Coalescent HMM analysis of great ape speciation
Anders Egerup Halager, 20063252

Master’s Thesis, Computer Science
December 2012
Advisor: Thomas Mailund
Abstract

Creating a generalized framework for generating continuous Markov chains allows us to represent a large number of different evolutionary scenarios without having to derive the math for each case. It also allows us to scale up to a number of states that would be near impossible to handle by hand. In this thesis I explain how the general framework functions and then test it on several different scenarios and with varying number of states, data and free parameters.
Contents

Abstract

1 Introduction

2 Modelling the coalescent process
   2.1 The coalescent process
      2.1.1 A colored Petri net model for pairwise genealogies
      2.1.2 Building coalescent CTMCs from the Petri net specification
   2.2 Constructing sequential Markov coalescent models
      2.2.1 Projecting runs of the CPN model onto pairs of genealogies
      2.2.2 Dealing with different demographic epochs
      2.2.3 State space statistics

3 Isolation model
   3.1 The model
   3.2 Simulated data
      3.2.1 Parameter estimation
      3.2.2 Number of states in HMM
      3.2.3 Estimation with varying data size
      3.2.4 Full estimates

4 Isolation-with-migration model
   4.1 The model
   4.2 Simulated data
   4.3 Parameter estimation
      4.3.1 Number of HMM states
   4.4 Model checking
      4.4.1 Akaike’s information criteria
      4.4.2 Real data

5 Varying coalescent rate over time
   5.1 Data
   5.2 Results
Chapter 1

Introduction

The cost of digitizing an individual’s full genetic material, his genome, has gone down dramatically for many years now. This means full data is available for multiple individuals of humans and all of our closely related species (chimpanzee, bonobo, gorilla and orangutan – the great apes).

All this data can potentially tell us more about our shared history with these animals than fossils alone or previous studies looking at single genes.

What we would like to estimate are things like when the different species diverged and the change in population sizes over time.

One method for analyzing full genome sequences is Sequential Markov Coalescent (SMC) and its inference method Coalescent hidden Markov models (CoalHMMs)

CoalHMMs have a major disadvantage in that they do not trivially generalize to more complex scenarios like bottlenecks in the population size or exchange of genes between divergent species (gene flow, or migration). By switching to continuous time Markov chains (CTMCs) [7] this problem is partly solved, since this mathematical model extends to the more complicated scenarios.

Constructing these CTMC models describing the complex demographies is done by hand, which is error prone and limits the complexity.

The three main contributions from this thesis are a) a method for generating CTMC models automatically from a simple colored Petri net (CPN) specification, b) tests that verify that said models can actually tell us something and c) an analysis of real great ape genome data utilising the models.

• I explain the framework used to generate arbitrary genealogies in chapter 2. I also show that the method grows very fast in complexity, when the number of species increase. This work is also the basis for a paper I co-authored [8].

• This method of analysis works well for basic simulated data. The first scenario is examined in chapter 3 – it is a simple isolation model, where two species separate cleanly. Next, in chapter 4, the model is extended with gene-flow after the species have diverged.
I also show that I can probably replicate the results from [6] by analysing simulated data with a similar model in chapter 5.

By varying the migration rate over time it should also be possible to approximate admixture events. The results are described in chapter 6.

Some of the chapters in this thesis cover information already published or in the pipeline, there is some similarity with their text and especially graphs. This includes the already mentioned Petri net article of course. There is also a paper [9] which should be published by this date that share a lot of data and graphs with chapter 3 and 4.

Finally this work has also been used for analysis of some great ape genomes in a paper that is yet to appear.
Chapter 2

Modelling the coalescent process

The general idea behind coalescent hidden Markov models is to approximate the coalescent process by a Markov model along a genomic alignment. Below I first present the coalescence process and then present a CPN model of the coalescence process over two neighboring nucleotides. This method was also released as a separate paper [8].

2.1 The coalescent process

The coalescent process [4] is a statistical model describing the genetic relationship between a sample of genes. The coalescent process assumes that \( k \) genes have been sampled in a population, and models how their ancestry (or “genealogy”) could be, providing probabilities to different scenarios of the genes ancestry from which a number of properties of the population can be inferred.

The coalescent process runs backwards in time, and in its simplest form a pair of genes can coalesce with a fixed rate. When two genes coalesce, it models the time where they last shared an ancestor (known as the most recent common ancestor, or \( \text{MRCA} \), of the two genes). After a pair of genes have coalesced, they are replaced by a gene representing their MRCA, and the process continues further back in time, now with \( k - 1 \) genes. The process is continued until there is only one gene left (\( k = 1 \)). A run of the coalescent process corresponds to a tree, where the shape of the tree is determined by the order of coalescence events and the branch lengths equal to the coalescence times – see Fig. 2.1 (a).

If we treat this process as a continuous time Markov chain, each tree can be assigned a probability, and by placing mutations on the tree we can compute the probability that a given tree gave rise to the observed genes. From this we can get the joint probability of the tree and the observed genes, and use this to make statistical inference. Since the true ancestry of genes is unknown, and in general unknowable, statistical inference based on the coalescence process involves integrating over all trees, either explicitly (for small \( k \)) or with statistical Monte
Figure 2.1. A coalescence tree and an ancestral recombination graph.
(a) A coalescence tree, where first genes 1 and 2 coalesce into their most recent common ancestor, \(\{1, 2\}\), then genes 3 and 4 coalesce into their most recent common ancestor \(\{3, 4\}\) and finally all genes coalesce into the grand most recent common ancestor. (b) An ancestral recombination graph of four sequences of length two. First genes 3 and 4 coalesce, where both their left and right nucleotide find an ancestor at the same time. Then gene 2 recombines, leading to independent genes for its left and right nucleotide. The right nucleotide of gene 2 coalesces with the ancestor of genes 3 and 4 while the left nucleotide of gene 2 coalesces with gene 1, before all genes find their most recent common ancestor. The left and right nucleotide in this example have different genealogies, with the left having topology \(((1, 2), (3, 4))\) and the right having topology \((1, (2, (3, 4)))\).

Carlo integration (for larger \(k\)).

Using a simple tree to represent the history of genes, is an inaccurate model since sex cells in species with two genders are constructed as “recombinations” of the genetic material inherited by each parent. In the coalescent process, this is modeled by adding a second type of event. Each gene can undergo recombination, in which case the gene is split in two at a random point, the left and right side of the recombination point. The process then continues with \(k + 1\) genes as the left and right part of the recombined gene is assumed to have independent ancestries.

