Utrecht University

Bayesian Inference of Phylogeny Using Variable Number of Tandem Repeats and Markov Chain Monte Carlo

Author
Arjun Dhawan

Supervisors
Dr. Martin Bootsma[*]
Dr. Don Klinkenberg[†]
Dr. Hester Korthals Altes[†]

[*]Mathematical Institute, Utrecht University
[†]RIVM National Institute for Public Health and the Environment

A thesis submitted in partial fulfilment of the requirements for the degree of

Master of Science
in
Mathematical Sciences

7th November 2016

Abstract

We implement and evaluate methods to infer the phylogeny of Variable Number of Tandem Repeats (VNTR) isolates of tuberculosis through Bayesian inference and Markov Chain Monte Carlo, using an existing transition rate matrix [Sai+04]. By also inferring the phylogeny through the model of Hasegawa, Kishino and Yano (HKY) using nucleotide data of the same isolates, we are able quantitatively and qualitatively compare the phylogenies obtained through both models. By simulating data, we assess how well the true phylogeny can be inferred for both the Sainudiin and HKY model, for different levels of mutational saturation in the data. We show how both the Sainudiin and HKY model can be combined to yield a phylogeny that is better resolved and more accurate than by the use of either model. By changing the model for the mutation rate proportionality in the Sainudiin model, we are able to use the estimates of the model parameters to speculate on the mechanisms by which VNTR mutates. The developed methods have been made available in the package BEASTvntr for BEAST2.

Contents

— 1 —
Introduction

1.1 Tuberculosis

Tuberculosis (TB) is an infectious disease caused by *Mycobacterium tuberculosis*. Even though preventable and curable, each year approximately 10 million people fall ill with tuberculosis and approximately 2 million die from having the disease (WHO). In order to improve the control the disease, the epidemiology has to be better understood.

One way to achieve this goal is to analyse the genetic data of TB. For example, understanding the genetic diversity helps in the development of novel antibiotics for TB, as some of the lineages of TB can be associated with drug resistance [Kös+12]. Another example is the identification of new clonal lineages and understanding which mutations suddenly make them successful, an important step in the control and eradication of TB [Smi+09].

1.2 Phylogenetics

In order to make progress in these areas, one has to study the hereditary relationships between lineages. The field of research that studies these genetic relationships using molecular data (i.e. data about DNA or protein sequences), is called phylogenetics. Reconstruction of the evolutionary history is especially relevant for infectious diseases since it also gives insight into the spread of it.

The goal of these reconstruction methods is to find the phylogenetic tree that shows the evolutionary relationship between species. Following the assumption that evolution is made up of bifurcating events, this tree is assumed to be a bifurcating tree. However, if the reconstruction method fails to distinguish between these events, it can also yield a multifurcating tree. [HRW92]. The root of a phylogenetic tree is interpreted as the most recent common ancestor (MRCA) of the leaves of the tree.

If the branch lengths of the tree have no interpretation, the phylogentic tree only represents topology. Otherwise, the branch lengths can be interpreted as evolutionary distance, which can also be scaled to represent time.

The data that is taken as input for these methods are isolates (data sequenced from the DNA of members) from a population. Many different ways of sequencing the DNA exist: what is important is that the resulting data captures the mutations that differentiate the members of the population.

1.2.1 Genome Sequencing

Such data can be obtained by Whole Genome Sequencing (WGS), whereby the whole genome is sequenced, or by Multilocus Sequence Typing (MLST), whereby a set of genes is sequenced. One type of mutations of the genome captured in these data, are substitutions, otherwise known as point mutations. A substitution occurs when any of the four bases A, C, T and G transforms into another. For molecular reasons, transformations A $\leftrightarrow$ G and C $\leftrightarrow$ T, called *transitions*, are more likely to happen than transformations {A,G} $\leftrightarrow$ {C,T}, called *transversions*.

Almost all of the genome remains unchanged by these mutations. Therefore, other ways to assess changes in the DNA have been developed: so-called genetic markers focus on the parts of the DNA known to exhibit much genetic variation between samples.

1.2.2 Variable Number of Tandem Repeat

One such marker is called Variable Number of Tandem Repeat (VNTR). It captures the mutations which occur by the process of slippage, which is treated more extensively in chapter 2. Slippage duplicates or removes entire sections of strands.

VNTR focusses on locations on the genome where particular stretches of DNA are repeated stretches that are between 10 and 100 base pairs long. Since the stretches which are repeated are likely identical, only the numbers of repeats for each location (or locus) are measured when typing VNTR. The standard typing method for TB is Mycobacterial Interspersed Repetitive Unit VNTR (MIRU-VNTR), which comprises the typing of VNTR on 24 pre-defined loci on the DNA.

WGS used to be a time-consuming, expensive and tedious task. However, the recent development of cheaper next-generation sequencing (NGS) means that WGS is a viable candidate to replace VNTR in the future for providing the identifying information on TB samples. However, some of these NGS techniques have a limited read length (between 35 and 700 base pairs) [GMM16], which means that a very high number of repeats or very long repeats, might not be observed with WGS, but could possibly still be found using VNTR.

1.2.3 Methods for inferring phylogenies

Two of the most common phylogenetic inference methods are Minimum Spanning Tree (MST) and Neighbour Joining (NJ). NJ is a greedy algorithm which seeks to produce the minimum evolution tree, i.e. at each iteration of the algorithm it connect nodes such that evolutionary distance is as small as possible. MST is the tree that connects all vertices (i.e. samples) such that the sum of the length of the edges is minimal. Both methods are based on evolutionary distance data, i.e. for any two samples i, j some (genetic) distance $d(i, j)$ between them must be defined.

These two methods are heuristics. They give quick (and sometimes) satisfactory solutions, but are certainly not guaranteed to be the most truthful trees that we could infer based on the data. By solely looking at distance for any two samples we can not possibly capture all information that is present in the data. Furthermore, the distance between samples is ambiguously defined. For example, for VNTR it is common to define the distance as the proportion of loci that are unequal (Jaccard distance) or as the sum of the difference in repeats for each locus (Manhattan distance). It has also been shown that NJ produces trees of insufficient quality, meaning only major lineages can be identified, for VNTR, compared to trees based on the nucleotide data [Com+09]. This shows that the current methods for inferring the phylogeny for VNTR are inadequate.

Bayesian inference

By using an evolutionary model, in contrast to the above methods, we define the probabilities by which the VNTR pattern mutates from one state to another. At the heart of such a model lies the definition of a matrix $\mathbf{Q}$ which specifies the instantaneous rate of mutations of one state into another, i.e., the number of mutations that take place per time unit. This enables us to compute the likelihood of finding some data $\mathbf{D}$ in the tips of the tree, given that the data $\mathbf{D}$ are the result of mutations along some single bifurcating tree τ. This can be done using the Felsenstein algorithm [Fel73]. A tree for which the likelihood function is maximized is called the maximum-likelihood tree, which can also be used as the phylogeny. But this is not the proper criterion for finding the most truthful tree. To find the most truthful tree, we are actually interested in the posterior probability of a tree, instead of the likelihood of finding data.

This posterior probability can be computed using Bayes' theorem. In the process of transforming the likelihood of data given a tree into the posterior probability of a tree given the data, we must normalize over all possible trees. Since the number of all possible trees grows very fast with the size of the tree, this is infeasible. This problem can be circumvented by sampling trees and parameters from this posterior distribution via Markov Chain Monte Carlo (MCMC), which is explained in section 2.5.

1.3 Aim

To expand upon the current insufficient heuristic methods for the inference of phylogeny using VNTR for TB, our goal in this thesis is to explore a new method for the inference of phylogeny by making use of the MCMC method, employing an existing evolutionary model for VNTR. We will implement this method in BEAST2, which is software for phylogenetic analysis.

In addition, using this inference method, we will provide an answer to the following related issues:

- What is the most parsimonious, i.e. simplest version of this method, that can still accurately infer the phylogeny?
- How does this method compare to existing, well-established methods that can infer the phylogeny for nucleotides, in terms of being able to accurately recover the true phylogeny?
- Subsequently, can information from both VNTR and nucleotide data be combined, and does this improve the accuracy of the obtained phylogeny?
- Can we use the model of the inference method to shed light on the mechanisms by which VNTR mutates?

— 2 —
The evolution of VNTR and the use of MCMC

In this chapter we will specify the evolutionary model which we will use in the MCMC method to infer the phylogeny. As said in the previous chapter, these MCMC methods will sample the posterior probability distribution of trees and parameters. To calculate this posterior probability, we must first compute $f(\mathbf{D}|\tau, \boldsymbol{\theta})$, which is the likelihood of data $\mathbf{D}$ given tree τ and model parameters $\boldsymbol{\theta}$. This is done using the Felsenstein algorithm.

The first step in calculating the likelihood $f(\mathbf{D}|\tau, \boldsymbol{\theta})$ is the specification of the transition rate matrix $\mathbf{Q}$ of the VNTR states, for which we need a model for the evolution of these states. By solving a linear differential equation involving $\mathbf{Q}$, via matrix exponentiation, we can find $\mathbf{P}(t)$ which gives the probability of a transition between any two states in time interval t. Matrix $\mathbf{P}(t)$ in conjunction with Felsenstein's algorithm allows us to calculate likelihood $f(\mathbf{D}|\tau, \boldsymbol{\theta})$.

2.1 A model for the evolution of VNTR

Microsatellites are tandem repeats where the repeat is between 2 and 5 base pairs in length. Minisatellites (such as VNTR) are tandem repeats of which the repeated sequence is 10 to 100 base pairs in length. For the former, an extensive evolutionary model given by Sainudiin exists in the literature [Sai+04]. This model combines all the previously known models for Microsatellites into one 'supermodel'. We have no reason to assume that the mechanisms of mutations would be any different for minisatellites compared to microsatellites, therefore we shall use this model to infer the phylogeny for VNTR.

In this model it is assumed that the alleles can either gain or lose repeats by the process of *replication slippage*, described in [Ell04]. During replication, the code of the DNA, consisting of base-pairs A,C and T,G, gets replicated in another strand which consists of the opposite bases. Any base on one strand only 'fits' with its paired base on the other strand, which makes it a fail-safe feature of the DNA to correct any misalignments. However, when parts of the DNA are repeated, this feature may fail to recognise the misalignments of any repeated stretch that is aligned to another repeated stretch, thus forming a loop in the DNA (also see figure 3.10).

The simplest model which describes the misalignment is the stepwise mutation model (SMM). In the SMM, a repeated stretch on the VNTR locus can either gain or lose 1 repeat per mutation event. This model can be extended to allow for mutations of step size ≥ 1, where the distribution of step sizes is taken to be geometric. This distribution arises by assuming that in a mutation event, each extra change in the number of repeats, happens with equal probability.

If gaining or losing a repeat would be equally likely, this would lead to unconstrained growth of the number of repeats, which is not observed in nature. Therefore an extension to this model exists, where upon mutation a bias in the probability for expansion β and contraction $1-\beta$ is introduced, which controls the growth of the repeats. This bias may depend on the number of repeats. One can think of several mechanisms which cause this bias. For example, higher repeats are more likely to shorten due to the point mutations. There may also be a lower fitness of bacteria carrying unnecessary DNA.

Finally, it is possible that alleles with more repeats mutate at higher rate, which can be modelled by letting the underlying mutation rate increase proportionally with the number of repeats. The idea of this is that more repeats offer more opportunities for a slippage event [Kru+98].

The 3 ingredients discussed have an effect on the transition rate of a mutation from state i to j. The dependency of the rate proportionality on the number of repeats is captured in α, the mutational bias in β, and the step-size distribution in γ. All the effects of the 3 ingredients are assumed to be independent of each other. Furthermore, we have a mutation rate μ which scales all these factors. Thus, the 4 factors can be multiplied to give a rate associated with a mutation from state i to j. Using these rates directly we can only calculate the probability of instantaneously mutating, i.e. for an infinitesimal small time interval. To calculate the probability of a mutation from state i to j on a longer time interval, the theory of continuous time Markov Chain is used. By specifying rate matrix $\mathbf{Q} = (q_{i,j})_{i,j=i_{\text{min}},\ldots,i_{\text{max}}}$ we describe the rate of the transitions from state i to state j, limited on the state space $\{i_{\text{min}},\ldots,i_{\text{max}}\}$. It can be used to compute the probabilities of transitioning between states i, j in time interval t, with (constant) mutation rate μ:

$$\mathbf{P}(t) = e^{\frac{\mathbf{Q}}{||\mathbf{Q}||}\mu t} \tag{2.1}$$

Here $\mathbf{P}(t)$ is a solution to $\mathbf{P}'(t) = \mathbf{P}(t)\frac{\mathbf{Q}}{||\mathbf{Q}||}\mu$. Quantity $||\mathbf{Q}||$ means the net mutation rate that can be associated with rate matrix $\mathbf{Q}$, and will be defined in section 2.3.

In our case, $\mathbf{Q}$ is specifically defined as

$$q_{i,j} = \begin{cases} \alpha(a_0,i)\beta(r_b,\theta_b,i)\gamma(g,i,j) & \text{if } j \geq i+1 \\ \alpha(a_0,i)(1-\beta(r_b,\theta_b,i))\gamma(g,i,j) & \text{if } j \leq i-1 \\ -\sum_{k\neq i} q_{i,k} & \text{if } j = i \end{cases} \tag{2.2}$$

The diagonals are defined such that the rate of probability flowing out of a state is equal to the sum of the rates of probabilities flowing into other states. The formulas used in the definition of **Q** are explained below.

Mutation rate proportionality α

We assume that the mutation rate is linearly dependent on the number of repeats, consisting of a 'baseline' rate μ_0, and a rate μ_1 that increases with the repeats:

$$\tilde{\alpha}(\mu_0,\mu_1,i) = \mu_0 + \mu_1(i - i_{\text{min}}) \tag{2.3}$$

This expression can be normalized using μ_0, to arrive at

$$\alpha'(a_1,i) = 1 + a_1(i - i_{\text{min}}) \tag{2.4}$$

which is the model used in [Sai+04]. In our practical implementation of the model, we normalized using μ_1 to yield

$$\alpha(a_0, i) = a_0 + i - i_{\text{min}} \tag{2.5}$$

Here we have parameter $1/a_1 = a_0$, representing the offset of the rate for $i = i_{\text{min}}$, instead of a_1. The reason for this is that if the mutation rate of repeat i_{min} is far lower than the others, a_1 will grow to very large values, while a_0 conveniently remains bounded. Since rate matrix $\mathbf{Q}$ is normalized with net rate $||\mathbf{Q}||$ as explained in section 2.3, both parametrizations are equivalent.

