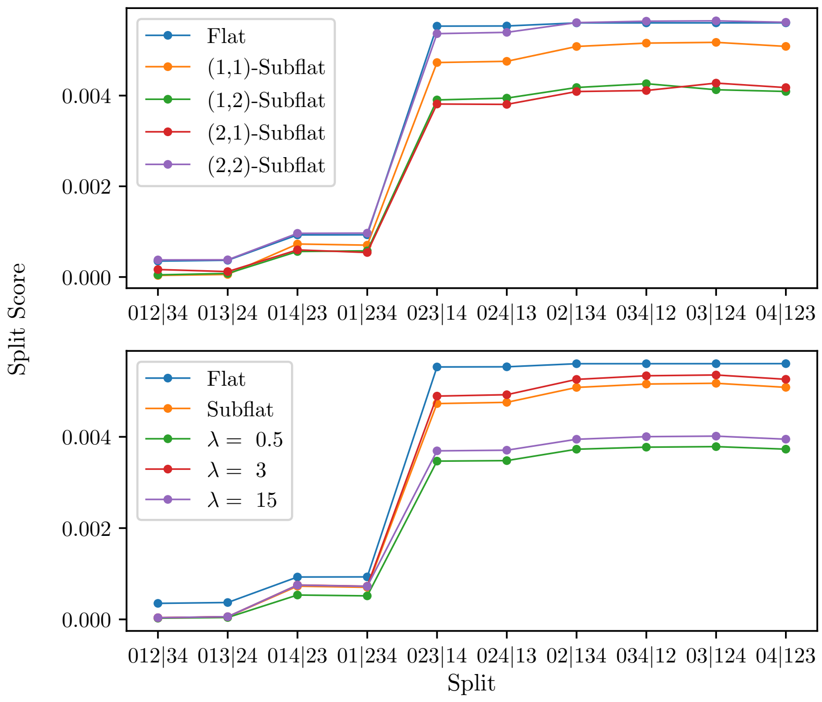


Figure 5.12: Split scores for each split on the primate DNA alignment data set, using subflattenings with different S matrices, and generalised subflattenings

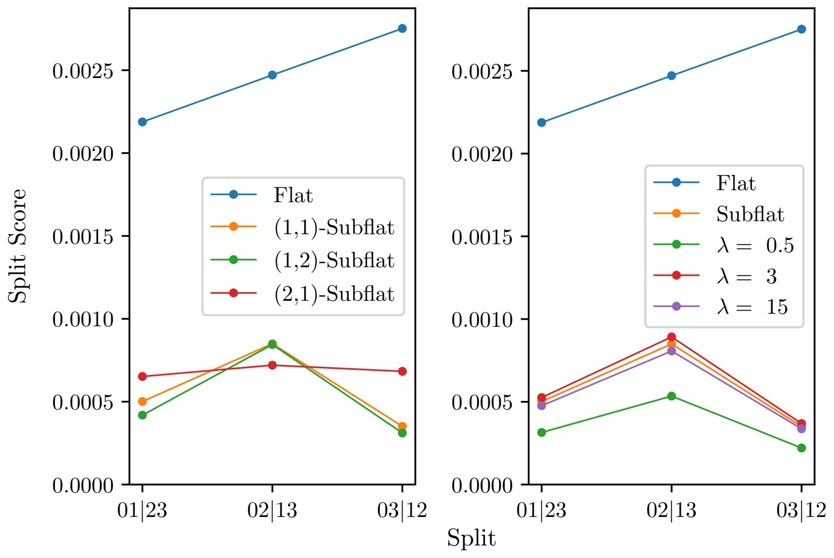


Figure 5.13: Split scores for each split on the mosquito DNA alignment data set, using subflattenings with different S matrices, and generalised subflattenings

subflattenings showed similar scores, and the $(2, 1)$ subflattening scores were quite different. The $(2, 2)$ subflattening is not shown here, since for 4 taxa, it is simply the transformed flattening for every split. For choices of λ, we saw that $\lambda = 0.5$ showed scores which were lower than the other scaling factors (which were fairly similar to each other). The scores for all the subflattenings were very different from the flattening scores.

Chapter 6

Conclusion

6.1 Discussion

Flattenings and subflattenings, and more generally split and rank based tools, encompass some interesting algebraic and statistical ideas, and motivate methods for phylogenetic inference. This work has provided an introduction to some of these tools and methods, as well as new insight into related ideas through both the development of some new definitions and results, and an analysis of these ideas on simulated and real data sets. Here, we summarise the theoretical results from the research undertaken, as well as the insight gained from simulations and analyses conducted in the previous chapter.