A run of the coalescent process with recombination can no longer be represented as a tree but we must instead use a directed acyclic graph, known as the ancestral recombination graph or ARG, see Fig. 2.1 (b). Scanning from left to right along the sampled genes, at each point the ancestry of the genes will be a tree, but the trees can change whenever a recombination point is seen. The tree at each point is known as a local genealogy while the ARG is known as the global genealogy of the genes.

The state space of possible ARGs for a gene sample is generally intractable for all but the smallest samples [15], even for statistical integration, and to deal
with large sample sizes or long gene sequences approximations to the process is necessary. The approximation I work under is to assume that the relationship between genealogies satisfies the Markov property (meaning that the next state in a sequence can be predicted from the current state just as well as from the full history of states). Assuming the Markov property essentially means that we only need to model pairs of nucleotides rather than the full DNA sequence, since the probability of a sequence can be specified through all the pairwise probabilities [1,11].

### 2.1.1 A colored Petri net model for pairwise genealogies

While the coalescence process is difficult to make inference from, the rules for how the process generates genealogies are straightforward and can be expressed as a very simple colored Petri net. The way the coalescence process treats genes as independent items, with events that can affect one or two genes, maps straightforwardly to a CPN model where genes become tokens and coalescence and recombination events become transitions. Such a CPN model is shown in Fig. 2.2. The CPN model has a single place, containing the genes of the process, and two transitions modeling the two operations Coalescence and Recombination. The tokens on the single place consists of pairs – the left and right nucleotide of the genes – and each nucleotide will contain the set of original sampled genes. The initial marking consists of pairs \( \{i\}, \{i\} \) for genes \( i = 1, \ldots, k \).

A run of this CPN, producing the ARG in Fig. 2.1 (b), might look like this:

---

<table>
<thead>
<tr>
<th>State: ( 1'({1}, {1}) + 1'({2}, {2}) + 1'({3}, {3}) + 1'({4}, {4}) )</th>
</tr>
</thead>
<tbody>
<tr>
<td>Binding: ( \text{Coalescence; } 1'({3}, {3}) + 1'({4}, {4}) )</td>
</tr>
<tr>
<td>State: ( 1'({1}, {1}) + 1'({2}, {2}) + 1'({3, 4}, {3, 4}) )</td>
</tr>
<tr>
<td>Binding: ( \text{Recombination; } 1'({2}, {2}) )</td>
</tr>
<tr>
<td>State: ( 1'({1}, {1}) + 1'({2}, \emptyset) + 1'(\emptyset, {2}) + 1'({3, 4}, {3, 4}) )</td>
</tr>
<tr>
<td>Binding: ( \text{Coalescence; } 1'({3, 4}, {3, 4}) + 1'(\emptyset, {2}) )</td>
</tr>
<tr>
<td>State: ( 1'({1}, {1}) + 1'({2}, \emptyset) + 1'({3, 4}, {2, 3, 4}) )</td>
</tr>
<tr>
<td>Binding: ( \text{Coalescence; } 1'({1}, {1}) + 1'({2}, \emptyset) )</td>
</tr>
<tr>
<td>State: ( 1'({1, 2}, {1}) + 1'({3, 4}, {2, 3, 4}) )</td>
</tr>
<tr>
<td>Binding: ( \text{Coalescence; } 1'({1, 2}, {1}) + 1'({3, 4}, {2, 3, 4}) )</td>
</tr>
<tr>
<td>State: ( 1'({1, 2, 3, 4}, {1, 2, 3, 4}) )</td>
</tr>
</tbody>
</table>

When we have different populations or different species, the probability of coalescing genes in different populations/species is zero, and we cannot model genes in this simple way. To model this, we can annotate tokens with populations and only allow Coalescence and Recombination to affect genes within a single population, but instead add a new event that migrates a gene from one population to another, see Fig. 2.3.
Figure 2.2. CPN model of the basic two-nucleotide coalescence. This rather simple colored Petri net can construct all two-nucleotide coalescence runs for any number of samples in a single population. The set of genes in the process are represented as tokens on the single place, where each token contains a pair of sets of sampled genes. The pair represent the left and right nucleotide in the gene, and the sets the genes or most recent common ancestor of a set of genes. A coalescence event combines the left and right sets of the genes, while a recombination breaks up one gene into two: the left and right nucleotide of the original gene.

Figure 2.3. CPN model with migration. To model different populations, we annotate each token with the population it belongs to. Coalescence events are only possible between lineages in the same population. Recombination, as well, although this only involves a single lineage so the difference is only seen in the arc annotation. To allow lineages to move from one population to another, a new transition is added that moves one lineage from one population to another.
2.1.2 Building coalescent CTMCs from the Petri net specification

From the CPN specification we can build a state space capturing all possible ancestries of a sample. Our goal is to assign probabilities to all such ancestries. We do this by considering the process as a continuous time Markov chain (CTMC), and then building the complete state space graph of the system. This corresponds to a matrix of rates between states where the rate between states is given by the type of transition in the CPN.

In terms of CTMC theory, what we construct is the instantaneous transition matrix, usually denoted $Q$, and from this we can derive the probability of any run of the system. Obtaining a probabilistic model of the ancestries of a sample thus involves building the complete state space of the CPN model, translating this into a matrix of rates of transitions and considering this a CTMC rate matrix. For samples from a single population, we assign a fixed rate to transitions and recombinations (see Fig. 2.2), while for a scenario with multiple populations, we allow different coalescence rates for each population and different migration rates between different pairs of populations.

2.2 Constructing sequential Markov coalescent models

The computational efficiency of CoalHMMs stems from assuming that the probability distribution of genealogies along a genome alignment satisfies the Markov property, meaning that the probability of a local genealogy depends on its immediate neighbor, but not the more distant genealogies [1,11,12]. This way, the probability of a genealogy of the entire alignment can be specified from just the probability distribution of genealogies of two neighboring nucleotides [3,7].