Mutational bias β

A mutation event might have a preference for a certain direction, in the sense that given this event it is an expansion with probability β and a contraction with probability $1 - \beta$. As the number of repeats i grows, the probability of contraction increases, while the probability of expansion decreases. This behaviour of the mutational bias in favour of expansion, can be modelled with a logistic formula, proposed by Wu and Drummond [WD11]

$$\beta(b_0, b_1, i) = \frac{1}{1 + \exp(-(b_0 + b_1(i - i_{\text{min}})))}, \tag{2.6}$$

whilst the bias of contraction is modelled with factor $(1 - \beta(b_0, b_1, i))$. Here, b_0 is the parameter that determines the offset of β at $i = i_{\text{min}}$, while larger negative values for b_1 mean that β reaches a probability of 0 for lower i. An example graph of $\beta, 1 - \beta$ is shown in figure 2.1.

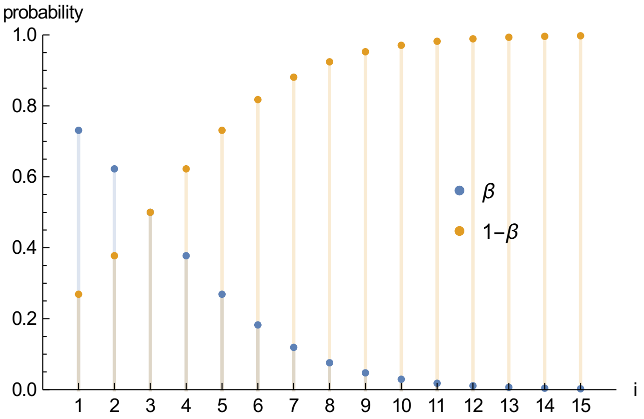


Figure 2.1: An example of a possible mutational bias: graph of $\beta, 1 - \beta$ for $b_0 = 1, b_1 = -\frac{1}{2}, i_{\text{min}} = 1$.

Preliminary results of the MCMC algorithm showed us that b_0, b_1 are linearly correlated. To investigate the reason for this correlation, we looked at the equilibrium states of the mutational bias. Equilibrium states are alleles $i = i_{\text{eq}}$ for which the following condition is satisfied.

$$\beta(b_0, b_1, i) = 1 - \beta(b_0, b_1, i) \tag{2.7}$$

Solving equation (2.7) for i yields the solution

$$i_{\text{eq}} = -b_0/b_1 + i_{\text{min}} \tag{2.8}$$

Assuming there exists such an equilibrium state for a locus, we would expect b_0, b_1 to be linearly correlated.

To increase the number of effective samples from our MCMC output, we used polar coordinates: $\big((b_0, b_1) = (r_b \cos(\theta_b), r_b \sin(\theta_b))\big)$ which were then uncorrelated. Observe that in this parametrization, θ_b uniquely determines the focal point

$$i_{\text{eq}} = -\frac{1}{\tan(\theta_b)} + i_{\text{min}}, \tag{2.9}$$

while r_b determines the slope of functions $\beta, 1-\beta$ and can thus be thought of as the 'force' pointing to this state. It is only natural to rewrite θ_b in terms of i_{eq}, via

$$\theta_b = \arctan\left(\frac{-1}{i_{\text{eq}} - i_{\text{min}}}\right),$$

foremost because it is likely that we would have prior information on i_{eq} and not (directly) on θ_b. This saves us the extra work of calculating the PDF of the prior of θ_b using the prior of i_{eq}. This re-parametrisation yields

$$\beta(b_0(r_b, i_{\text{eq}}), b_1(r_b, i_{\text{eq}}), i) = \frac{1}{1 + e^{-(b_0 + b_1(i - i_{\text{min}}))}} \tag{2.10}$$

$$b_0(r_b, i_{\text{eq}}) = \frac{r_b}{\sqrt{1 + 1/(i_{\text{eq}} - i_{\text{min}})^2}} \tag{2.11}$$

$$b_1(r_b, i_{\text{eq}}) = \frac{-r_b}{\sqrt{(i_{\text{eq}} - i_{\text{min}})^2 + 1}} \tag{2.12}$$

If $r_b = 0$, then the probability of expansion is always equal to that of contraction. This is called the *unbiased model*. Otherwise we have a *biased model* where the probability of expansions is not equal to that of contraction.

Step-size distribution γ

The distribution of step sizes $|i - j|$ is modelled using a geometric distribution $\gamma(g, i, j)$, where g is the parameter such that $1 - g$ reflects the probability that a mutation would be a single step mutation. The distribution is normalized on the left and right side of i, such that the surface area there is equal to 1. This is done to ensure that β indeed means the probability of expansion given a mutation. For $0 < g < 1$, we have multi-step mutations, given by the *two-phase model* (TPM):

$$\gamma(g, i, j) = \begin{cases} \frac{(1-g)g^{|i-j|-1}}{1-g^{i_{\max}-i}} & \text{if } i < j \\ \frac{(1-g)g^{|i-j|-1}}{1-g^{i-i_{\min}}} & \text{if } i > j \end{cases} \tag{2.13}$$

For the case $g = 0$, only 1 repeat can be gained or lost. This modelled with the *stepwise mutation model (SMM)* or the *one-phase model*, where we only allow mutations to and from adjacent states:

$$\gamma(0,i,j) = \begin{cases} 1 & \text{if } |i-j| = 1 \\ 0 & \text{otherwise} \end{cases} \tag{2.14}$$

Note that $\gamma(0,i,j)$ is a limiting case of $\gamma(g,i,j)$, by taking $g \to 0$.

The parameters $a_0, r_b, i_{\text{eq}}, g, \mu$ are contained in $\boldsymbol{\theta}$.

2.1.1 State space bounds

For our purposes we will set $i_{\text{min}} = 1, i_{\text{max}} = 15, n_s = i_{\text{max}} - i_{\text{min}} + 1 = 15$, where n_s is the number of states. Even though setting $i_{\text{min}} \leq 0$ is mathematically possible in the model, it is not biologically realistic since replication slippage cannot explain how a repeat can grow in size from 'nothingness'. An upper bound $i_{\text{max}} = 15$ was chosen since in the data we used in chapter 3, the maximum observed repeat was 14. We choose the upper bound near the maximum observation, since we do not want to allow any repeat from existing in the model, that is far above what is observed in the data.

All the effects of the model, the parameters, and the corresponding allowed ranges, are shown in table 2.1.

Effect	Parameters		Range	Parameter meaning
Mutation rate proportionality	a_0	$\in$	$[0, \infty)$	Offset of proportional rate at $i = i_{\text{min}}$.
Mutational bias	r_b	$\in$	$[0, \infty)$	Magnitude of bias.
	i_{eq}	$\in$	$[i_{\text{min}}, i_{\text{max}}]$	Focal point of bias.
Multi-step mutations	g	$\in$	$[0, 1]$	Success probability, $1 - g =$ probability of mutation being single-step.

Table 2.1: All effects in the Sainudiin model and their parameters.

2.2 Stationary distribution of the Sainudiin model

In the theory of continuous time Markov chains, the stationary distribution is a distribution of states $\boldsymbol{\pi}$ defined such that

$$\boldsymbol{\pi}\mathbf{Q} = \mathbf{0}$$

holds. If $\mathbf{Q}$ is primitive, i.e., there is some $k > 0$ such that $\mathbf{Q}^k$ has no entries equal to 0, this distribution is unique, and can always be found via

$$\lim_{t \to \infty} p_{i,j}(t) = \pi_j$$

We are interested in this distribution, since it is necessary for the definition of a net mutation rate, which is used for normalizing $\mathbf{Q}$ in section 2.3.

2.2.1 Calculating the stationary distribution

In case of the stepwise mutation model, the stationary distribution can be calculated in a very straightforward manner, by considering it as the stationary distribution of a birth-death chain. The states which represent the repeats, can undergo births, i.e., a transition to a higher state or upward mutation, or deaths, i.e., a transition to a lower state or downward mutation. For a state i the upward transitions happen with (birth) rate b_i and the downward mutations with (death) rate d_i. Then, the stationary distribution is the state $\boldsymbol{\pi}$ for which the relation $b_i\pi_i = d_{i+1}\pi_{i+1}$ holds, i.e., the inflow in a state is equal to outflow. For the stepwise model, this means that the stationary distribution must be given by

$$\pi_i \propto \prod_{j=i_{\min}}^{i-1} \frac{b_j}{d_{j+1}} = \prod_{j=i_{\min}}^{i-1} \frac{\beta(b_0, b_1, j)\alpha(a_1, j)}{(1-\beta(b_0, b_1, j+1))\alpha(a_1, j+1)}$$

Computing the stationary distribution for the general case

The above expression can only compute the stationary distribution for the special case of the stepwise model. In order to compute it for all other cases - without any significant additional computational cost - we will use the eigendecomposition of $\mathbf{Q}$. This approach does not come at the cost of extra computation, since in BEAST2 the eigendecomposition is already constructed to compute $\mathbf{P} = e^{\mathbf{Q}\mu t}$.

In order to perform matrix exponentiation of $\mathbf{Q}$, BEAST2 first decomposes $\mathbf{Q}$ in a matrix of eigenvectors $\mathbf{U}$ and a diagonal matrix of eigenvalues $\boldsymbol{\Lambda}$

$$\mathbf{Q} = \mathbf{U}\boldsymbol{\Lambda}\mathbf{U}^{-1}.$$

By observing that $\mathbf{Q}^n = \mathbf{U}\boldsymbol{\Lambda}^n\mathbf{U}^{-1}$, it follows from the definition of matrix exponential that

$$e^{\mathbf{Q}\mu t} = \mathbf{U}e^{\boldsymbol{\Lambda}\mu t}\mathbf{U}^{-1}.$$

Thus $\mathbf{P}$ can very easily be computed by decomposing $\mathbf{Q}$ in eigenvectors and eigenvalues, exponentiating the values on the diagonal matrix, and performing two matrix multiplications.

The stationary distribution by definition is a left eigenvector of $\mathbf{P}$, corresponding to eigenvalue 1. We see that this eigenvector is also an eigenvector of $\mathbf{Q}$, corresponding to eigenvalue 0, by looking at the derivative with respect to t of $\mathbf{P}$

$$\boldsymbol{\pi}\mathbf{P} = \boldsymbol{\pi}$$
$$\boldsymbol{\pi}\mathbf{Q}\mu\mathbf{P} = \mathbf{0}$$

Since $\mathbf{P}$ is always invertible (its inverse being $e^{-\mathbf{Q}\mu t}$), $\mathbf{P}$ cannot have eigenvalue 0. Therefore, it always holds that

$$\boldsymbol{\pi}\mathbf{Q} = \mathbf{0}$$

To express $\boldsymbol{\pi}$ in terms of $\mathbf{U}$, note that $\boldsymbol{\pi}^\intercal$ must be a right eigenvector of $\mathbf{Q}^\intercal$, and that

$$\mathbf{Q}^\intercal = (\mathbf{U}\boldsymbol{\Lambda}\mathbf{U}^{-1})^\intercal = (\mathbf{U}^{-1})^\intercal\boldsymbol{\Lambda}^\intercal\mathbf{U}^\intercal.$$

Therefore, (un-normalized) $\boldsymbol{\pi}^\intercal$ can be found in the column of $(\mathbf{U}^{-1})^\intercal$ corresponding to eigenvalue 0.

2.3 Mutation rate

Note that in equation (2.1) we are inherently dealing with an ambiguity: if changing any of the parameters r_b, i_{eq}, g causes the net mutation rate associated with $\mathbf{Q}$ to increase, then such an increase is also caused by an increase in the mutation rate μ.

In this section, we will define a mutation rate ν^* that can be associated with $\mathbf{Q}\mu$. Normalizing $\mathbf{Q}$ using this rate, via $q_{i,j} \to q_{i,j}/(\nu^*/\mu)$ (we divide by μ, since in the normalization of $\mathbf{Q}$ we are only interested in the rate contribution coming from $\mathbf{Q}$, not μ), allows us to capture the net mutation rate of $\mathbf{Q}\mu$ in parameter μ.

2.3.1 Defining the net mutation rate

When defining a net mutation rate, it is possible to weigh a mutation from state i to j in different ways. In our first definition of the net mutation rate that follows, we will weigh the mutations according to their step size.

When a mutation $i \to j$ takes place, we can weigh the change in repeats σ_{ij} as

$$\sigma_{ij} := |i - j| \tag{2.15}$$

The expected change for a certain repeat i is then given by

$$\langle \sigma \rangle_i (t) = \sum_{j=i_{\text{min}}}^{i_{\text{max}}} \sigma_{ij} p_{ij}(t) \tag{2.16}$$

where we measure the change in repeats at time t relative to $t = 0$. Since we are interested in the rate of this (mutational) change, we take the derivative with respect to time to arrive at

$$\frac{d}{dt} \langle \sigma \rangle_i (t) = \sum_{j=i_{\text{min}}}^{i_{\text{max}}} \sigma_{ij} \frac{d}{dt} p_{ij}(t) = \sum_{j=i_{\text{min}}}^{i_{\text{max}}} \sigma_{ij} (\mathbf{P}(t)\mathbf{Q})_{ij} \mu \tag{2.17}$$

Observe that this quantity is independent of any normalization of the rate matrix $\mathbf{Q}$, since it depends on the product of both $\mathbf{Q}$ and μ. Since we are interested in the instantaneous rate of change, we arrive at

$$\frac{d}{dt} \langle \sigma \rangle_i (0) = \sum_{j=i_{\text{min}}}^{i_{\text{max}}} \sigma_{ij} q_{ij} \mu =: \nu_i \tag{2.18}$$

where we have defined ν_i to be the expected mutational change per time unit at this instant for repeat i.