In Chapter 3, we provided a new proof of the flattening rank theorem which is more closely related to the proof of the subflattening rank theorem. It is hoped that this new proof gives more insight into subflattening matrices themselves, and how the entries of these matrices change under divergence events. The proof also relies only on tools commonly used in phylogenetics to prove results concerning trees and the general Markov model.

Chapter 4 provided a proof of an alternate definition of the subflattening. This alternate construction is included in the software provided, alongside the original implementation of the subflattening. This proof was followed by an effort to generalise subflattenings, which began with the identification of additional faithful subrepresentations of $\times^m \operatorname{Aff}(k-1)$. The proof of the existence of these representations was structured in a way that clearly motivates the definition of the generalised subflattenings. Finally, we conjecture that the rank conditions of these (r, c)-subflattenings generalises Theorem 2.6.3.

The simulations and analysis in Chapter 5 answer some questions about the performance of the various subflattening constructions, and also raises some new ones (which we will discuss in Section 6.2). First, we have established that particular choices of the matrix S lead to far superior performance compared to other valid choices, while all of the valid choices we tested seem to be usable for inferring phylogenies. On the trees we tested the appropriately sized Hadamard matrix appeared to be the best choice for S, and while there was some evidence that scaling this matrix by certain amounts could lead to improvements, this was not verified by our subsequent analysis. We saw that—on the particular trees we looked at—tree reconstruction via Eriksson's SVD algorithm proposed in [11] was much more successful using subflattenings compared to flattenings, and we found some evidence that suggests subflattenings are perhaps less impacted by the effect of long branch attraction on quartets. Finally, we verified that on the trees and data sets we used for our simulations and analysis, generalised subflattenings indeed have similar properties to flattenings and subflattenings, and are therefore useful for phylogenetic inference in a similar way.

Having discussed the new theoretical results which we have provided, as well as the conclusions we have been able to draw from our practical analysis, we now discuss the several avenues for future research in this area.

6.2 Further Questions and Future Research

This research motivates a number of questions relating to split and rank based methods for phylogenetic inference, and helps bring to light a number of opportunities for future research. Representative of the diversity of ideas in phylogenetics, these questions require research from statistical, algebraic and biological points of view.

First, more work can be done to systematically compare rank based tools—such as flattenings and subflattenings—to other methods for phylogenetic inference. This could include the implementation of some of the other SVD-based methods which have been developed. Such research should also include further investigation into the connection between these rank based methods and those concerning maximum parsimony, and whether the use of these methods introduces the same biases inherent in the use of parsimony scores. It would also be beneficial to investigate the viability of subflattenings as a secondary method for inference used alongside other methods, for example maximum likelihood methods. It is worth considering whether or not flattenings, subflattenings and (r, c)-subflattenings could be combined in order to reveal additional structure within the true phylogenetic tree.

Another opportunity for future work is to continue the investigation into the possible choices of matrix S in the construction of subflattenings. We have shown that some choices are considerably more effective than others, and that Hadamard matrices are perhaps the most viable choice. A theoretical understanding of why Hadamard matrices seem to work well is yet to be developed.

There may be alternative methods for constructing generalised subflattenings which are analogous to the construction of the subflattening which we proved in Section 4.2. Attempting to identify and prove any such constructions is another possible avenue for future work.

From a practical perspective, more work needs to be done to determine a fast, effective, unbiased and statistically consistent algorithm for utilising the properties of flattenings and subflattenings for phylogenetic inference. From the literature, it seems that SVD-based algorithms might be the best option, however more work could be done to evaluate other ways of deciding how close a given matrix is to having to a particular rank, for use with flattenings and subflattenings. Another part of this work should involve identifying the most CPU and memory efficient process for constructing flattenings and subflattenings. There are several opportunities to improve upon the code developed for analysis conducted in this thesis[1], including the implementation of sparse matrix representations, and the replacement of the current tree implementation with one that utilises an existing open-source package such as *NetworkX* [23].