Let $\Pr (G_L, G_R)$ denote the joint probability of genealogies, $G_L$ and $G_R$ of two nucleotides $L$ and $R$ (left and right). If this probability can be efficiently computed, then the probability of a genealogy over $L$ nucleotides, $\Pr (G_1, G_2, \ldots, G_L)$ can efficiently be computed as $\Pr (G_1, G_2, \ldots, G_L) = \Pr (G_1) \prod_{i=1}^{L-1} \Pr (G_{i+1} | G_i)$ where $\Pr (G_1) = \sum_g \Pr (G_1, g)$ and $\Pr (G_{i+1} | G_i) = \Pr (G_i, G_{i+1}) / \Pr (G_i)$.

The key idea in Mailund et al. [7], that is generalized here, was that these joint probabilities can be computed from a two-nucleotide CTMC. We can explicitly enumerate all possible states and state changes in the ancestry of two neighboring nucleotides, construct the corresponding CTMC, and obtain probabilities from this. Constructing the CTMC manually is feasible for small systems, as in Mailund et al. [7], but quickly becomes unmanageable. Below I show how the system can be constructed from a colored Petri net, and how the joint probability of a pair of genealogies can be algorithmically constructed from this.
2.2.1 Projecting runs of the CPN model onto pairs of genealogies

A run of the CTMC involves coalescence events, recombination events and migration events. Of these, only coalescence events, where two lineages find a MRCA, are observable in the genealogies. All other events are important for computing the probability of the genealogies, but only the times of MRCAs are directly observable as genealogies and all other events should be integrated out when the probabilities of genealogies are computed.

The time points where two lineages find their MRCA correspond to transitions in the CTMC state space where the system moves from one strongly connected component (SCC) to another since both migration and recombination events are reversible through a migration back or a coalescence of the two genes recombined, respectively. The genealogies of interest thus correspond to the paths in the SCC graph of the CTMC state space. Enumerating all paths in the SCC graph thus gives us all the genealogies to be considered, and there is a one-to-one
correspondence between pairs of genealogies and paths in the SCC graph. Fig. 2.4 shows a run of the CPN of a coalescent system. Here an ARG (from Fig. 2.1) is shown together with the states of the CPN that can produce this system, the SCC run of the system and the left and right genealogies of this run.

Paths in the SCC graph correspond to pairs of genealogies and will be the state transitions in the hidden Markov model we construct. Since our time is still continuous we have an infinite number of paths through the SCC graph, and correspondingly an infinite state space. We can fix this by discretizing time into a finite, fixed set of non-overlapping time intervals, \([\tau_0, \tau_1], [\tau_1, \tau_2], \ldots, [\tau_{n-1}, \tau_n]\) with \(\tau_0 = 0\) and \(\tau_n = \infty\). We obtain finite state spaces by only considering which state the system is in at the time points between these intervals \((\tau_1, \tau_2, \ldots, \tau_{n-1})\).

We combine the discretized time with the valid SCC runs by assigning mono-
cOMPONENTS_to each component. So, as an example, with three time intervals \([\tau_0 = 0, \tau_1], [\tau_1, \tau_2] \text{ and } [\tau_2, \tau_3 = \infty]\) and an SCC path with two components, \(c_1, c_2\), we would get the following three timed paths:

<table>
<thead>
<tr>
<th>Interval (\tau_1)</th>
<th>Interval (\tau_2)</th>
</tr>
</thead>
<tbody>
<tr>
<td>([\tau_0, \tau_1])</td>
<td>(c_1) ([\tau_1, \tau_2]) (c_1) ([\tau_2, \tau_3])</td>
</tr>
<tr>
<td>([\tau_0, \tau_1])</td>
<td>(c_1) ([\tau_1, \tau_2]) (c_2) ([\tau_2, \tau_3])</td>
</tr>
<tr>
<td>([\tau_0, \tau_1])</td>
<td>(c_2) ([\tau_1, \tau_2]) (c_2) ([\tau_2, \tau_3])</td>
</tr>
</tbody>
</table>

Notice that not all components need to be assigned a time point, and some can be assigned to several. This reflects that the system can move through several components within a single time interval and also stay in one component over several time intervals.

CTMC theory provides us with the mechanism for integrating over all paths leading from one state to another. If \(Q\) denotes the instantaneous rate matrix of the CTMC, then the probability of being in state \(s\) at time \(\tau_i\) and state \(t\) at time \(\tau_{i+1}\) is given by \(P^\tau_1 \tau_{i+1} - \tau_i\) where \(P^\tau_1 \tau_{i+1} - \tau_i = \exp (Q [\tau_{i+1} - \tau_i])\) (where \(\exp (M)\) denotes matrix exponentiation [13]). The probability of being in component \(c_i\) at time \(\tau_i\) and component \(c_j\) at time \(\tau_{i+1}\) is then computed by summing over all transitions from a state \(s \in c_i\) to a state \(t \in c_j\) in the time interval \([\tau_i, \tau_{i+1}]\):

\[
\sum_{s \in c_i} \sum_{t \in c_j} P^\tau_1 \tau_{i+1} - \tau_i \]

The probability of an entire SCC path assigned to time intervals is obtained by summing across all time intervals in this way (see Fig. 2.5), e.g. for the example above:

\[
\Pr ([\tau_0, \tau_1] \; [\tau_1, \tau_2] \; c_1 \; [\tau_2, \tau_3]) = \sum_{s \in c_1} \sum_{t \in c_1} P^\tau_1 \tau_3 \cdot P^\tau_2 \tau_{3} - \tau_1
\]

\[
\Pr ([\tau_0, \tau_1] \; [\tau_1, \tau_2] \; c_2 \; [\tau_2, \tau_3]) = \sum_{s \in c_1} \sum_{t \in c_2} P^\tau_1 \tau_3 \cdot P^\tau_2 \tau_{3} - \tau_1
\]

and

\[
\Pr ([\tau_0, \tau_1] \; [\tau_1, \tau_2] \; c_2 \; [\tau_2, \tau_3]) = \sum_{s \in c_2} \sum_{t \in c_2} P^\tau_1 \tau_3 \cdot P^\tau_2 \tau_{3} - \tau_1
\]
Figure 2.5. Computing path probabilities. When computing the probability of the timed path \([\tau_0, \tau_1] \ c_1 \ [\tau_1, \tau_2] \ c_2 \ [\tau_2, \tau_3] \ c_3 \ [\tau_3, \tau_4]\) we implicitly sum over all paths between the time interval breakpoints using CTMC transition probability matrices \(P_{\tau_{i+1}-\tau_i}\) and explicitly sum over states in the strongly connected components at the breakpoints \(c_i\).