These rates per state have to be weighted into a single mutation rate, which can be done using the stationary distribution $\boldsymbol{\pi}$. This leads to

$$\nu^* = \sum_i \pi_i \nu_i \tag{2.19}$$

In the second definition of the net mutation rate, instead of measuring it by the number that the repeats change, it can also be measured disregarding this number, and solely noticing a repeat that has changed. In that case, we weigh any change from state i to j $(i \neq j)$ as 1, which leads to the definition

$$\delta_{i,j} = \begin{cases} 1 & \text{if } i \neq j \\ 0 & \text{else} \end{cases} \tag{2.20}$$

Substituting $\delta_{i,j}$ for $\sigma_{i,j}$ in equation (2.18), we have $\nu_i = -q_{i,i}\mu$ (recall that $q_{i,i} = -\sum_{k\neq i} q_{i,k}$), and $\nu^* = -\sum_i \pi_i q_{i,i}\mu$. Normalizing $\mathbf{Q}$ to this rate is the exact same normalization used by Wu in [WD11]. Therefore, $||\mathbf{Q}|| = -\sum_i \pi_i q_{i,i}$ is the normalization we will use in our implementation of the Sainudiin model in BEAST2.

The relation with Sainudiin's average rate

In addition to the previous two definitions, Sainudiin provides an 'average rate', which only holds for the stepwise model, given by

$$\sum_i \pi_i \mu\alpha(a_0, i) \tag{2.21}$$

Looking at equation (2.18), we note that in case of the stepwise model (i.e $g = 0$) $\mathbf{Q}$ is zero except on the lower, middle and upper diagonal. Furthermore, $\boldsymbol{\sigma}$ (or $\boldsymbol{\delta}$) is zero on the middle diagonal, which means that for the stepwise model, ν_i can be rewritten as:

$$\nu_i = \begin{cases} \mu\alpha(a_1, i)\beta(b_0, b_1, i) & \text{if } i = i_{\text{min}} \\ \mu\alpha(a_1, i)(1 - \beta(b_0, b_1, i)) + \mu\alpha(a_1, i)\beta(b_0, b_1, i) & \text{if } i_{\text{min}} < i < i_{\text{max}} \\ \mu\alpha(a_1, i)(1 - \beta(b_0, b_1, i)) & \text{if } i = i_{\text{max}} \end{cases} \tag{2.22}$$

Thus, for $i_{\text{min}} < i < i_{\text{max}}$ we have

$$\nu_i = \mu\alpha(a_0, i) \tag{2.23}$$

Note that the weighted average $\sum_i \pi_i \nu_i$, is almost equal to equation (2.21) used by Sainudiin. For the first and last terms $i = i_{\text{min}}, i_{\text{max}}$ the rate of Sainudiin misses respectively a factor $\beta < 1$, $1 - \beta < 1$. It makes sense that the mutation rates on the boundary of our state space is less than inside of it: on the boundary we can mutate in only 1 direction, whilst inside we can mutate into 2 directions. This is exactly the reason why the β's do not cancel out on the boundaries in equation (2.22).

2.4 Computing the likelihood

For a given tree $\boldsymbol{\tau}$ and parameters $\boldsymbol{\theta}$, we can calculate the likelihood $f(\mathbf{D}^l|\tau, \boldsymbol{\theta})$ of finding data $\mathbf{D}^l$ on locus l in the tips of the tree, using $\mathbf{P}(t)$ and the algorithm of Felsenstein. It works by summing over possible ways of assigning states i to the internal nodes. For each term in that sum, it subsequently computes the product of $p_{i,j}(b)$ for all branch lengths b of the branches in the tree.

We can have a prior belief about the distribution of states at the root node in the tree. For example, we might believe they have uniform distribution, the empirical distribution of the states that we see in the tips of the tree, or are that they are distributed according to the stationary distribution. In addition, instead of having a prior belief about this distribution, this distribution can also be considered as extra (to be estimated) parameters to the model. This means that in the algorithm of Felsenstein, we can weigh each way of assigning states, according to these distributions, to incorporate these beliefs.

2.4.1 Connecting the likelihoods across sites

To compute the likelihood on all 24 sites, we consider them to be independent, and the parameters $a_0, r_b, i_{\text{eq}}, g$ to be identical on the 24 loci:

$$f(\mathbf{D}|\tau, \boldsymbol{\theta}) = \prod_{l=1}^{24} f(\mathbf{D}^l|\tau, \boldsymbol{\theta}) \tag{2.24}$$

However, it might be the case that some loci mutate faster than others. Beast2 has built-in an extension to the algorithm of Felsenstein to accommodate this effect. It allows the mutation rate to be distributed according to a Gamma distribution with mean μ and shape parameter α. It numerically integrates the likelihood $f(\mathbf{D}^l|\tau, \boldsymbol{\theta})$, this time with the rate parameter μ in $\boldsymbol{\theta}$ replaced by r, over all possible rates r. Thus, the likelihood $f(\mathbf{D}^l|\tau, \boldsymbol{\theta})$ computed with the Felsenstein algorithm is readjusted to give

$$f(\mathbf{D}^l|\tau, \boldsymbol{\theta}, \alpha) = \int f(\mathbf{D}^l|\tau, \boldsymbol{\theta}) f(r|\alpha) dr \tag{2.25}$$

2.5 Markov Chain Monte Carlo

Converting this likelihood in a posterior probability $f(\tau, \boldsymbol{\theta}|\mathbf{D})$, is where MCMC and Bayes' theorem come into play. The MCMC is a necessity to keep all computations feasible. It relies on Bayes' theorem:

$$f(\tau, \boldsymbol{\theta}|\mathbf{D}) = \frac{f(\mathbf{D}|\tau, \boldsymbol{\theta}) f(\tau, \boldsymbol{\theta})}{f(\mathbf{D})} \tag{2.26}$$

This expression equals the posterior probability of finding some tree τ and parameters $\boldsymbol{\theta}$ given the data $\mathbf{D}$ incorporated with the prior information $f(\tau, \boldsymbol{\theta})$.

The denominator is the sum of the enumerator over all possible trees and parameters. Computing this sum for large trees is almost impossible: we would need to consider too many possibilities.

A solution is given by the Metropolis-Hastings algorithm: we generate a chain of states such that its stationary distribution reaches the posterior distribution we are interested in. If at state $(\tau, \boldsymbol{\theta})$, we accept a proposed state $(\tau', \boldsymbol{\theta}')$ with probability

$$\min\left(1, \frac{f(\mathbf{D}|\tau', \boldsymbol{\theta}') f(\tau', \boldsymbol{\theta}')}{f(\mathbf{D}|\tau, \boldsymbol{\theta}) f(\tau, \boldsymbol{\theta})} \frac{f(\tau, \boldsymbol{\theta}|\tau', \boldsymbol{\theta}')}{f(\tau', \boldsymbol{\theta}'|\tau, \boldsymbol{\theta})}\right) \tag{2.27}$$

then the sequence of states sampled forms a Markov Chain with the posterior as stationary distribution [RC13]. Quantity $f(\tau, \boldsymbol{\theta}|\tau', \boldsymbol{\theta}')/f(\tau', \boldsymbol{\theta}'|\tau, \boldsymbol{\theta})$ is the proposal ratio, and is usually set by the user to create a balanced acceptance rate. As can be seen in equation (2.27), no computation of the denominator of equation (2.26) is needed for the generation of this chain.

2.5.1 Monte Carlo algorithm used in BEAST2

The MCMC algorithm used by BEAST2 works as follows

1. Start with some initial tree τ and parameters $\boldsymbol{\theta}$ with corresponding prior information.
2. Propose some update $\tau', \boldsymbol{\theta}'$ of the tree or parameters.
3. For the proposed parameters, create $\mathbf{Q}$ and compute $\mathbf{P}$. Then use Felsenstein's algorithm to calculate the likelihood.
4. Accept the proposed update with probability given by equation (2.27). Else, reject it. Go back to 2.

Each iteration in the above MCMC algorithm will output a tree with parameters which are sampled according to the posterior distribution. Even though we are also sampling the posterior distribution when the MCMC run starts, we definitely do not want our results to depend on the starting values chosen in step 1. Therefore, all the MCMC runs discussed in the next chapter will consist of 100.000.000 samples, of which we discarded the first 10%, which we considered by viusal inspection to be a good threshold reach a convergent MCMC chain.

Initially, BEAST2 was not capable of using the Sainudiin model described in this chapter, nor was it capable of handling VNTR data. Therefore we amended BEAST2 with a package, which implements the Sainudiin model. It is available through the package manager of BEAST2 as the package BEASTvntr, and can also be retrieved from github.com/arjun-1/BEASTvntr

— 3 —
Experiments

In this chapter we will design and perform 5 experiments on the Sainudiin model.

In experiment 3.1, we will check whether the model of Sainudiin yields trees that are similar to trees yielded by another well-established model. To this end, we first infer phylogeny for VNTR data of TB that were published before. In that publication, nucleotide data were also provided for the same samples. This means we can also infer a (reference) phylogeny using the model of Hasegawa, Kishino and Yano (HKY) for nucleotide substitutions [HKY85]. The trees obtained with the models of Sainudiin and HKY can then be compared to each other.

In experiment 3.2, we will find out whether simpler sub-models of the Sainudiin model, thus the single-step model (i.e. $g = 0$) and the unbiased model (i.e. $r_b = 0$), and more complex versions of the model, are also able to correctly infer the phylogeny and the parameters. If we can simplify the Sainudiin model, and still be able to get phylogenies with the same accuracies as for the fully featured Sainudiin model, we would prefer to use the simplified version, since that model can accurately find the phylogeny with the least amount of assumptions on the underlying mutation processes. To this end we will simulate data, and check how well we can correctly infer the original phylogeny for different versions of the model.

After finding this most parsimonious version of the Sainudiin model, we will show in experiment 3.3 how information from nucleotides and VNTR data can be combined and how this leads to a phylogeny that is better resolved than by the use of either model.

In experiment 3.4, we will compare the performance of the Sainudiin and HKY model independently by simulating data as in experiment 3.2 via an input tree and looking at how well the inferred phylogenies resembles the input trees. In this experiment, we will also investigate the effect of mutation rate on the performance of the models.

In experiment 3.5, we use a changed model for the mutation rate proportionality the results of which allow us to speculate on the mechanisms by which VNTR mutates.

Before all this can be addressed, we first need to establish how to generate a single tree from many samples from the posterior distribution of trees, and how to quantitatively (and qualitatively) compare such a tree with a reference tree.

Methods to summarise trees

For summarizing many trees into one, a lot of possibilities exist. When considering which topology is ‘best’ from an output of many trees, we might first look at the topologies which occur most often in the output. However, when dealing with very large trees or phylogenies that remain unresolved in the output, it might very well be that every sampled tree is unique, making this approach useless. Another possibility would be considering the tree which has the largest sampled posterior probability. But such a tree can have large posterior probability due to many other reasons than its topology (e.g. branch lengths). A solution might be to consider the posterior probabilities of topologies which are averaged over all the other parameters, but this again is useless when dealing with many unique topologies.

To find the tree with the best topology, we explore another solution: the maximum clade credibility tree. For every clade (this is a sub-tree that consists of an ancestor together with all its descendents) that is sampled, we can compute its *support* by calculating the proportion of all the samples of trees in which it occurs. For a sampled tree, the clade credibility (CC) score is given by the logarithm of the products of these proportions for all clades:

$$\text{CC score} = \log\left(\prod_{\text{all clades } c \text{ in } \tau} p_c\right) \tag{3.1}$$

$$p_c = \frac{\text{number of trees in which clade } c \text{ occurs}}{\text{total number of trees}} \tag{3.2}$$

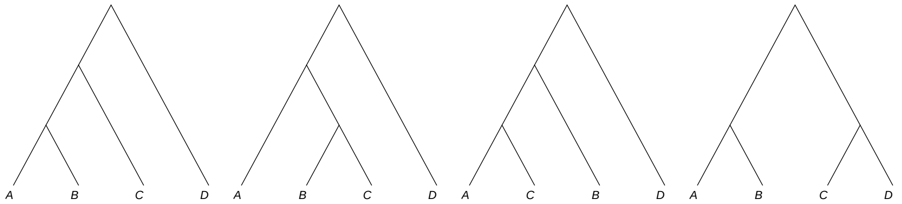
The maximum clade credibility (MCC) tree is then defined as the tree for which the CC score is maximum. In the case of the 4 sampled trees in figure 3.1, tree (1) has the highest CC score of $\log(\frac{1}{2} \cdot \frac{3}{4}) \approx -0.98$.

A higher CC score of the MCC tree implies a tree that is inferred with higher confidence according to the model, since the clades in that particular tree must have been sampled more often. This is explained in more detail in section 3.3.

A program which can perform this summary of trees is *TreeAnnotator* and is included in BEAST2.

Methods to compare trees

When visually comparing trees, note that even identical trees can be represented in such a way that they would appear dissimilar by simply rotating the clades around their branches. A visual comparison of two trees, represented so that they would appear most similar, can be performed by the *dendextend* [Gal15] package in R. It produces so-called *tanglegrams*: two opposed dendrograms whose leaves are connected. The tanglegrams connect corresponding tips in colour if they belong to the same sub-tree. Furthermore, nodes which contain some tips, which the node in the other tree does not contain, are highlighted with a dashed line.



(1) (A,B); (A,B,C) (2) (B,C); (A,B,C) (3) (A,C); (A,B,C) (4) (A,B); (C,D)

Figure 3.1: An Example of 4 sampled trees, with their clades listed. Clades consisting of the entire tree, or only a single leave, are omitted.

We use the Robinson-Foulds (RF) metric [RF81] to quantitatively compare trees. For two trees τ_1, τ_2 it is defined as $d_{\mathrm{RF}}(\tau_1, \tau_2) = A + B$ where A is the number of clades in τ_1 but not in τ_2 and B is the number of clades in τ_2 but not in τ_1. Observe that we measure the difference between trees using clades, just as we did when summarizing trees using the clade credibility score of equation (3.1). for trees τ_1, τ_2 which have n tips, we always have that $0 \leq d_{\mathrm{RF}}(\tau_1, \tau_2) \leq 2n - 4$.

3.1 Comparing the phylogenies of the Sainudiin and the HKY model

In this experiment we will check whether the phylogeny obtained with the Sainudiin model resembles a reference phylogeny, obtained from the HKY model.