[1]`https://github.com/js51/SplitP/`

Appendix A

Markov Models

Neyman 2-state Model (2-State-Symmetric) (N_2):

$$M_{2SS} = \left\{ \begin{bmatrix} 1-q & q \\ q & 1-q \end{bmatrix} : q \in [0,1] \right\}$$

Jukes-Cantor (4-State-Symmetric) (JC69):

$$M_{JC69} = \left\{ \begin{bmatrix} 1-3q & q & q & q \\ q & 1-3q & q & q \\ q & q & 1-3q & q \\ q & q & q & 1-3q \end{bmatrix} : q \in [0,1] \right\}$$

Kimura's Two Substitution Type Model (K2ST):

$$M_{K2ST} = \left\{ \begin{bmatrix} 1-a-2b & a & b & b \\ a & 1-a-2b & b & b \\ b & b & 1-a-2b & a \\ b & b & a & 1-a-2b \end{bmatrix} : a, b \in [0,1] \right\}$$

Kimura's Three Substitution Type Model (K3ST):

$$M_{K3ST} = \left\{ \begin{bmatrix} 1-a-b-c & a & b & c \\ a & 1-a-b-c & c & b \\ b & c & 1-a-b-c & a \\ c & b & a & 1-a-b-c \end{bmatrix} : a, b, c \in [0,1] \right\}$$

General Markov Model (GMM):

$$M_{GMM} = \left\{ \begin{bmatrix} 1-\sum_j a_{0,j} & a_{0,1} & a_{0,2} & a_{0,3} \\ a_{1,1} & 1-\sum_j a_{1,j} & a_{1,2} & a_{1,3} \\ a_{2,1} & a_{2,2} & 1-\sum_j a_{2,j} & a_{2,3} \\ a_{3,1} & a_{3,2} & a_{3,3} & 1-\sum_j a_{3,j} \end{bmatrix} : a_{i,j} \in [0,1] \right\}$$

Note that $M_{JC69}, M_{K2ST}, M_{K3ST} \subset M_{GMM}$.

Appendix B

Subflattening Example

Consider the following flattening matrix for split 12|34 on a 4-taxon tree, with binary state-space.

$$\begin{bmatrix} p_{0000} & p_{0001} & p_{0010} & p_{0011} \\ p_{0100} & p_{0101} & p_{0110} & p_{0111} \\ p_{1000} & p_{1001} & p_{1010} & p_{1011} \\ p_{1100} & p_{1101} & p_{1110} & p_{1111} \end{bmatrix}$$

The corresponding subflattening obtained by choosing matrix $S = \left[\begin{smallmatrix} \lambda & -\lambda \\ 1 & 1 \end{smallmatrix}\right]$:

$$\left[\begin{array}{c} \lambda^2\,(p_{0000}+p_{0001}-p_{0010}-p_{0011}+p_{0100}+p_{0101}-p_{0110}-p_{0111}-p_{1000}-p_{1001}+p_{1010}+p_{1011}-p_{1100}-p_{1101}+p_{1110}+p_{1111}) \\ \lambda^2\,(p_{0000}+p_{0001}-p_{0010}-p_{0011}-p_{0100}-p_{0101}+p_{0110}+p_{0111}+p_{1000}+p_{1001}-p_{1010}-p_{1011}-p_{1100}-p_{1101}+p_{1110}+p_{1111}) \\ \lambda\,(p_{0000}+p_{0001}-p_{0010}-p_{0011}+p_{0100}+p_{0101}-p_{0110}-p_{0111}+p_{1000}+p_{1001}-p_{1010}-p_{1011}+p_{1100}+p_{1101}-p_{1110}-p_{1111}) \end{array}\right.$$

$$\begin{array}{c} \lambda^2\,(p_{0000}-p_{0001}+p_{0010}-p_{0011}+p_{0100}-p_{0101}+p_{0110}-p_{0111}-p_{1000}+p_{1001}-p_{1010}+p_{1011}-p_{1100}+p_{1101}-p_{1110}+p_{1111}) \\ \lambda^2\,(p_{0000}-p_{0001}+p_{0010}-p_{0011}-p_{0100}+p_{0101}-p_{0110}+p_{0111}+p_{1000}-p_{1001}+p_{1010}-p_{1011}-p_{1100}+p_{1101}-p_{1110}+p_{1111}) \\ \lambda\,(p_{0000}-p_{0001}+p_{0010}-p_{0011}+p_{0100}-p_{0101}+p_{0110}-p_{0111}+p_{1000}-p_{1001}+p_{1010}-p_{1011}+p_{1100}-p_{1101}+p_{1110}-p_{1111}) \end{array}$$