(notice changes in subscripts) where we assume that the system always starts in a fixed state \(\iota\).

For the general case \([\tau_0, \tau_1] \ c_{i_1} \ [\tau_1, \tau_2] \ c_{i_2} \cdots [\tau_{n-2}, \tau_{n-1}] \ c_{i_{n-1}} \ [\tau_{n-1}, \tau_n]\) this becomes

\[
\sum_{s_1 \in c_{i_1}} \sum_{s_2 \in c_{i_2}} \cdots \sum_{s_{n-1} \in c_{i_{n-1}}} P_{\tau_{i_{i_1}}-s_1} \cdot P_{s_1,s_{i_2}} \cdots P_{s_{n-2},s_{n-1}}
\]

which is a sum of \(|c_{i_1}| \times |c_{i_2}| \times \cdots \times |c_{i_{n-1}}|\) terms, where each term is a product of \(n-1\) transition probabilities. To efficiently compute this for all paths, we can rewrite this to an expression that can can be evaluated inside-out using dynamic programming, like this:

\[
\sum_{s_1 \in c_{i_1}} P_{\tau_{i_{i_1}}-s_1} \left( \sum_{s_2 \in c_{i_2}} P_{s_2,s_{i_2}} \left( \cdots \left( \sum_{s_{n-1} \in c_{i_{n-1}}} P_{s_{n-1},s_{n-1}} \right) \right) \right)
\]

### 2.2.2 Dealing with different demographic epochs

When modeling the history of a set of genomes from different species, we need to consider different time periods of their history. An example would be a model of the ancestry of humans, chimpanzees and gorillas. Today these three different species cannot exchange genes, but as we go back in time we first enter a period where humans and chimpanzees share an ancestral species, allowing them to exchange genes, and further back in time all three species share an ancestor where they exchange genes.

To deal with this we can use different “epochs”. Each epoch corresponds to separate model in terms of transitions and transition rates, but all epochs for the same analysis can be embedded in the same (often large) state space, enabling us
to map states between them. For the human, chimpanzee and gorilla example, we would have three populations/species and one sample from each. So the type used for lineages would have three colors (e.g. H, C and G for human, chimpanzee and gorilla) and the type used for populations also three colors. The space of all possible states would be all states that the CPN could be in. The different epochs would consist of restrictions to this state space, and typically we would never enumerate the full state space but only the sub-state spaces reachable in the different epochs.

A simple human, chimpanzee and gorilla model could have three epochs, one where all three species are isolated, one where humans and chimpanzees have found a common ancestor and one where all three species have found a common ancestor. This model will not allow migrations in any epochs. The first epoch will have each species in its own population, the second epoch will have humans and chimpanzees in the same population, and the third epoch would have all three species in the same population.

We construct this model by first constructing the state space of the first epoch, where the populations are H, C and G. We then take all reachable states in this system and map H and C tokens to the same population, e.g. H, so tokens are mapped \( (p, (l, r)) \mapsto (H, (l, r)) \) whenever \( p \) is H or C. For the second epoch, we compute the state space of all states reachable from these mapped states (but not states from the first epoch where tokens can be in population C). For the third epoch we repeat this, but now mapping G populations to H as well.

When computing the probability of paths in the system, we add this projection of states as well. If the time point \( \tau_i \) is between two different epochs, we use a matrix \( P^{n_i} \cdot I_i \) instead of \( P^{n_i} \) where \( I_i \) is a projection matrix mapping states from the epoch before \( \tau_i \) to the epoch after \( \tau_i \). For the transition between the first and second epoch in the human, chimpanzee and gorilla example, this projection matrix would have a 1 in all entries where the states are equal exact for all C populations being set to H and 0 in all other entries, and for the projection from the second to the third epoch, the projection matrix would have a 1 in entries where the states are equal except that now G populations are set to H as well. The projection onto left and right genealogies, and the sums used for computing the probabilities of strongly connected components paths is not changed otherwise.

### 2.2.3 State space statistics

I constructed the state space and HMM transition matrix for a number of different configurations, varying the number of populations from one to three and varying the number of chromosomes from one to four. With one population there is a single time epoch, with two populations there are two epochs, one before and after the populations merge, and with three populations there are three time epochs: the first before any populations merge, the second after the first and second population merge, and the last when all three populations have merged.
Table 2.1. Summaries of the state space sizes, SCCs and construction time for both the state space and the hidden Markov model transition matrix. Configurations $n = i, j, k$ should be read as population one containing $i$ chromosomes, population two containing $j$ chromosomes and population three containing $k$ chromosomes. Construction time is measured in seconds and – indicates that the computation was terminated before finishing.

Table 2.1 shows the size of the state spaces in the various configurations and epochs and the time it takes to construct the HMM transition matrix. The HMM construction time is split in three components: 1) the time it takes to construct the CTMC (i.e. build the state space of the CPN and translate it into a rate matrix), 2) pre-processing time for the HMM construct, involving building the SCC graph and assigning all possible SCC paths to time intervals, and 3) the time it takes to construct the actual transition matrix, involving exponentiating rate matrices and summing over SCC paths. Of these three, the first two needs only be computed once for a given model, while the third needs to be recomputed whenever the parameters of the HMM changes, and must potentially be computed hundreds of times in a numerical optimization of the HMM likelihood.

The most time consuming part of constructing the HMM is clearly not constructing the state space of the model but rather the alignment of the SCC graph onto time intervals for constructing the HMM states and the exponentiation of rate matrices for computing transition probabilities. The configurations in Table 2.1 where the construction time is missing were terminated after hours of run-time indicating a very steep exponential growth in running time as the size of the system grows.
Chapter 3

Isolation model

Given the CTMC generating framework described in the previous chapter we can generate several models of evolutionary ancestries. We will start by exploring a simple isolation model prove that it can produce reasonable results before moving on to a more complicated model in chapter 4.

3.1 The model

We want to model a simple speciation event, with two separate species alive today and some common ancestor lineage back in time.

There are up to seven parameters in the model. The split time $\tau$ and then a different coalescence rate and recombination for each of the two present day species($(C_1, R_1), (C_2, R_2)$) and for the ancestor species($(C_A, R_A)$).