The HKY model is a substitution model for the evolution of nucleotides, and specifies the relative rates of mutations per site. Furthermore, the rates for transitions A ↔ G and C ↔ T are differentiated with a factor κ, leading to the definition of rate matrix

$$\mathbf{Q}_{\mathrm{HKY}} = \begin{array}{ccccc} & \mathrm{T} & \mathrm{C} & \mathrm{A} & \mathrm{G} \\ \mathrm{T} & -\sum_{j\neq \mathrm{T}} q_{\mathrm{T},j} & \kappa\pi_{\mathrm{C}} & \pi_{\mathrm{A}} & \pi_{\mathrm{G}} \\ \mathrm{C} & \kappa\pi_{\mathrm{T}} & -\sum_{j\neq \mathrm{C}} q_{\mathrm{C},j} & \pi_{\mathrm{A}} & \pi_{\mathrm{G}} \\ \mathrm{A} & \pi_{\mathrm{T}} & \pi_{\mathrm{C}} & -\sum_{j\neq \mathrm{A}} q_{\mathrm{A},j} & \kappa\pi_{\mathrm{G}} \\ \mathrm{G} & \pi_{\mathrm{T}} & \pi_{\mathrm{C}} & \kappa\pi_{\mathrm{A}} & -\sum_{j\neq \mathrm{G}} q_{\mathrm{G},j} \end{array}$$

The diagonals are defined such that the rate of probability flowing out of a state is equal to the total inflow to other states.

Since nucleotide data is not constrained to 24 sites, it is expected to contain much more information than VNTR and can therefore serve to generate a reference tree.

3.1.1 Methods

The data we use comes from a collection of 97 MTBC strains, and was published in a study by Comas [Com+09].

For these isolates, both nucleotide and VNTR data were made available in this study. The VNTR data consists of 24 loci, and was sampled specially for this study. The nucleotide data was obtained from a previous study of Hershberg [Her+08], and consisted of 339 chosen bases which showed difference among the 97 (such differences are called single-nucleotide polymorphism, or SNP). In the study of Hershberg, the DNA of several genes of 108 MTBC strains (including the 97 sampled in the Comas study) was sampled using MLST.

Of these 108 strains, 99 included human-adapted strains from a global collection of 875, and 7 were selected to represent animal-adapted strains. From the VNTR strains of the Comas study, we removed any strains with missing data, or containing $i = 0 < i_{\min}$, leaving 92 strains. Of these 92 strains, only 5 were animal-adapted strains. The distribution of the states in the data is shown in figure 3.5.

We reconstructed the entire original nucleotide sequences of the samples, using the 339 SNPs. To this end, we created a dataset which consists of the 339 SNPs from the Comas dataset, filled to its original sequence length of 65.829 bases, by randomly drawing nucleotides from the sequences of the genes that were originally sampled. These genes were listed in the study of Hershberg [Her+08], and were obtained from [Gal+10].

The settings of BEAST2 to create the MCMC runs are explained below.

Set-up of BEAST2

The priors for the parameters $a_0, r_b, i_{\text{eq}}, g$ of the Sainudiin model, are chosen such that they reflect no prior knowledge on the parameters. For the HKY model, we use the default log-normal prior for κ. All the priors are shown in table 3.1.

A parameter in table 3.1 not yet discussed is population size. BEAST2 assumes the tree to be the result of some population model, which in turn puts a prior probability $f(\tau)$ on tree τ. We used the Wright-Fisher model which assumes a constant effective population size (a common choice), which means that the number of reproducing individuals in a population remains constant.

Note that our data has no time information. This means that estimating either μ or the tree height is impossible, as μ and t appear together in equation (2.1). For that reason, we set $\mu = 1.0$, so that we implicitly estimate quantity $\mu\times$tree height, instead of tee height.

We furthermore assume that the states at the root node are distributed according to the stationary distribution of matrix $\mathbf{Q}$.

Parameter	Prior
a_0	$\mathcal{U}(0.0, 10.0)$
r_b	$\mathcal{U}(0.0, 10.0)$
i_{eq}	$\mathcal{U}(1.0, 15.0)$
g	$\mathcal{U}(0.0, 1.0)$
κ	$\ln\mathcal{N}(1.0, 1.25)$
Population size	$1/x$[1]
Gamma shape	Exp(1)
μ	1.0

Table 3.1: Priors used on the parameters of the HKY and Sainudiin model, for inferring the phylogeny of the Comas data.

We use the output of trees to generate an MCC tree for both models and subsequently use a tanglegram to compare them.

3.1.2 Results

The posteriors of the parameters of the Sainudiin model are shown in table 3.2. We notice that the obtained value for a_0 closer to 0 than to 1. If it were the case that $a_0 = 1$, then for $i_{\text{min}} = 1$, it follows from equation (2.5) that the mutation rate of repeat i is proportional to the number of repeats i:

$$\alpha(1, i) = i\alpha(1, 1) \tag{3.3}$$

[1] For the population size, $1/X$ is a Jeffrey's prior [DB15], meaning it is a 'good' non-informative prior, i.e. the posterior produced from it, best reflects the information that is in the data.

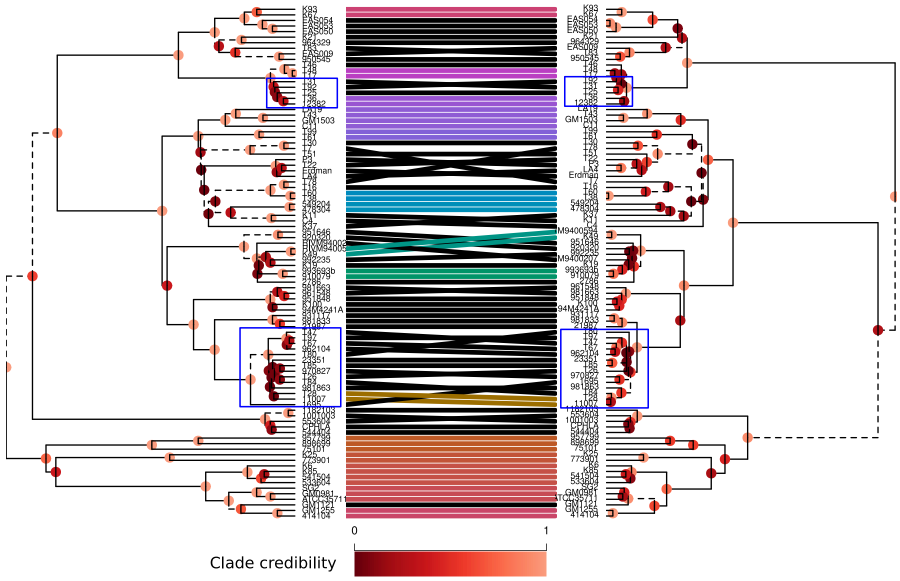


Figure 3.2: Tanglegram of the inferred phylogenies of the Comas dataset. Left: the nucleotide tree, CC score = -71.21. Right: the VNTR tree CC score = -93.38. The parts marked in blue are where the clades in the VNTR tree have a higher CC score than the nucleotide tree.

If however $a_0 = 0$, it follows that the mutation rate of repeat i is proportional to the number of repeats $- 1$.

$$\alpha(0, i) = (i - 1)\alpha(0, 2), \tag{3.4}$$

in which case state $i = 1$ is an absorbing state. Thus it appears the rate is more proportional to the rate of $i - 1$, which we will investigate further in experiment 3.5.

The generated tanglegram for the MCC trees obtained using the model of Sainudiin and the HKY model are shown in figure 3.2. At first sight, the two trees appear to be similar quite similar, but not identical. There are parts in the VNTR tree marked in blue, where the clades have a higher CC score, than in the nucleotide tree, even though the overall CC score of the VNTR tree is lower than that of the nucleotide tree. This suggests these parts of the tree are better determined in the VNTR tree, which will be further explained and investigated in experiment 3.3.

The Robinson-Foulds distance between the two trees is $d_{\mathrm{RF}} = 104$. In Experiments 3.2 and 3.4 we will investigate the performance of the Sainudiin model together with another model, on simulated data, which will help to interpret this obtained value.

Parameter	Posterior	
	Median	95% HPD Interval
a_0	0.11	$[5.35 \cdot 10^{-6}, 0.45]$
r_b	0.35	$[0.16, 0.72]$
i_{eq}	1.88	$[1.00, 2.79]$
g	0.36	$[0.31, 0.42]$
Population size	2.47	$[1.65, 3.45]$
Gamma shape	1.71	$[0.94, 2.50]$
Tree height	0.91	$[0.57, 1.36]$

Table 3.2: Posteriors of the parameters in the Sainudiin model, found when inferring the phylogeny of the Comas data.

3.2 The performance of simpler and more complex versions of the Sainudiin model

In this experiment we will investigate whether simpler versions of the Sainudiin model can still correctly infer the trees and parameters, and whether a more complex locus model will increase the accuracy of the found phylogeny. For example, the model is capable of modelling mutational bias via parameters r_b, i_{eq}, but it might be possible that such an effect is not necessary when inferring the phylogeny. To make the bias for expansion and contraction is always the same, we can set $r_b = 0$. To investigate such possible simplification, we have simulated data and inferred the phylogenies using different model settings.

Also, until now we have implicitly assumed that all model parameters $\boldsymbol{\theta}$ are the same on all 24 loci, only allowing for rate variation via equation (2.25). Because in our data, the number of repeats on the different loci showed a large variation in distributions, it is possible that estimating the model parameters, $\mu^l, i^l_{\text{eq}}, g^l$ for each locus l separately, increases the accuracy of the found phylogeny. Also, knowing the model parameters for each locus separately allows us to simulate data with larger variation, thus more realistically. Adding heterogeneity for the other parameters r_b, a_0 as well resulted in too many parameters being estimated which caused numerical instability in the MCMC chain in BEAST2.

The likelihood that is used in this way in the updating algorithm of section 2.5.1, is given by the product of the likelihood for all loci

$$f(\mathbf{D}|\tau, \boldsymbol{\theta}^1, \ldots, \boldsymbol{\theta}^{24}) = \prod_{l=1}^{24} f(\mathbf{D}^l|\tau, \boldsymbol{\theta}^l) \tag{3.5}$$

3.2.1 Methods

We will first estimate the model parameters of the Comas dataset for each locus separately, using settings as described in section 3.1, but with different parameters μ, i_{eq}, g on all loci, and r_b, a_0 the same.

To simulate data, we use the tree obtained from nucleotide data in section 3.1. Since the tree obtained from nucleotide data is of different height than the tree obtained of VNTR data, for $\mu = 1$, the nucleotide tree is rescaled to have length identical to the VNTR tree obtained in section 3.1.

Using the median of the parameters $\mu^l, i^l_{\text{eq}}, g^l$ and r_b, a_1, shown in table 3.3, we construct $\mathbf{Q}$ for each of the 24 loci, having states i for[2] $i_{\text{min}} = 0 \leq i \leq i_{\text{max}} = 14$.

Before we can start simulating data, we must first specify a sequence for the root node. For this sequence we choose the repeats that occur most frequently on each locus in the data $\mathbf{D}$.

BEAST2 then simulates data in the following way:

- For each branch in τ, calculate the probability transition matrix $\mathbf{P}(b)$ from $\mathbf{Q}$ where b is the corresponding branch length. Descending from the root node, we can calculate from $\mathbf{P}$, for each locus l, the distribution of the repeats for the children of that node. By taking a draw from that distribution we simulate some sequence for the children of that node.
- Descending through all the nodes, we generate sequences for all the nodes in the tree, and ultimately for the tips. The sequence data on the tips of the tree constitutes the simulated data.

The above procedure is repeated 32 times, to generate 32 datasets. These simulated data are analysed in the following ways:

- using the same model parameters for each locus (homogeneous)
- using separate model parameters μ, i_{eq}, g for each locus (inhomogeneous)

Combined with:

- using no mutational bias, i.e. $r_b = 0$, or
- using mutational bias.

Combined with:

- using the single step model, i.e. $g = 0$, or
- using the multi step model.

Thus in total each dataset is analysed in 8 different ways. From the generated posteriors, we will compare the Robinson-Foulds distances between the MCC trees and the tree used for simulating. We will also compare the posteriors of the parameters with the parameters used for simulating.

3.2.2 Results

The posteriors of the estimated parameters used of the most heterogeneous and complex model (the results of which were used for simulating data) are shown in figure 3.4 and their median values in table 3.3.

In figure 3.3, the Robinson-Foulds distance between the MCC tree and the tree used for simulating is plotted for each dataset. By looking at the mean of the different analyses in this figure, we can see that from all versions of the model only the multi-step model yields a significant decrease in Robinson-Foulds distance, since the difference in results for the multi step model and single step model are greater than the standard error. When the mutational bias is modelled in the analysis, the

[2] This is equivalent to setting $i_{\text{min}} = 1 \leq i \leq i_{\text{max}} = 15$, as done when analysing the Comas data.

inferred phylogenies do not substantially become more accurate. Also, we can see that using inhomogeneous parameters for the loci does not yield a significant better result compared to using homogeneous parameters. Thus we can conclude, that for inferring the phylogeny, it is not necessary to model the mutational bias, or use site heterogeneity.

For each simulated dataset for the analysis with bias, the multi-step model, and inhomogeneous parameters, the 93.75%[3] confidence interval of the mean of the posteriors, and the average of the mean of the posteriors were also calculated. They are also shown in table 3.3. As can be seen, parameter g can be quite accurately determined, and parameter μ somewhat less, but the confidence intervals show that the precision is quite low for all obtained parameters.

In table 3.4, we show the same parameters of the analysis of the model that was deemed most parsimonious, i.e., using homogeneous parameters and the multi-step model without bias. For completion, we have also listed results for the model with bias. As can be seen, a_0 is less accurately determined for the case of unbiased model, compared with the biased model. A possible explanation is that without the extra rate coming from the bias as in equation (2.10), on repeat $i = 1$, the missing rate might get compensated by an increase in a_0.

Re-inferring the phylogeny of the Comas data using the most parsimonious model

When we re-infer the phylogeny as done in Experiment 3.1, using a version of the Sainudiin model which was deemed most parsimonious, (i.e., with homogeneous parameters and without bias), we obtain posteriors of the parameters as shown in table 3.5. The Robinson-Foulds distance to the phylogeny inferred using nucleotides (as done in Experiment 3.1), amounted $d_{\mathrm{RF}} = 104$.

From now on, whenever we infer the phylogeny using the Sainudiin model, we always use this most parsimonious model.

3.3 Joining information in VNTR and nucleotides

As suggested in section 3.1, and shown in figure 3.2, with VNTR data we can infer the phylogeny of different parts of the tree with different results. In this experiment, we shall combine the information coming from both VNTR and nucleotide data, and find out whether the phylogeny obtained with the HKY model using nucleotide data can benefit from the VNTR data.