$$\left.\begin{array}{c} \lambda\,(p_{0000}+p_{0001}+p_{0010}+p_{0011}+p_{0100}+p_{0101}+p_{0110}+p_{0111}-p_{1000}-p_{1001}-p_{1010}-p_{1011}-p_{1100}-p_{1101}-p_{1110}-p_{1111}) \\ \lambda\,(p_{0000}+p_{0001}+p_{0010}+p_{0011}-p_{0100}-p_{0101}-p_{0110}-p_{0111}+p_{1000}+p_{1001}+p_{1010}+p_{1011}-p_{1100}-p_{1101}-p_{1110}-p_{1111}) \\ p_{0000}+p_{0001}+p_{0010}+p_{0011}+p_{0100}+p_{0101}+p_{0110}+p_{0111}+p_{1000}+p_{1001}+p_{1010}+p_{1011}+p_{1100}+p_{1101}+p_{1110}+p_{1111} \end{array}\right]$$

BIBLIOGRAPHY

[1] Elizabeth S. Allman and John A. Rhodes. Quartets and parameter recovery for the general Markov model of sequence mutation. *Applied Mathematics Research eXpress*, 2004(4):107–131, 07 2004. ISSN 1687-1200. doi: 10.1155/S1687120004020283. URL https://doi.org/10.1155/S1687120004020283.

[2] Elizabeth S. Allman and John A. Rhodes. Phylogenetic invariants. In Olivier Gascuel and Mike Steel, editors, *Reconstructing Evolution: New Mathematical and Computational Advances*, chapter 19, pages 108–147. Oxford University Press, 2007.

[3] Elizabeth S. Allman and John A. Rhodes. Phylogenetic ideals and varieties for the general Markov model. *Advances in Applied Mathematics*, 40(2):127–148, 2008. ISSN 0196-8858. doi: https://doi.org/10.1016/j.aam.2006.10.002. URL http://www.sciencedirect.com/science/article/pii/S0196885806002077.

[4] Elizabeth S. Allman, Laura S. Kubatko, and John A. Rhodes. Split Scores: A Tool to Quantify Phylogenetic Signal in Genome-Scale Data. *Systematic Biology*, 66(4):620–636, January 2017. ISSN 1063-5157. doi: 10.1093/sysbio/syw103. URL https://doi.org/10.1093/sysbio/syw103.

[5] David Bryant. Hadamard Phylogenetic Methods and the n-taxon Process. *Bulletin of Mathematical Biology*, 71(2):339–359, Oct 2008. ISSN 1522-9602. doi: 10.1007/s11538-008-9364-8. URL https://doi.org/10.1007/s11538-008-9364-8.

[6] Joseph H. Camin and Robert R. Sokal. A Method for Deducing Branching Sequences in Phylogeny. *Evolution*, 19(3):311–326, 1965. ISSN 00143820, 15585646. URL http://www.jstor.org/stable/2406441.

[7] James A. Cavender and Joseph Felsenstein. Invariants of phylogenies in a simple case with discrete states. *Journal of Classification*, 4(1):57–71, Mar 1987. ISSN 1432-1343. doi: 10.1007/BF01890075. URL https://doi.org/10.1007/BF01890075.

[8] Julia Chifman and Laura Kubatko. Quartet Inference from SNP Data Under the Coalescent Model. *Bioinformatics*, 30(23):3317–3324, 08 2014. ISSN 1367-

4803. doi: 10.1093/bioinformatics/btu530. URL `https://doi.org/10.1093/bioinformatics/btu530`.

[9] Carl Eckart and Gale Young. The approximation of one matrix by another of lower rank. *Psychometrika*, 1(3):211–218, Sep 1936. ISSN 1860-0980. doi: 10.1007/BF02288367. URL `https://doi.org/10.1007/BF02288367`.

[10] Anthony W.F. Edwards and Luigi L. Cavalli-Sforza. The reconstruction of evolution. *Annals of Human Genetics*, 27:105–106, 1963.

[11] Nicholas Eriksson. Tree Construction Using Singular Value Decomposition. In Lior Pachter and Bernd Sturmfels, editors, *Algebraic Statistics for Computational Biology*, chapter 19, pages 360–368. Berkeley, California, 2005.

[12] Joseph Felsenstein. Maximum Likelihood and Minimum-Steps Methods for Estimating Evolutionary Trees from Data on Discrete Characters. *Systematic Zoology*, 22 (3):240–249, 1973. ISSN 00397989. URL `http://www.jstor.org/stable/2412304`.

[13] Joseph Felsenstein. Cases in which Parsimony or Compatibility Methods Will be Positively Misleading. *Systematic Zoology*, 27(4):401–410, 1978. ISSN 00397989. URL `http://www.jstor.org/stable/2412923`.