![Diagram](image-url)

**Figure 3.1. The model.** An illustration of the model used in this chapter, with all possible parameters to the model indicated.
3.2 Simulated data

In order to make some experiments we can simulate ancestral recombination graphs with the CoaSim tool [10] which can then be fed into the Bio++ suite [2] to simulate alignments with the Jukes-Cantor substitution model.

All of the simulations in this section will follow the same overall structure as our isolation model. We will simulate two individuals of separate species with some divergence time ($\tau$), a fixed coalescence rate ($C_1 = C_2 = C_A$) for the whole tree and similarly a fixed recombination rate ($R_1 = R_2 = R_A$).

In order to test how well we can estimate parameters I generate 10 independent 1Mbp segments and analyze them jointly. For all simulations the coalescence rate is $C = 2500$ – corresponding to an effective population size of $N_e = 10000$ assuming a substitution rate of $\mu = 10^{-9}$ substitutions per year and 20 years per generation – and the recombination rate is $R = 0.4$ – corresponding to 0.8 cM/Mb with the assumed mutation rate and generation time.

The first thing to check is whether or not the model has a maximum likelihood at the correct parameter values. We do this by plotting the likelihood curve for one parameter at a time, while fixing the other parameters at the true values. Figure 3.2 shows the likelihood curves for $C$, $R$ and $\tau$ and the maximum generally falls very close to the simulated values, except for $R$. After running 500 simulations, the ratio between the simulated $R$ and the $R$ that gives the maximum likelihood is off by a factor of 1.76 on average, which may be a difference in units between CoaSim, BppSeqGen and CoalHMM or maybe a bug in one of the three pieces of code. Assuming that it is indeed a minor problem somewhere I have simply scaled up the $R$ given to CoaSim in the next set of graphs so that we can see the top of the likelihood curve.

The likelihood surface for pairs of parameters can be seen in figure 3.3. It would have been nice if all parameters were independent of each other, but there seems to be some interdependence between $T$ and $C$.

Other than $R$ being off by the mentioned factor it seems the parameters have a maximum right around the simulated values – it would be reasonable to expect that we can also optimize multiple parameters at the same time with a standard multidimensional numerical optimizer.

3.2.1 Parameter estimation

To do the actual maximum likelihood estimation I used the function provided by SciPy [5] in scipy.optimize called fmin (an implementation of the Nelder-Mead simplex algorithm) while the likelihoods are calculated by HMMLib [14], although any HMM library ought to give the same result within some numerical precision.
Figure 3.2. Likelihood curves for individual parameters. These graphs show the likelihood for a single parameter each – the other parameters are fixed at the value used in the simulation. Each parameter likelihood is shown for two different simulated datasets. The vertical red lines indicate the true value.
Figure 3.3. Likelihood curves for pairs of parameters. Each sub-figure shows the likelihood surface for a pair of parameters, the third parameter is kept at the simulated value. Each set of parameters is plotted for two different simulated datasets. The coordinate representing the simulated parameters are marked at the center of the plots.
Figure 3.4. Coalescence rate estimates. Box-plots showing the accuracy of estimated coalescence rates as a function of the number of time intervals used in the model. The simulated value is indicated by a horizontal red line.

3.2.2 Number of states in HMM

When constructing a model we can adjust the number of states used in the hidden Markov model. The number of states will affect how we estimate our parameters, Figure 3.4 shows the estimation accuracy of the coalescence rate. With only 5 intervals there are a lot of outliers, but with 10 and 15 intervals the parameter seems to be well recovered. Figure 3.5 shows the accuracy of the recombination rate estimates. There are some extreme outliers with 5 states (values over 30), so I have limited the view to a narrower window. This rate seems to be consistently under-estimated, so adding states won’t fix what we saw when looking at $R$ in isolation. Figure 3.6 shows the estimation accuracy of the split time estimate. Again I’ve had to discard some extreme outliers for the 5 interval dataset, but we recover the value quite well.
Figure 3.5. Recombination rate estimates. Box-plots showing the accuracy of estimated recombination rates as a function of the number of time intervals used in the model. The simulated value is indicated by a horizontal red line.
Figure 3.6. Split time estimates. Box-plots showing the accuracy of estimated split time as a function of the number of time intervals used in the model. The simulated value is indicated by a horizontal red line.
3.2.3 Estimation with varying data size

We would expect that the variation in estimates drops as we increase the data size. To explore this, I created 128 data sets with lengths 1, 10 and 20 Mbp. Still the same parameters, $C = 2500$, $R = 0.4$ and $\tau = 1\text{Myr}$. Figure 3.7 shows the result. Going from 1 to 10Mbp gives a large improvement, while 20Mbp doesn’t seem to add much further. The reduced variance make the bias in the recombination estimates clear.

![Figure 3.7](image)

**Figure 3.7.** Estimates as a function of data size. Box-plots showing the variation in 128 datasets of size 1, 10 or 20Mbp. Going to 10Mbp improves estimates a lot, 20Mbp doesn’t improve much further.
3.2.4 Full estimates

If we try to estimate all three parameters on 128 independent sets of 10x1Mbp the result will look like figure 3.8. It seems that the divergence time and coalescent rate are estimated very consistently, while the recombination rate seems to be consistently underestimated, which is consistent with what we saw when looking at the parameters individually and again the ratio of the simulated value to the estimated values is about 1.75.

![Histogram of Coalescence rate](image1)

(a) Estimate for $C$. All close to the true value.

![Histogram of Recombination rate](image2)

(b) Estimate for $R$. The correct value is not even in sight, but the results seem to be fairly consistently wrong.

![Histogram of Divergence](image3)

(c) Estimate for $T$. Close to the true value.

Figure 3.8. Estimates for all parameters. Each sub-figure shows a histogram of all the estimates for that particular parameter. The true value is shown in the middle of the plot as a vertical red line.
Chapter 4

Isolation-with-migration model

4.1 The model

We can expand the isolation model from the previous chapter to include a phase with migration between the two species after the split.

Like with the isolation model we assume that $C$ and $R$ don’t vary throughout the tree. This means we have 5 parameters in all – $C$ and $R$ as before, but also times marking the beginning and end of the migration period ($\tau_1, \tau_2$) and finally the rate of mutation, $M$.

![Figure 4.1. The model. An illustration of the model used in this chapter, with the new parameters indicated.](image)

4.2 Simulated data

We can use the same tools to simulate data for this model.