Since there is no reference tree available beside the tree generated from nucleotide data, we have no comparison for evaluating the MCC trees obtained from both models. Instead we will look at the *precision* or variability of the obtained trees from the MCMC output. More variability being present in the obtained trees, indicates less information being present in the data. A natural candidate for measuring this information content is the clade credibility score, discussed at the beginning of this chapter. If the genetic data would be void of any phylogenetic information, we would expect the MCMC algorithm to sample each possible tree with the same probability. Since number of possible trees is very large, this would result in the MCC tree having a very low clade credibility score. If in contrary the data is very rich in phylogenetic information, we would imagine that the MCMC

[3] Since we are dealing with 32 datasets, $30/32 = 93.75\%$

VNTR	g			μ			i_{eq}		
	True	Mean	93.75% CI	True	Mean	93.75% CI	True	Mean	93.75% CI
1	0.21	0.41	[0.25,0.59]	0.25	0.07	[0.02,0.11]	7.07	6.98	[6.09,7.81]
2	0.51	0.47	[0.28,0.67]	1.01	0.89	[0.51,1.36]	2.40	3.47	[1.06,7.61]
3	0.16	0.17	[0.06,0.29]	1.26	0.83	[0.48,1.42]	3.11	3.52	[1.52,5.31]
4	0.15	0.22	[0.11,0.46]	0.73	1.62	[0.80,2.65]	0.45	1.58	[0.50,4.47]
5	0.52	0.48	[0.22,0.75]	0.50	0.92	[0.26,1.47]	0.56	2.63	[0.77,5.79]
6	0.40	0.39	[0.21,0.63]	0.30	0.07	[0.04,0.12]	8.75	7.34	[4.07,9.36]
7	0.52	0.53	[0.40,0.65]	0.22	0.34	[0.19,0.63]	5.82	6.27	[5.35,6.76]
8	0.42	0.40	[0.16,0.61]	0.73	1.28	[0.76,1.89]	0.92	1.92	[0.76,3.39]
9	0.32	0.42	[0.22,0.65]	0.20	0.06	[0.02,0.15]	6.48	6.88	[4.31,8.37]
10	0.27	0.24	[0.09,0.41]	0.83	1.14	[0.64,1.55]	1.47	2.43	[1.03,3.98]
11	0.08	0.18	[0.07,0.37]	0.56	0.50	[0.26,0.76]	2.04	4.09	[1.73,6.38]
12	0.07	0.19	[0.08,0.36]	2.00	2.28	[1.62,3.52]	1.17	1.62	[0.90,2.87]
13	0.32	0.37	[0.18,0.56]	2.19	2.30	[1.05,3.27]	1.60	2.32	[1.13,4.56]
14	0.22	0.24	[0.13,0.44]	0.28	0.12	[0.05,0.21]	4.64	6.50	[3.63,9.30]
15	0.30	0.29	[0.16,0.42]	2.46	1.40	[0.98,2.14]	4.02	5.04	[3.26,6.68]
16	0.52	0.54	[0.37,0.69]	2.42	1.49	[1.02,2.19]	3.88	4.58	[1.67,7.37]
17	0.38	0.31	[0.17,0.46]	1.10	1.29	[0.73,2.09]	2.06	3.44	[1.87,5.64]
18	0.37	0.37	[0.15,0.63]	0.20	0.10	[0.05,0.22]	3.95	6.04	[2.69,8.67]
19	0.64	0.59	[0.22,0.78]	0.34	0.56	[0.36,0.89]	1.10	2.37	[0.97,6.60]
20	0.34	0.43	[0.23,0.68]	1.20	1.68	[0.98,2.43]	0.90	2.17	[0.86,4.28]
21	0.24	0.29	[0.10,0.59]	0.48	0.59	[0.33,0.95]	1.75	2.74	[1.24,5.28]
22	0.39	0.42	[0.27,0.62]	1.86	1.27	[0.82,1.76]	3.49	3.97	[2.77,6.23]
23	0.45	0.39	[0.21,0.58]	1.55	1.23	[0.79,1.62]	3.33	4.96	[3.03,7.25]
24	0.52	0.48	[0.24,0.71]	1.32	1.97	[0.82,3.90]	0.84	2.00	[0.98,4.86]

(a) The inhomogeneous model parameters.

r_b			a_0		
True	Mean	93.75% CI	True	Mean	93.75% CI
0.89	1.11	[0.59,1.45]	0.16	0.22	[0.07,0.43]

(b) The homogeneous model parameters.

Table 3.3: The true parameters are the medians of the found posterior of the parameters of the Sainudiin model (using inhomogeneous and homogeneous parameters), for simulated VNTR data. Listed are also the average of the mean of the parameters and their confidence interval.

algorithm only samples a few, obvious trees. This in turn would result in a MCC tree with a very high clade credibility score.

3.3.1 Methods

In BEAST2, combining the two models can be done in a very straightforward manner. The trees inferred from the HKY and the Sainudiin model can be linked: this means that the same tree is being used for updating the state in the MCMC algorithm. The MCMC algorithm still looks as described in section 2.5.1, except that for the

	g		r_b			i_{eq}		a_0		
Analysis	Mean	93.75% CI	True	Mean	93.75% CI	Mean	93.75% CI	True	Mean	93.75% CI
bias	0.33	[0.25,0.38]	0.89	0.51	[0.30,0.90]	1.83	[1.38,2.36]	0.16	0.25	[0.09,0.43]
no bias	0.27	[0.20,0.34]						0.16	0.46	[0.24,0.75]

Table 3.4: The true parameters are the medians of the found posterior of the parameters of the most parsimonious Sainudiin model, using homogeneous parameters, for simulated VNTR data. Listed are also the average of the mean of the parameters and their confidence interval.

Parameter	Posterior	
	Median	95% HPD Interval
a_0	0.23	$[0.00011, 0.87]$
g	0.31	$[0.26, 0.37]$
population size$\times\mu$	2.44	$[1.51, 3.54]$
gamma shape	1.6577	$[0.92, 2.47]$
tree height$\times\mu$	0.71	$[0.41, 1.12]$

Table 3.5: Posteriors of the parameters in the Sainudiin model without mutational bias, found when inferring the phylogeny of the Comas data.

updating of trees and parameters, it uses a likelihood that is the product of the tree likelihoods for the HKY and Sainudiin model:

$$f(\mathbf{D}_{\mathrm{HKY}}, \mathbf{D}_{\mathrm{Sai}}|\tau, \boldsymbol{\theta}_{\mathrm{HKY}}, \boldsymbol{\theta}_{\mathrm{Sai}}) = f(\mathbf{D}_{\mathrm{HKY}}|\tau, \boldsymbol{\theta}_{\mathrm{HKY}}) f(\mathbf{D}_{\mathrm{Sai}}|\tau, \boldsymbol{\theta}_{\mathrm{Sai}})$$

3.3.2 Results

We compare to the MCC tree inferred with solely nucleotide data to the MCC trees inferred to any dataset of {VNTR, VNTR and nucleotide} in figure 3.6. The nodes of the trees are highlighted with their respective clade credibility score.

As can be seen in figure 3.6a, the clade credibilities of the nucleotide tree are in general higher than those of the VNTR tree, which is confirmed by the nucleotide tree having a higher CC score than the VNTR tree. However, it is also clear that there are some parts of the nucleotide tree with low clade credibility, marked in blue, that are clearly better inferred (i.e. with higher clade credibility) in the VNTR tree.

Looking at figure 3.6b, it becomes clear that in the tree where information of nucleotides and VNTR is combined, those parts where VNTR outperformed nucleotides, are still present. In addition, those parts where the nucleotide tree was more certain about the phylogeny, are still determined by the nucleotides tree. If the information in VNTR and nucleotides was contradictory, we would expect the CC score to decrease, since the MCMC algorithm would sample more, different trees. Since the CC score of the combined tree is higher than that of both the nucleotide and the VNTR tree, this means the information in both the VNTR and nucleotides data is complementary.

3.4 The effect on mutation rate on the inferred phylogeny and the performance of the Sainudiin model compared to the HKY model

In this section, we will investigate the effect of $\lambda = \mu t$ on the performance of the Sainudiin model, and also perform a comparison with the performance of the HKY model. Large values of λ, cause the rows of $\mathbf{P}(\lambda) = e^{\mathbf{Q}\lambda}$ to reach the stationary distribution. Therefore, λ is a parameter which models the mutational saturation of the data. Since VNTR practically only occupies a limited number of states, having too much saturation could mean we lose the discriminatory power that the repeats have to infer the phylogeny. To investigate this effect, we will simulate VNTR data and see how the d_{RF} between the inferred and true phylogeny behaves for different λ (or equivalently, for different μ and fixed t).

Another reason for varying λ, is that we can simulate different scales of an outbreak. For example, smaller values of λ represent data of outbreaks on a local scale, while as λ increases, it represents data sampled across lineages, and finally across different TB species.

By also simulating nucleotide data for different μ, and seeing how d_{RF} behaves for this data, we can see how much worse or better the Sainudiin model performances compared to the HKY model. We can also infer the phylogeny using the HKY and Sainudiin model combined, and using nucleotide and VNTR data, as done in section 3.3. Then, by again looking at d_{RF} between the inferred and true phylogeny, we can see how beneficial the VNTR data is compared to solely nucleotides.

3.4.1 Methods

We simulate VNTR data as described in section 3.2.1. However, we will transform the mutation rate: $\mu^l \to m\mu^l$, and vary m when simulating.

We will also simulate nucleotide data. Using the Comas data and the HKY model in BEAST2, we can infer the phylogeny and the model parameters as done in section 3.1. For this tree and parameters, simulated data can be generated in the same way as described in section 3.2.1. Also, when simulating nucleotide data, we will transform $\mu \to m\mu$, and vary m.

3.4.2 Results

In figure 3.7a we show the resulting computed values of d_{RF} for different λ of simulated VNTR data. For very low and very high mutation rate, the computed distance d_{RF} is high which means that the model performs badly. Of special interest is the minimum that appears. Apparently there is an optimal amount of (mutational) change that can happen to a tree, such that this change is optimal in the sense that it is most 'helpful' for the inference of the phylogeny.

Compared to the case of simulated nucleotide data in figure 3.7b, we can see d_{RF} decreases with increasing μ. Furthermore, at $\lambda = \lambda_{\mathrm{comas}}$, we can see that we obtain $d_{\mathrm{RF}} \approx 50$ for nucleotide data, and $d_{\mathrm{RF}} \approx 100$ for VNTR data. This means that using HKY on the Comas dataset will yield a phylogeny which is twice as close to the true phylogeny compared to using the Sainudiin model.

In figure 3.7b, we can also see that for saturation levels λ of that of the Comas data, having the phylogeny inferred using both VNTR and nucleotide yields phylogenies that have substantially less distance to the true phylogeny, than solely using nucleotides. This means that VNTR in combination with nucleotides has substantially more information on the phylogeny, than solely nucleotides, for this level of saturation. We also see that around 1000 SNPs, nucleotides become so rich in information, that VNTR no longer contributes any information not already contained in the nucleotides.

To further investigate the possible criteria which can be associated with an optimal mutational change for VNTR data, we look at the inferred phylogenies of the red, green and blue dot of figure 3.7a, and compare them with the original phylogeny using tanglegrams.

Closer inspection of the trees

We compare the inferred phylogenies of the red, green and blue dot in figure 3.8. It can clearly be seen that the inferred phylogeny of the green dot in figure 3.8b is the best. Although the inferred phylogeny of the red dot in figure 3.8a has not found the neighbours of the tips very well, it has been quite capable of finding the structure that defines the clades, when compared to the inferred phylogeny of the blue dot in figure 3.8c. Looking at the inferred phylogeny of the blue dot in figure 3.8c, the many colours indicate that many sub-trees, i.e. the neighbours of the tips have been found. However, the many crossings of lines between the tips indicate that the clades of larger size could not correctly be inferred.

Looking at the original tree in the left of figures 3.8a to 3.8c, we can see that longer branches are present higher up in the tree, whereas the branches closest to the tips are clearly shorter.

If μ is large, as is the case with the blue dot, what happens is that the larger branches will become over-saturated with mutations. However, along the short branches near the tips, only a few mutations happen, which will be enough to discriminate the closest family members in the tree. This is why we can clearly see close neighbours are correctly identified for the inferred phylogeny of the blue dot in figure 3.8c, whereas the more distant clades, which are separated by the longer branches, are not.

If μ is small, as is the case with the red dot, what will happen is exactly the opposite: only along the long branches a few mutations will happen, whereas along the short branches no mutations will happen at all. This is why the inferred phylogeny of the red dot in figure 3.8a correctly identifies the larger clades, but not the neighbours.

Summarizing, it is important that branch length times mutation rate μ is not too small, but not too big either. Thus we could say that is essential that as many branches as possibles become not over-saturated and not under-saturated.

Quantifying a criterion for optimal mutational change

We want to count the number of branches that are 'helpful' in inferring the phylogeny. For a branch of (time) length b, the number of mutations along it is given by μb. The mutations are only helpful when there are not too few, and not too many of them. This means that only branches for which

$$\lambda_{\text{lower}} \leq \mu b \leq \lambda_{\text{upper}} \tag{3.6}$$

holds, are helpful in inferring the phylogeny. To find the bounds $\lambda_{\text{lower}}, \lambda_{\text{upper}}$, we will look (for a given μ) at the arrival times $\underline{t}, \bar{t}$ of the mutations between which they are still 'helpful'.

If we assume that only a single mutation along a branch is helpful for the inference of the phylogeny, the values $\underline{t}, \bar{t}$ can be chosen to reflect this assumption. Our choice is to look at the arrival time, of the first mutation on any locus, defined by T_{min}, and at the arrival time, of the mutation on the last locus to receive its first mutation, defined by T_{max}. Of course $T_{\text{min}}, T_{\text{max}}$ are random variables, and depend on T_1^l, the arrival time of the first mutation on locus l, via:

$$T_{\text{min}} = \min_l T_1^l \tag{3.7}$$

$$T_{\text{max}} = \max_l T_1^l \tag{3.8}$$

We will now proceed to find the 5% and 95% percentiles of these distributions, and subsequently use these values in determining $\underline{t}\mu, \bar{t}\mu$.