[14] Joseph Felsenstein. Evolutionary trees from DNA sequences: A maximum likelihood approach. *Journal of Molecular Evolution*, 17(6):368–376, Nov 1981. ISSN 1432-1432. doi: 10.1007/BF01734359. URL `https://doi.org/10.1007/BF01734359`.

[15] Joseph Felsenstein. *Inferring Phylogenies*. Sinauer Associates, Inc., 2004. ISBN 978-0-878931-77-4.

[16] Jesús Fernánndez-Sánchez and Marta Casanellas. Invariant Versus Classical Quartet Inference When Evolution is Heterogeneous Across Sites and Lineages. *Systematic Biology*, 65(2):280–291, 11 2015. ISSN 1063-5157. doi: 10.1093/sysbio/syv086. URL `https://doi.org/10.1093/sysbio/syv086`.

[17] Tan Guolü. The Tensor Production of Block Matrix and its Parallel Computing. *Journal of Algorithms & Computational Technology*, 3(4):503–515, 2009. doi: 10.1260/174830109789621329. URL `https://doi.org/10.1260/174830109789621329`.

[18] Michael D. Hendy and Michael A. Charleston. Hadamard conjugation: A versatile tool for modelling nucleotide sequence evolution. *New Zealand Journal of Botany*, 31(3):231–237, 1993. doi: 10.1080/0028825X.1993.10419500. URL `https://doi.org/10.1080/0028825X.1993.10419500`.

[19] Michael D. Hendy and David Penny. A Framework for the Quantitative Study of Evolutionary Trees. *Systematic Biology*, 38(4):297–309, 12 1989. ISSN 1063-5157. doi: 10.2307/2992396. URL `https://doi.org/10.2307/2992396`.

[20] Joseph E. Johnson. Markov-type lie groups in gl(n,r). *Journal of Mathematical Physics*, 26(2):252–257, 1985. doi: 10.1063/1.526654. URL https://doi.org/10.1063/1.526654.

[21] Young Rock Kim, Oh-In Kwon, Seong-Hun Paeng, and Chun-Jae Park. Phylogenetic tree constructing algorithms fit for grid computing with SVD. *arXiv e-prints*, art. q-bio/0611015, Nov 2006. preprint; available at: https://arxiv.org/abs/q-bio/0611015.

[22] J A Lake. A rate-independent technique for analysis of nucleic acid sequences: evolutionary parsimony. *Molecular Biology and Evolution*, 4(2):167–191, 03 1987. ISSN 0737-4038. doi: 10.1093/oxfordjournals.molbev.a040433. URL https://doi.org/10.1093/oxfordjournals.molbev.a040433.

[23] NetworkX developers. NetworkX. https://networkx.github.io/.

[24] David G. Poole. The stochastic group. *The American Mathematical Monthly*, 102(9):798–801, 1995. ISSN 00029890, 19300972. URL http://www.jstor.org/stable/2974507.

[25] Mike Steel. *Phylogeny—Discrete and Random Processes in Evolution.* David Marshal, 2016. ISBN 978-1-611974-47-8.

[26] Jeremy G. Sumner. Dimensional Reduction for the General Markov Model on Phylogenetic Trees. *Bulletin of Mathematical Biology*, 79(3):619–634, March 2017. ISSN 1522-9602. doi: 10.1007/s11538-017-0249-6. URL https://doi.org/10.1007/s11538-017-0249-6.

[27] Jeremy G. Sumner, Michael A. Charleston, Lars S. Jermiin, and Peter D. Jarvis. Markov invariants, plethysms, and phylogenetics. *Journal of Theoretical Biology*, 253(3):601–615, 2008. ISSN 0022-5193. doi: https://doi.org/10.1016/j.jtbi.2008.04.001. URL http://www.sciencedirect.com/science/article/pii/S0022519308001720.

[28] Derrick S. Tracy and Kankanam G. Jinadasa. Partitioned Kronecker Products of Matrices and Applications. *The Canadian Journal of Statistics / La Revue Canadienne de Statistique*, 17(1):107–120, 1989. ISSN 03195724. URL http://www.jstor.org/stable/3314768.

[29] Ziheng Yang. Statistical properties of the maximum likelihood method of phylogenetic estimation and comparison with distance matrix methods. *Systematic Biology*, 43(3):329–342, 1994. ISSN 10635157, 1076836X. URL http://www.jstor.org/stable/2413672.

INDEX