Like with the isolation model we will start by making sure the likelihoods look sensible for both one parameter at a time and for pairs of parameters. We plotted likelihood curves for single parameters (see Figure 4.2) and for all pairs of parameters (see Figure 4.3).
The maximum likelihood is generally close to the simulated values, but we do see linearity in some of the pairs. One of them ($C$ and $\tau_2$) we already saw with the isolation model where it didn’t seem to hurt our ability to estimate. The other noticeable linearity is between $M$ and $\tau_1$ which is somewhat expected as a short period with high migration would produce results similar to a long period with lower migration. It might be possible to re-parameterize the model so that the parameters become decoupled, but I have not looked into the details of this.

### 4.3 Parameter estimation

Like with the isolation model each estimate is made on 10 independent 1 Mbp segments that are analyzed jointly. Simulations still use a coalescence rate of $C = 2,500$, a substitution rate of $\mu = 10^{-9}$ substitution per year, 20 years per generation and a recombination rate of $R = 0.4$. We simulated 10 independent data sets for each combination of parameters $\tau_1 \in \{0.00025, 0.00050\}$ (250 and 500 thousand years ago), $\tau_2 \in \{0.001, 0.002\}$ (1 and 2 million years ago), and $M \in \{62.5, 125.0, 250.0\}$.

#### 4.3.1 Number of HMM states

First we would like to figure out if 10 states is still enough now that we have introduced 2 extra parameters to tune. I have estimated parameters with 5, 10, 15, and 20 time intervals in the gene-flow period (from $\tau_1$ to $\tau_2$) and in the ancestral population (above $\tau_2$).

Figure 4.4 shows the estimation accuracy of the coalescence rate. For all configurations, the parameter seems to be well recovered. Figure 4.5 show the estimation accuracy of the recombination rate. For all configurations, this rate is under-estimated. This is consistent with the bias we have seen all along.

Figure 4.6 shows the estimation accuracy of the time where gene-flow stops completely ($\tau_1$ on the left) and when the ancestral population split in two ($\tau_2$ on the right). The latter is generally well recovered, while the former is recovered but with a larger uncertainty and possibly a slight up-wards bias.

Figure 4.7 shows the estimation accuracy of the migration rate. While the median estimate seems to recover the true value the distribution of estimates has a very wide right-tail as seen on Figure 4.8.

For most parameter estimates, the number of HMM states does not seem to have a large impact on the estimation accuracy, but the variance in estimates of migration rates is large for 5 time intervals and is reduced when the number of states is increased. It is acceptable with 10 time intervals, however, and since the HMM algorithms are quadratic in the number of states, we use 10 time intervals in all analyses unless otherwise stated.

24
Figure 4.2. Likelihood curves for individual parameters. Each sub-figure shows the likelihood for a single parameter, where all other parameters are kept at their simulated value. Each parameter likelihood is shown for two different simulated datasets. The true value is shown in the middle of the plot as a vertical red line.
Figure 4.3. Likelihood curves for pairs of parameters. Each sub-figure shows the likelihood surface for pair parameters, where all other parameters are kept at their simulated value. Each likelihood is shown for two different simulated datasets. The true values are shown in the center of the plot as a circle.
**Figure 4.4. Coalescence rate estimates.** Box-plots showing the accuracy of estimated coalescence rates as a function of the number of time intervals used in the CoalHMM. The vertical red lines indicate the true value.

**Figure 4.5. Recombination rate estimates.** Box-plots showing the accuracy of estimated recombination rates as a function of the number of time intervals used in the CoalHMM. The vertical red lines indicate the true value.
Figure 4.6. Split time estimates. Box-plots showing the accuracy of estimated split time as a function of the true split times and the number of time intervals used in the CoalHMM. Plots on the left shows the estimates of the time when gene-flow subsides ($\tau_1$) while plots on the right shows the estimates of the time when the ancestral population split in two ($\tau_2$). The vertical red lines indicate the true value.
Figure 4.7. Migration rate estimates. Box-plots showing the accuracy of estimated migration rates as a function of the true migration rates and the number of time intervals used in the CoalHMM. The vertical red lines indicate the true value.
Figure 4.8. Distribution of migration rate estimates. Ordered estimates of migration rates for the three different simulated rates and for the different configurations of time intervals. The distribution of estimates show a very wide right-tail, a problem that seems to be worse for few time intervals and low migration rates.
Increasing the number of intervals used (and in turn the states in the HMM) reduces the variance in the estimates and since the forward algorithm is quadratic in the number of states I have chosen 10 time intervals per epoch, since it seems to be enough to seriously reduce the variance.

### 4.4 Model checking

It would be nice to say whether or not a piece of data is likely to have migration – ideally we would want to do parameter estimation with the isolation model and use it as a null hypothesis that we might be able to reject with our isolation-with-migration estimates.

To get a feeling for how this could work, I simulated data with and without migration and then performed maximum likelihood estimation using both models. Figure 4.9 and Figure 4.10 show the maximum likelihood in each step of the numerical optimization for data simulated under an isolation model and an isolation-with-migration model, respectively. The value plotted is all the log likelihoods evaluated by the optimizer which gives it a very jagged look.

The optimization requires many more steps for the IM model since there are two more parameters, but at the end it ends with roughly the same value as the I model when that is what was simulated under. When the data is simulated with migration the IM model usually also ends at a much higher likelihood.
Figure 4.9. Log likelihoods during the optimization for data simulated under a pure isolation model. The plot shows the log-likelihood at each step in the numerical optimization. Blue lines correspond to maximum likelihood optimization under the isolation model, while red lines correspond to optimization with the IM model.
Figure 4.10. Log likelihoods during the optimization for data simulated under an isolation-with-migration model. The plot shows the log-likelihood at each step in the numerical optimization. Blue lines correspond to maximum likelihood optimization under the isolation model, while red lines correspond to optimization with the IM model.
4.4.1 Akaike’s information criteria

The obvious way to choose which model fits best is of course a typical likelihood ratio test, but the isolation and isolation-with-migration models aren’t nested, so doing a likelihood ratio test is strictly speaking wrong. While it is possible to get something useful I instead focus on Akaike’s information criteria (AIC) which penalizes the number of parameters used in fitting models but does not require nested models. It is defined simply as $AIC = 2k - 2\ln(L)$, where $k$ is the number of free parameters and $L$ is the likelihood.