The distribution of the time of the k-th arrival T_k is Erlang:

$$f_T(x; k, \mu) = \frac{\mu^k x^{k-1} e^{-\mu x}}{(k-1)!}$$

Assuming independence of the 24 loci, we can easily find the CDF of $T_{\text{min}}, T_{\text{max}}$ by computing

$$F_{T_{\text{min}}}(x; \mu) = 1 - \prod_{l=1}^{24} (1 - F_T(x; 1, \mu))$$

$$F_{T_{\text{max}}}(x; \mu) = \prod_{l=1}^{24} F_T(x; 1, \mu)$$

We then choose $\underline{t}, \bar{t}$ such that

$$F_{T_{\text{min}}}(\underline{t}; \mu) = 5\%$$

$$F_{T_{\text{max}}}(\bar{t}; \mu) = 95\%$$

holds. These values reflect the percentiles, such that with 95% probability the first mutation on the 24 loci has already arrived, and with 95% probability the event that the last locus to receive its first mutation has not yet occurred. Note that we need to specify μ to solve this equation, but since $\underline{t}\mu$ and $\bar{t}\mu$ are always constant for different choices of μ, this is no issue. Using the `FindRoot` routine of *Mathematica* [Wol16] we find

$$\underline{t} = 0.0021 \tag{3.9}$$

$$\bar{t} = 6.1493 \tag{3.10}$$

We can then count the loci satisfying criterion equation (3.6):

$$|\{b : \underline{t} \leq b\mu \leq \bar{t}\}| \tag{3.11}$$

In figure 3.7c we have plotted the number of branches satisfying equation (3.11) with the above values. As can be seen, this quantity can indeed reflect the pattern of the d_{RF} distance in figure 3.7a.

3.5 Investigating the mutation rate proportionality

Recall that the mutation rate in equation (2.5) is modelled to be linearly dependent on the number of repeats i, where the rate proportionality of i_{min} is a_0:

$$\alpha(a_0, i) = a_0 + i - i_{\text{min}}$$

The low values for a_0 in table 3.2 from the results of experiment 3.1, indicated that the mutation rate is (almost) proportional to the number of repeats − 1, since $a_0 = 0$ implies

$$\alpha(0, i) = (i - 1)\alpha(0, 2) \tag{3.12}$$

The biological interpretation of this is that the mutation rate is proportional to the number of edges between the repeats ($i - 1$). Note however, that an exact proportionality to $i - 1$, implies a mutation rate of 0 for state $i = i_{\text{min}} = 1$, i.e., it becomes an absorbing state. For this reason, the rate that repeat $i = 1$ has, is reflected in a_0, and can cause a_0 to increase.

This experiment will serve the purpose to determine whether the mutation rate is proportional to either the number of repeats (i), or the number of edges in between them ($i - 1$). The current model for the mutation rate proportionality, cannot answer this since it forces $a_0 > 0$, otherwise state $i = 1$ would become an absorbing state. Therefore, we will use a changed model for the mutation rate proportionality α to circumvent the issue of $i = 1$ becoming an absorbing state whenever there is proportionality to $i - 1$. We start by giving an explanation for the mutation mechanism of VNTR below, and subsequently design an experiment to verify the proportionality that this mechanism implies.

The mutation rate proportionality is modelled after the proportional slippage model of the repeats. During the replication of a DNA strand, either the original or new strand might slip, causing the repeats to misalign. This mechanism requires at least two repeats, and it could be reasonable to assume that each pair of repeats has an independent probability to mutate.

If we believe that a larger stretch of DNA offers (proportionally) more opportunity to mutate, we can model the mutation rate to (linearly) increase with the number of edges between the repeats ($i - 1$), as was done by Kruglyak [Kru+98]:

$$\tilde{\alpha}(\mu_1, i) = \mu_1(i - 1) \tag{3.13}$$

To still allow the number of repeats to increase from $i = 1$ to another state, an additional constant term might be added:

$$\tilde{\alpha}(\mu_1, i) = \mu_0 + \mu_1(i - 1) \tag{3.14}$$

This expression is equivalent to equation (2.3), since they only differ by a constant, which is captured in μ_0.

Even though this expression models the proportionality of the rate to the number of repeats, it is yet unclear *what* makes it proportional. For convenience, let's only consider single step mutations. A possible explanation then is that each of the repeats of one string, might misalign and map to the nearest neighbour of the opposing string, as shown in figure 3.10. Considering all possible opportunities of misalignment, by looking at figure 3.11, we can see that the rate must become proportional to $i - 1$. If a_0 is (almost) 0, then by equation (2.5) state $i = 1$ must (almost) be an absorbing state. This makes sense in the explanation of figure 3.11, since (for the mechanisms shown in this figure) there is no possible way of misalignment of state $i = 1$.

To still allow state $i_{\text{min}} = 1$ to mutate, implies a separate rate for repeat $i = 1$, while the rate of repeat ≥ 2 would be given by equation (3.12). To verify whether the mutation rate is proportional to either the number of repeats i, or the number of edges in between them $i - 1$, we change the model of the mutation rate proportionality as in 2.3, to

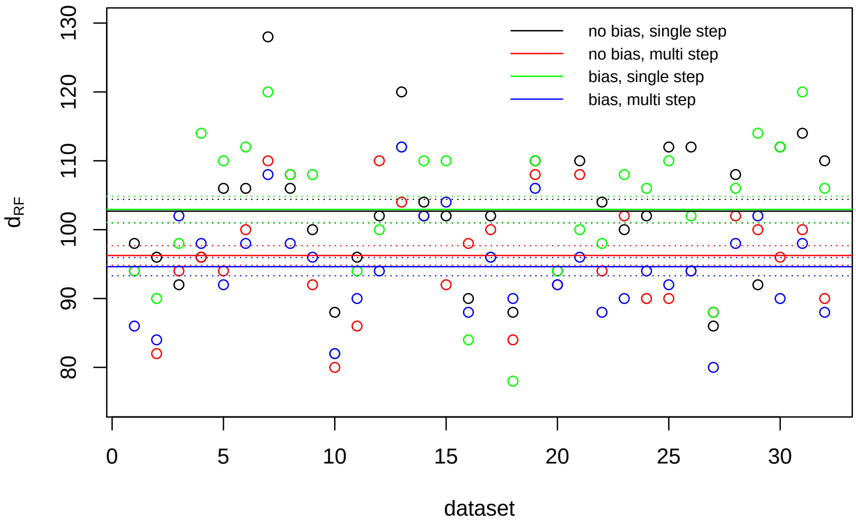
$$\tilde{\alpha}(\mu_0', \mu_0, \mu_1, i) = \begin{cases} \mu_0' & \text{if } i = 1 \\ \mu_0 + \mu_1(i-1) & \text{if } i \geq 2 \end{cases}$$

Which can be normalized to

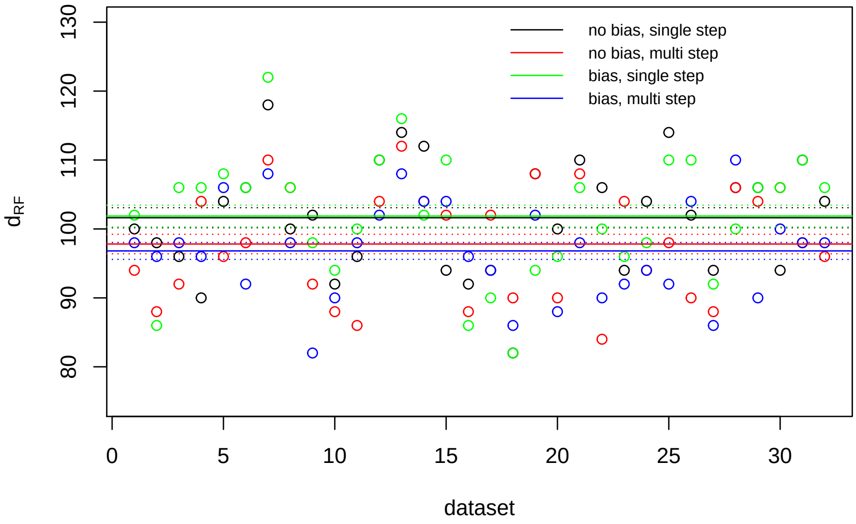
$$\alpha(a_0', a_0, i) = \begin{cases} a_0' & \text{if } i = 1 \\ a_0 + i - 1 & \text{if } i \geq 2 \end{cases} \tag{3.15}$$

One expects $a_0 = 0$ in the posterior, since then $\alpha(a_0', 0, i) = (i-1)\alpha(a_0', 0, 2) \propto (i-1)$, for $i \geq 2$. If in contrary, it were the case that the mutation rate of state i would be proportional to i, we would expect to find $a_0 = 1$.

When we perform the MCMC run with this changed model for the Comas dataset, using uniform wide priors on a_0', a_0, and further settings as described in section 3.1, we find posteriors for a_0', a_0 as shown in figure 3.9a. The median and 95% HPD interval for a_0 are 0.02 and $[-0.55, 0.97]$. The found values for a_0 do indeed suggest that in nature, the mutation rate of the repeats is proportional the number of edges between them. Also, $a_0 = 0$ implies equation (3.12). This means the parametrization as in 2.5 has sufficient parameters to model the mutation rate dependency on repeat length, i.e. there is no need for separate parametrisation for the proportional rate of $i = 1$.



(a) Analysis using homogeneous parameters.



(b) Analysis using inhomogeneous parameters.

Figure 3.3: Robinson-Foulds distance between input tree and the MCC tree, obtained from 4 different versions of the Sainudiin model that were used for inferring the phylogeny of simulated VNTR data. The solid line represents the mean of a type of analysis, the dashed lines represent the standard error.

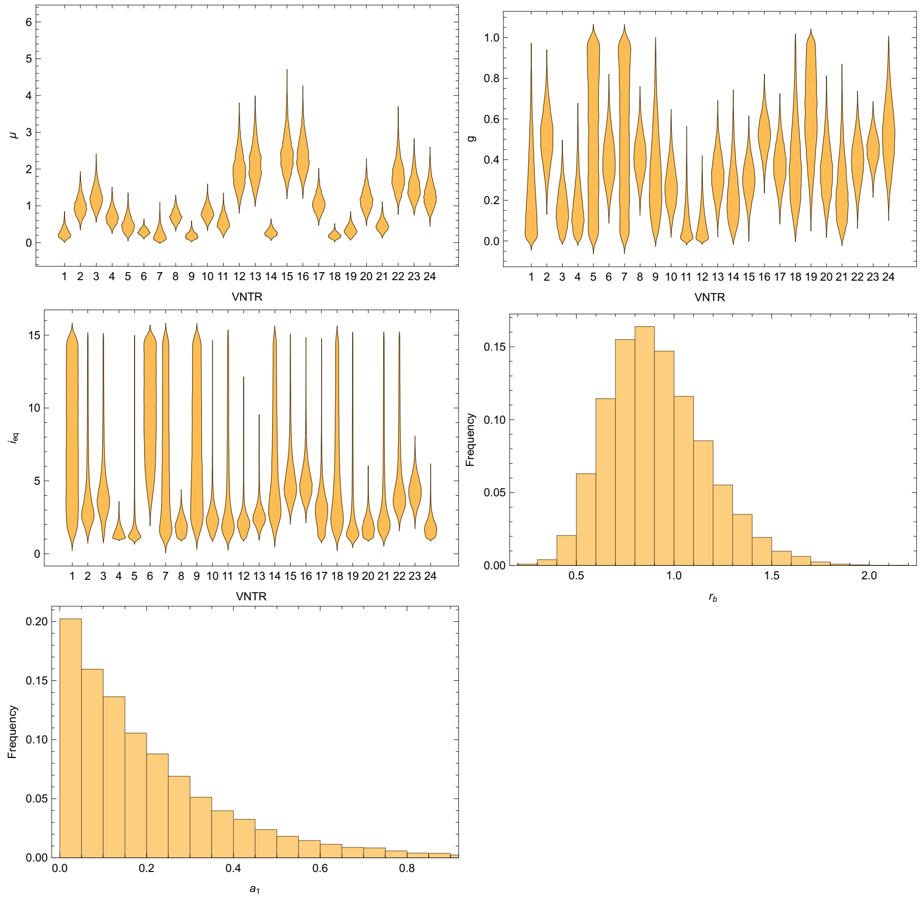
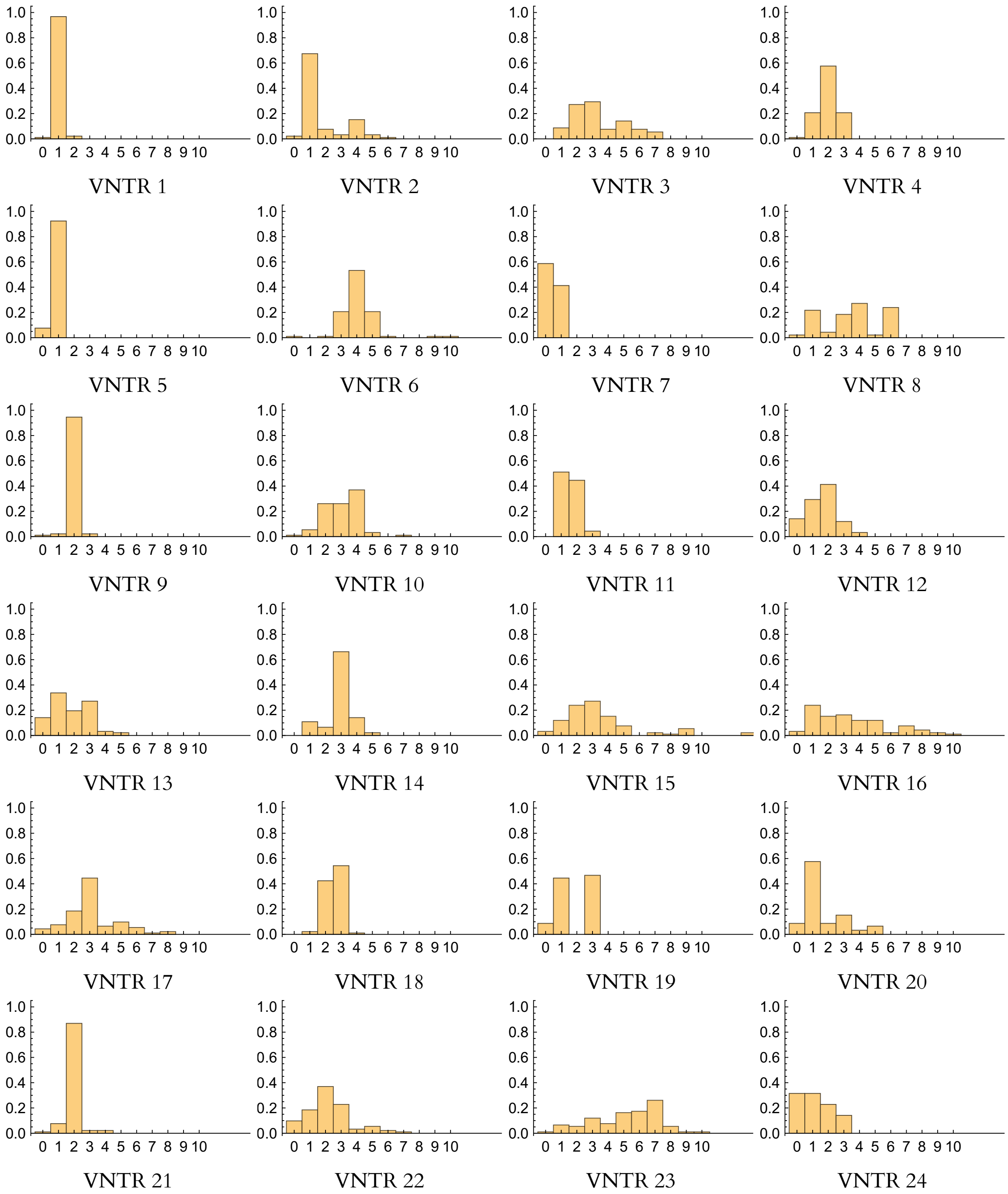


Figure 3.4: Posterior of the parameters $\mu^l, g^l, i^l_{\text{eq}}$ (different on loci l), and parameters r_b, a_0 (the same on loci l) of the Sainudiin model, found when inferring the phylogeny of the Comas data.



VNTR 1 VNTR 2 VNTR 3 VNTR 4

VNTR 5 VNTR 6 VNTR 7 VNTR 8

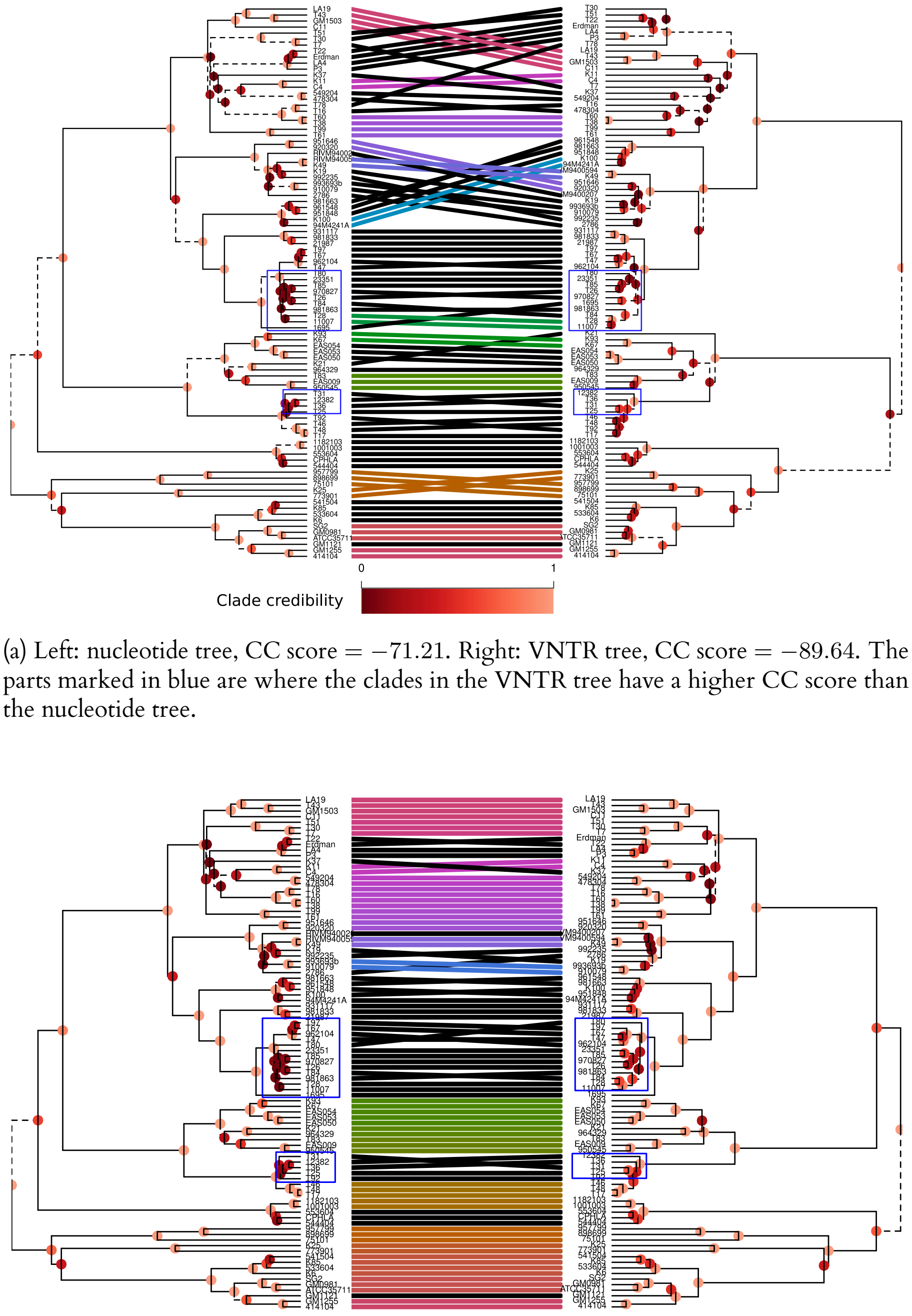
VNTR 9 VNTR 10 VNTR 11 VNTR 12

VNTR 13 VNTR 14 VNTR 15 VNTR 16

VNTR 17 VNTR 18 VNTR 19 VNTR 20

VNTR 21 VNTR 22 VNTR 23 VNTR 24

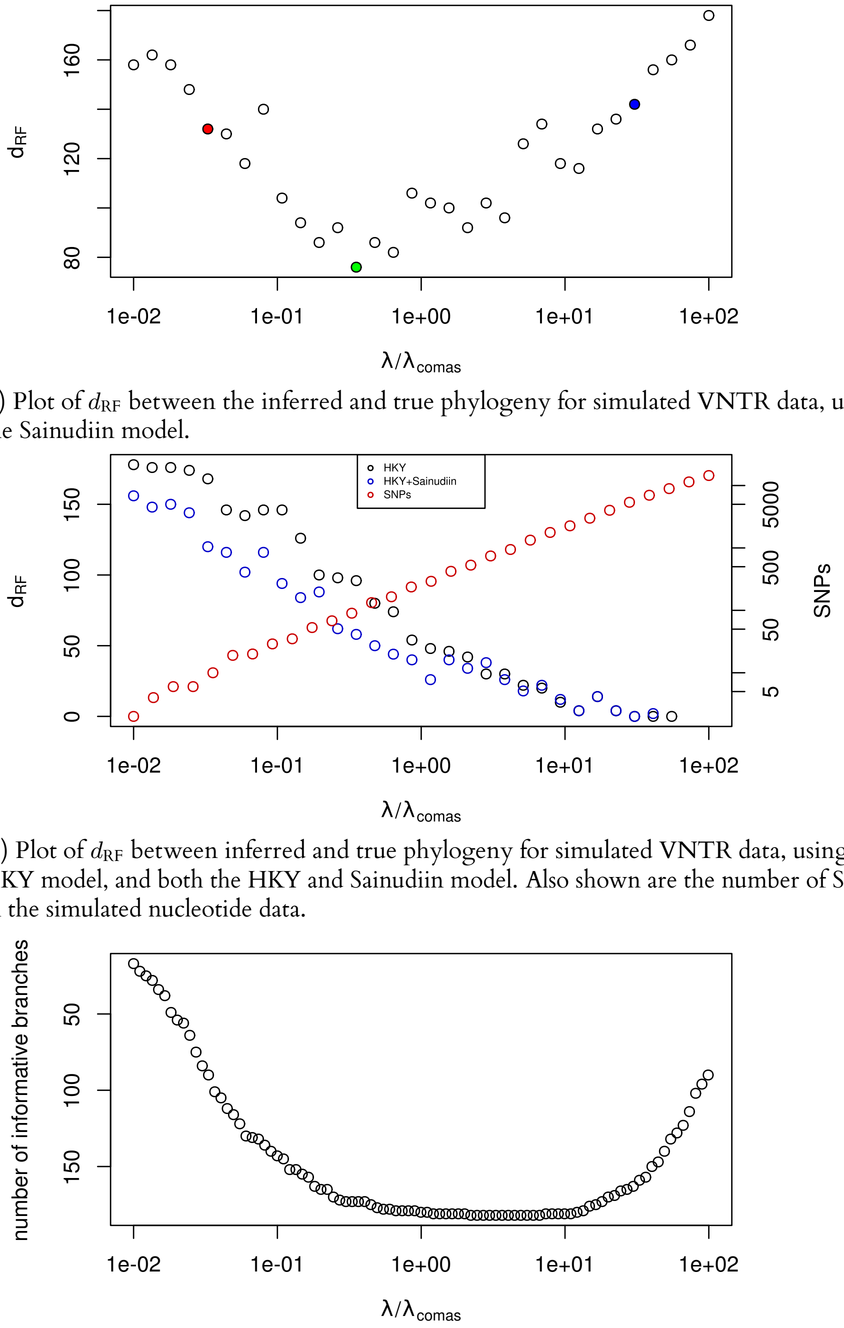
Figure 3.5: Distributions of the repeats $i - i_{\text{min}}$ on the 24 VNTR loci of the Comas dataset (with $i_{\text{min}} = 1$).



(a) Left: nucleotide tree, CC score = −71.21. Right: VNTR tree, CC score = −89.64. The parts marked in blue are where the clades in the VNTR tree have a higher CC score than the nucleotide tree.

(b) Left: nucleotide tree. Right: nucleotide and VNTR tree, CC score = −40.80. The parts marked in blue are where the clades in the VNTR and nucleotide tree have a higher CC score than the nucleotide tree.

Figure 3.6: Tanglegrams of the inferred phylogenies of the Comas dataset.

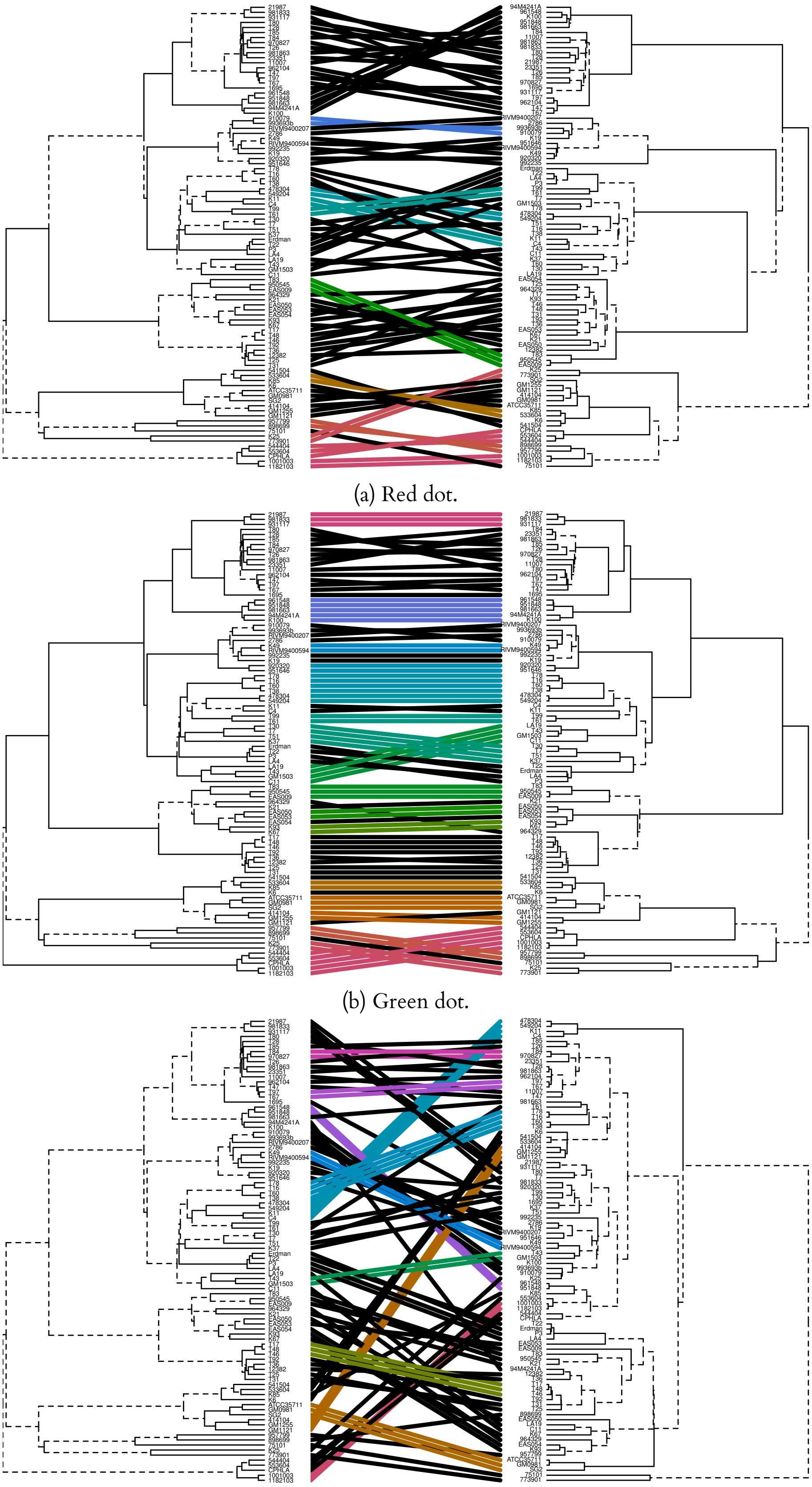


(a) Plot of d_{RF} between the inferred and true phylogeny for simulated VNTR data, using the Sainudiin model.

(b) Plot of d_{RF} between inferred and true phylogeny for simulated VNTR data, using the HKY model, and both the HKY and Sainudiin model. Also shown are the number of SNPs in the simulated nucleotide data.

(c) Plot of the the number of informative branches in the tree used for simulating VNTR data, defined by equation (3.11).