For AIC, the model with the smallest score should be preferred, so we looked at the AIC for the isolation model minus the AIC for the isolation-with-migration model. When this is positive, the IM model is preferred while when it is negative the I model is preferred. Figure 4.11 shows these results for data simulated either under the pure isolation model with varying $\tau$ (250, 500, 1000 and 2000 kya) or under the isolation-with-migration model with varying $\tau_1$ (250 kya and 500 kya), $\tau_2$ (1 mya and 2 mya), and migration rate (62.5, 125, and 250). We mostly see values below zero when data was simulated under the isolation model and above zero when data was simulated under the isolation-with-migration model, so AIC chooses the right model a large part of the time.

When simulating under the isolation model, the AIC strongly prefers the isolation model when the split times aren’t too small, and even then it is still in favour of the isolation model.

<table>
<thead>
<tr>
<th>$\tau$ (kya)</th>
<th>I preferred</th>
<th>IM preferred</th>
</tr>
</thead>
<tbody>
<tr>
<td>250</td>
<td>74</td>
<td>26</td>
</tr>
<tr>
<td>500</td>
<td>77</td>
<td>23</td>
</tr>
<tr>
<td>1000</td>
<td>93</td>
<td>7</td>
</tr>
<tr>
<td>2000</td>
<td>97</td>
<td>3</td>
</tr>
</tbody>
</table>

When data is simulated under the isolation-with-migration model, the AIC measure once again prefers the correct model. The bigger distance between $\tau_1$ and $\tau_2$ and the larger the migration rate, the more heavily favored the isolation-with-migration model gets. Which is consistent with both things meaning there are more actual migration event in the data and by extension a stronger signal for the model to pick up on.
Figure 4.11. AIC tests for migration. The plots show AIC for the isolation model minus AIC for the isolation-with-migration model. Values above zero means that the migration model is preferred. In (a) values are shown for simulations under the isolation model, with varying $\tau$, while in (b) values are shown for the simulation model with $\tau_1$ varying between 250 kya and 500 kya (assuming $\mu = 10^{-9}$), $\tau_2$ varying between 1 mya and 2 mya, and $M$ varying between 62.5, 125 and 250.
4.4.2 Real data

Now that we have established the AIC criteria as a way to distinguish between which of the models match the data better, we can investigate whether or not there is migration in a real data set. The data I look at is a full genome alignment of two gorillas. Since the gorillas are very closely related we would expect to see some migration, unless they were completely separated physically by something like a river.

In order to decide I divided the data into chunks of about 10Mbp each. On each chunk both models are used to optimize all the free parameters. Figure 4.12 shows the difference between the two AICs calculated from the final likelihood in the optimization. Larger values mean that the isolation-with-migration model is preferred.

Outside of chromosome X it seems like there is plenty of evidence for migration in this data. So what does the isolation-with-migration model tell us about these two gorillas? I will present a few graphs here but there are many ways to plot that will tell us different things, and many variations that could be run that might give extra information (filtering the data, using bigger chunks, etc.) In figure 4.13 we can see where the time estimates land. It seems like the start of the migration is placed very early – the first three quartiles below 90,000 years, and the median at about 25,000 years. But there is a hump on the right side which probably represents the chunks with the least evidence for migration. The spread is a lot larger when looking at the coalescence time, but very roughly 300,000 years ago. These times are under the usual assumptions about the mutation rate and generation time ($\mu = 10^{-9}, g = 20$).
Figure 4.12. AIC. The plot shows the difference between the calculated AIC values for an isolation model and an isolation-with-migration model. The AICs are calculated based on the final likelihood after estimating all parameters in each of the models.

Figure 4.13. Estimates. The plot shows a density plot of the estimated times from all chromosomes. $T_1$ is when the migration starts and $T_2$ is when the two individuals diverged.
Chapter 5

Varying coalescent rate over time

In their paper from 2011 Li & Durbin [6] create a model (PSMC) that can estimate the population size over time using a complete diploid gene (two samples from the same species) with a high resolution.

Setting up my model to represent a similar situation is straightforward. It is just a pair of individuals from the same population, with multiple epochs where the only change from epoch to epoch is the coalescent rate. If the model was supposed to match exactly what Li & Durbin did in their PSMC model we would need a lot of epochs each covering an exponentially longer time period. Instead I’ve tested a simpler version with just three epochs, from 0 to 100 kyr, from 100 kyr to 200 kyr and finally from 200 kyr to infinity.

If this simplified model produces reasonable results it might be possible to recreate the full PSMC model with sufficient data and computation time.

5.1 Data

For these experiments I used the same simulation pipeline of CoaSim and Bio++ that was used for the isolation and isolation-with-migration models.

The data is simulated with three periods, with splits at 100 kyr and 200 kyr.

The simulations work with three different population sizes – low, normal and high – and they are set to \( N = \{2500, 10000, 40000\} \), corresponding to coalescence rates of \( C = \{625, 2500, 10000\} \).

All the simulations assume a substitution rate of \( \mu = 10^{-9} \) substitutions per year, a 20 year generation time and a recombination rate of \( R = 0.4 \) – corresponding to 0.8 cM/Mb with the assumed mutation rate and generation time.

For each of the 9 ways we can combine three population sizes I generated datasets of 1, 10 and 50 Mbp. I only show a subset of the results as all 27 would take up too much space.
5.2 Results

The first thing to look at is the likelihood curves. This is done by varying over one parameter at a time and running the forward algorithm on the HMM to get a likelihood in that point.

What we would hope to see is that all the curves have a maximum at roughly the correct value and that the curve gets less flat as more data is added. If both things happen we should be able to estimate all the parameters, as seen with the isolation and isolation-with-migration model in previous chapters.

One of our goals is met here as it looks like extra data gives a sharper curve for all datasets. This mean we that if we are having problems getting good estimates we simply need to increase the amount of data going into each optimization process. With some network communication this can be done very fast for even a full genome, by having a master process that instead of evaluating the likelihood at each step of the optimization farms the work out to a bunch of slave processes – this way the optimizer is working on the full dataset at the same time.