Figure 3.7: Results of simulated data, for different levels of mutational saturation $\lambda (= \mu t)$.

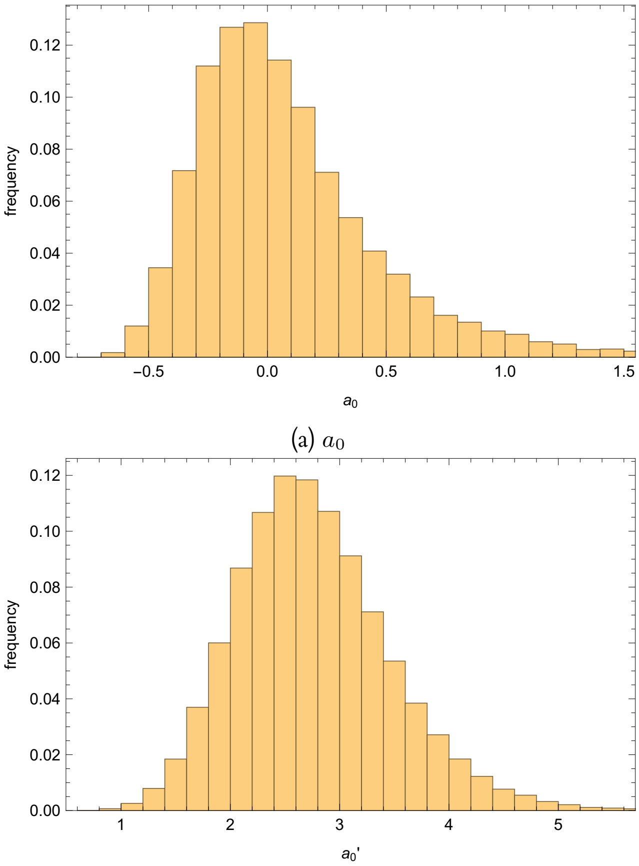


(a) Red dot.

(b) Green dot.

(c) Blue dot.

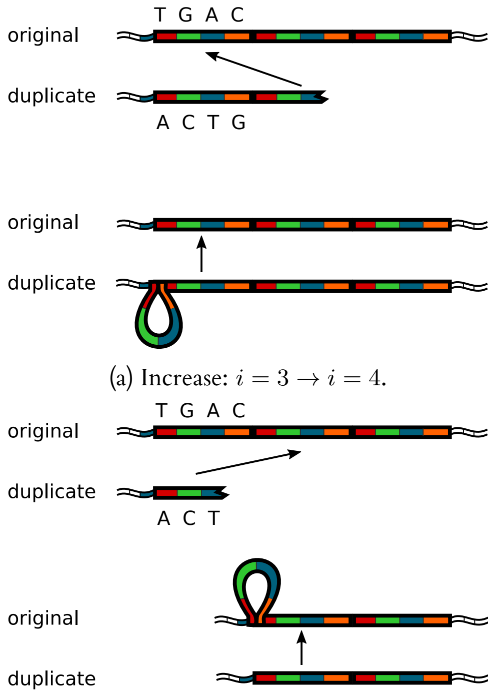
Figure 3.8: Inferred trees of simulated VNTR data, belonging to the red, green and blue dots of figure 3.7a. Left: the original phylogeny. Right: the inferred phylogeny.



(a) a_0

(b) a_0'

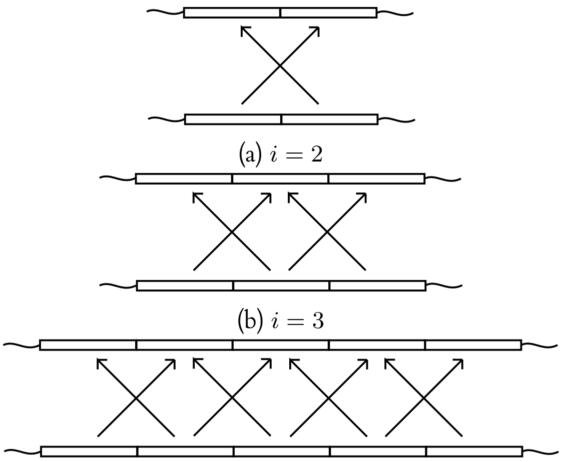
Figure 3.9: Posteriors of the parameters of the changed rate proportionality model equation (3.15), when inferring the phylogeny of the Comas data.



(a) Increase: $i = 3 \rightarrow i = 4$.

(b) Decrease: $i = 3 \rightarrow i = 2$.

Figure 3.10: Schematic of some of the possible ways state $i = 3$ of the VNTR can increase or decrease.



(a) $i = 2$

(b) $i = 3$

(c) $i = 5$

Figure 3.11: Schematic of possible misalignments of repeats for different states i of the VNTR during duplication. Each arrow depicts a possible way of misalignment.

— 4 —

Conclusion

Before starting this project, there was no capability of inferring the phylogeny based on VNTR data in BEAST2. By implementing the Sainudiin model in BEAST2 as an add-on, we extended its capabilities.

Some of the parameters originally used in this model, b_0, b_1 had no intuitive interpretation. In chapter 2 they were transformed into r_b, i_{eq}, which can be interpreted in an intuitive way as the magnitude of the bias and the focal point of the bias.

In experiment 3.1 and experiment 3.4 we compared the performances of the Sainudiin and the HKY model. From experiment 3.1 we saw that the phylogenies obtained from both models resemble each other.

In experiment 3.2, we showed that from all analyses of the VNTR data with different model settings, using homogeneous parameters on the loci, without any mutational bias, is sufficient for inferring the phylogeny. Concerning the parameters, only g could be quite accurately determined, while the precision of all obtained parameters was quite low.

In experiment 3.3, we combined the information that was present in the nucleotide and VNTR data. In case of the data from the Comas study, we were able to better infer the phylogeny in terms of consistency of the MCMC output for the combination of both models.

We showed in experiment 3.4 using simulated data, that for levels of saturation of that of the Comas data, the Sainudiin model can infer phylogenies that are about twice as far from the true phylogeny, compared to the HKY model. However, combining both models yields a phylogeny that is substantially closer to the true phylogeny, than solely using HKY.

This result together with the result of experiment 3.3 indicates that VNTR contains information not contained in nucleotides, at least for the case of the Comas data. This could imply that with the introduction of WGS for sequencing TB, it might still be necessary to also type VNTR, to be able to infer the phylogeny more accurately.

We also showed in this experiment 3.4 using simulated data, that the mutations can be both beneficial and detrimental when it comes to inferring the phylogeny: too little mutations, and there is not enough information present to say anything about the phylogeny, while too much mutations diminish the information about the phylogeny. Furthermore, we showed that the branch lengths in the tree play a role together with the mutation rate, in the overall capability of inferring the phylogeny.

In experiment 3.5, we showed by using a different model for the mutation rate proportionality, that the MCMC output suggests that the rate is directly proportional to the number of edges between the repeats.

Discussion

Even though we gave a possible explanation for the way that slippage can occur in experiment 3.5 for $i \geq 2$, it is still not clear how state $i = 1$ is able to mutate. Clearly, with only 1 repeat being present, there is no possible way of misalignment. Even more puzzling, is how it is possible that states $i = 0$ occur in the data, which are repeats of zero length. For our purposes, we threw away any samples that contained states $i = 0$, even though it is still possible to infer the phylogeny in those cases by setting $i_{\text{min}} = 0$. We saw no significant differences in the obtained phylogenies for the Comas data for the cases $i_{\text{min}} = 0, 1$. The reason for discarding samples with states $i = 0$ in our research, was that the model of Sainudiin was never intended to be able to explain repeats of zero length.

Even though we did not use samples which contained missing data for our research, BEAST2 has capabilities to handle such cases. Whenever the state of a locus of a sample is unknown, BEAST2 assumes that any of the possible states in the model could have occurred with equal probability on that locus. For the Sainudiin model, BEAST2 would thus assign equal probability to, for example, state $i = 1$ and $i = 15$, even though the latter is very rare in the data compared the former. This can be solved by implementing in BEAST2 the assumption that any missing state is distributed according to, for example, the stationary distribution of the model.

We recommend that the methods developed for this research be used, whenever accurate phylogenies are wanted for VNTR samples of TB. Concerning the computing power needed for these methods: during our research we achieved a performance of approximately 1.000.000 MCMC samples per 5 minutes, using about 100 isolates. Whenever inferring the phylogeny for a number of VNTR isolates that would exceed any available computing time or computing power, a solution might be to split the set of isolates into subsets, for which the phylogeny can be inferred separately. BEAST2 also has capabilities to perform computations in parallel when used in combination with the beagle[1] library, which can give a significant speed boost. For other cases where computational power is still an issue, it is possible to fall back on Minimum Spanning Tree and Neighbour Joining, even though these methods can only discriminate major lineages.

As was shown in figure 3.7a, for approximately 1/3 of the level of mutational saturation as that for the Comas data, the Sainudiin model is still quite capable of accurately inferring the phylogeny. This means that for other pathogens which have a mutation rate which is between 1/3 and 1 of the mutation rate of TB (approximately), we still expect to be able to accurately infer the phylogeny, when their isolates are sampled on the same (global) scale as the Comas data. In general, the ability of the Sainudiin model to infer the phylogeny for isolates of other pathogens depends the level of mutational saturation, or the variability in samples that the data has.

The added value of VNTR to nucleotides, depends on the amount of SNPs that the nucleotides capture. Whenever an amount of nucleotides comparable to that of the Comas data is sampled (approximately 66.000), we recommend that VNTR is also sampled in order to be able to more accurately determine the phylogeny. As more nucleotides are sampled, the added value of VNTR decreases.

[1] `https://github.com/beagle-dev/beagle-lib`

Bibliography

[Com+09] Iñaki Comas et al. “Genotyping of Genetically Monomorphic Bacteria: DNA Sequencing in Mycobacterium tuberculosis Highlights the Limitations of Current Methodologies”. In: *PLoS ONE* 4.11 (Nov. 2009), pp. 1–11. DOI: `10.1371/journal.pone.0007815`.

[DB15] Alexei J Drummond and Remco R Bouckaert. *Bayesian evolutionary analysis with BEAST*. Cambridge University Press, 2015.

[Ell04] Hans Ellegren. “Microsatellites: simple sequences with complex evolution”. In: *Nat Rev Genet* 5.6 (June 2004), pp. 435–445. ISSN: 1471-0056. DOI: `10.1038/nrg1348`.

[Fel73] Joseph Felsenstein. “Maximum Likelihood and Minimum-Steps Methods for Estimating Evolutionary Trees from Data on Discrete Characters”. In: *Systematic Biology* 22.3 (1973), pp. 240–249. DOI: `10.1093/sysbio/22.3.240`.

[Gal+10] James E. Galagan et al. “TB database 2010: Overview and update”. In: *Tuberculosis* 90.4 (2010), pp. 225–235. ISSN: 1472-9792. DOI: `10.1016/j.tube.2010.03.010`.

[Gal15] Tal Galili. “dendextend: an R package for visualizing, adjusting and comparing trees of hierarchical clustering”. In: *Bioinformatics* 31.22 (2015), pp. 3718–3720. DOI: `10.1093/bioinformatics/btv428`.

[GMM16] Sara Goodwin, John D McPherson and W Richard McCombie. “Coming of age: ten years of next-generation sequencing technologies”. In: *Nat Rev Genet* 17.6 (June 2016), pp. 333–351. ISSN: 1471-0056.

[Her+08] Ruth Hershberg et al. “High Functional Diversity in Mycobacterium tuberculosis Driven by Genetic Drift and Human Demography”. In: *PLoS Biol* 6.12 (Dec. 2008), pp. 1–14. DOI: `10.1371/journal.pbio.0060311`.

[HKY85] Masami Hasegawa, Hirohisa Kishino and Taka-aki Yano. “Dating of the human-ape splitting by a molecular clock of mitochondrial DNA”. In: *Journal of Molecular Evolution* 22.2 (1985), pp. 160–174. ISSN: 1432-1432. DOI: `10.1007/BF02101694`.

[HRW92] F.K. Hwang, D.S. Richards and P. Winter. *The Steiner Tree Problem*. Annals of Discrete Mathematics. Elsevier Science, 1992. ISBN: 9780080867939.

[Kös+12] Claudio U. Köser et al. "Importance of the Genetic Diversity within the Mycobacterium tuberculosis Complex for the Development of Novel Antibiotics and Diagnostic Tests of Drug Resistance". In: *Antimicrobial Agents and Chemotherapy* 56.12 (2012), pp. 6080–6087. DOI: `10.1128/AAC.01641-12`.

[Kru+98] Semyon Kruglyak et al. "Equilibrium distributions of microsatellite repeat length resulting from a balance between slippage events and point mutations". In: *Proceedings of the National Academy of Sciences* 95.18 (1998), pp. 10774–10778.

[RC13] C. Robert and G. Casella. *Monte Carlo Statistical Methods*. Springer Texts in Statistics. Springer New York, 2013. ISBN: 9781475730715.

[RF81] D.F. Robinson and L.R. Foulds. "Comparison of phylogenetic trees". In: *Mathematical Biosciences* 53.1-2 (1981), pp. 131–147. ISSN: 0025-5564. DOI: `10.1016/0025-5564(81)90043-2`.

[Sai+04] Raazesh Sainudiin et al. "Microsatellite Mutation Models". In: *Genetics* 168.1 (2004), pp. 383–395. DOI: `10.1534/genetics.103.022665`.

[Smi+09] Noel H Smith et al. "Myths and misconceptions: the origin and evolution of Mycobacterium tuberculosis". In: *Nat Rev Micro* 7.7 (July 2009), pp. 537–544. ISSN: 1740-1526. URL: `http://dx.doi.org/10.1038/nrmicro2165`.

[WD11] Chieh-Hsi Wu and Alexei J. Drummond. "Joint Inference of Microsatellite Mutation Models, Population History and Genealogies Using Transdimensional Markov Chain Monte Carlo". In: *Genetics* 188.1 (2011). Ed. by M. K. Uyenoyama, pp. 151–164. ISSN: 0016-6731. DOI: `10.1534/genetics.110.125260`.

[Whi+03] John C. Whittaker et al. "Likelihood-Based Estimation of Microsatellite Mutation Rates". In: *Genetics* 164.2 (2003), pp. 781–787. ISSN: 0016-6731.

[Wol16] Wolfram Research, Inc. *Mathematica*. Version 10.4.1.0. Champaign, Illinois, 2016.