The second goal was to have maxima reasonably close to the correct values. That is what we see for most of the datasets, but in hi periods (periods with a larger population size, meaning a low coalescence rate) we sometimes see a very low value. A possible explanation for this is that there simply isn’t enough data even with 50Mbp. In figure 5.2 I have plotted the proportion of sites that have yet to coalesce at at given time for a few of the different setups. It shows quite clearly that in a period of a 100 kyr with a population of 40000 there just isn’t a whole lot of coalescence events. But in the last infinite time period we can still find the correct value since we have enough time. So if the coalescence rate is too low we will either need a lot of data or a larger time frame to accurately estimate the rate.

Since the previous experiments seem to suggest that we can estimate all three rates at the same time, I tried doing exactly that. I estimated on 15 instances of each combination described (only the ones with 50Mbps).

In figure 5.3 there is a plot for each of the epochs/variables. Within the individual plots I have grouped the data by what the correct value is. So each box represents 135 estimates (eg. in C1 it might be lo-lo-lo, lo-lo-no, ..., lo-hi-hi times 15 instances from each). Despite the fact that we know some of these don’t hit very well individually, the overall picture is good – the estimates are close enough to the correct values that I will say that the model can recover changes in coalescence rate.

5.3 Future work

In order to replicate the results from Li & Durbin we would have to increase the number of epochs a lot, but in the current implementation that would cause the
Figure 5.1. Likelihoods curves. The plots show the likelihood around the correct value for each of the epochs using either 1, 10 or 50Mbp. As the amount of data increases, the maximum likelihood moves closer to the true value and the curves span a larger range of likelihoods, meaning our confidence in the result grows.
Figure 5.2. Coalescence. The plots show how large a part of the simulated genome that is still two separate species at a given time. They show that after two periods with a large population size, only around 10% of sites have coalesced – so there is not always a lot of signal in the data.
Figure 5.3. Estimates. The plots shows boxplots of the estimates in each epoch, grouped only by what the correct value is in the plotted epoch. Note that low, normal and high are population sizes, which means the coalescence is the other way around.
number of states in the HMM to increase greatly. It should be possible to avoid this since each of these epochs have identical states, only the rates are different. I have not looked further in to this.
Chapter 6

Varying migration rate over time

Similarly to the previous chapter it is also possible to vary the migration rate over time. This serves as an approximation to admixture events, which is basically relatively short periods of time with relatively high migration rates. Admixture events could be seen for instance if two closely related species are separated by a river which dries out for a time. A real-world example is the different subspecies of orangutan that live on Sumatra and Borneo – today there are no orangutans crossing between the islands, but there might have been during some of the previous ice ages. Admixture events can be very short, so the model will need to either have a lot of epochs or to let the times be estimated as well.

The model used in this chapter is simpler as it only goes for three different migration rates, which would still be able to tell us something about the history of two species. If it works there I see no reason a model with higher resolution wouldn’t work given enough data.

6.1 Data

Once again I used the same simulation pipeline of CoaSim and Bio++ to generate genetic sequences.

The data is simulated with three periods of migration, starting at 500 kyr and uniformly distributed up to 4000 kyr where the two species merge.

The simulations work with three different migration rates – low, normal and high – and they are set to $M = \{62.5, 125, 500\}$.

All the simulations assume a substitution rate of $\mu = 10^{-9}$ substitutions per year, a 20 year generation time, a coalescence rate $C = 2500$ and a recombination rate of $R = 0.4$ – corresponding to 0.8 cM/Mb with the assumed mutation rate and generation time.

For each of the 9 ways we can combine three migration rates I generated datasets of 1, 10 and 50 Mbp. Like in the previous chapter I only present a subset of the results as all 27 would take up too much space.
(a) Here the maximum likelihood is simply far from the correct value.

(b) The thing to notice here is the right curve which looks very nice, but the log likelihood only changes by about 1.5 over the whole range.

(c) This run went well, and the curves are nice when the model gets 50Mbp to work with.

**Figure 6.1. Log likelihoods.** The plots shows likelihoods in an area around the correct value. Only the 50Mbp datasets are shown. We would like to see a peak at roughly the correct value and with a large change in value from minimum to maximum.
6.2 Results

While not shown here, going from 1 to 10 to 50Mbp improves the accuracy of the estimates just like in the previous chapter with varying coalescence rates. But even with 50Mbp it can be hard to recover a low migration rate. Figure 6.1 (a) and (b) show examples of how either the maximum is at the wrong place or the change in log likelihood is barely significant over the entire range we look at. Most of the combinations look fine – an example can be seen in figure 6.1 (c) – but when there are problems it is always in the third epoch and most severe if the migration rate is low.

In figure 6.2 there is a plot for each of the epochs/variables. Within the individual plots I have grouped the data by what the correct value is, just like in the previous chapter.

The estimates are not as close as when we were varying the coalescence rate. This might indicate that the migration rate needs even more data or it could be more affected by the fact that the other parameters are fixed. But while it is not as accurate as estimating C, none of the boxes are completely off – well within a factor of two when the migration rate is not too low.

I will claim that the migration rate can be recovered, although it probably needs more data than the coalescence rate and the final estimate might not be as accurate. I don’t think migration rate needs to be as accurate since the relative levels are more interesting, whereas coalescence rate would be used to calculate a population size that people can relate to much easier.
Figure 6.2. Estimates. The plots shows boxplots of the estimates in each epoch, grouped by what the correct value is in the plotted epoch.
Chapter 7

Conclusion

I have described a system for generating CoalHMMs, based on colored Petri nets and continuous time Markov chains. With this framework I have generated four types of models. A model of speciation, with and without migration, a model for estimating variation in population sizes over time, similar to Li & Durbin [6] and finally a model for varying migration rate over time (as a way of modelling admixture). Subjectively I will claim that it is very easy to set up the various models and varying the number of states, which parameters are free and so on – almost all of the boilerplate that is involved stem simply from there being a lot of things to specify.

All the models were tested on simulated data first to see whether or not they are capable of producing useful results under ideal conditions. This was followed by runs on real data, although it of course is harder to verify the accuracy of these estimates.

I think it is safe to say that the models we generate are generally sound – and capable of inferring multiple parameters. From my experiments with the simulated data, we saw that the inferred results can get very close to the true values used in the simulation, except for the recombination rate which was consistently underestimated.

Unfortunately neither the real world or the process that produces the data is as neat as a simulation. The real data I have to work with has noise and missing data from both the DNA sequencing and the from the alignment with a reference genome which means the picture is a lot more murky.

Several things can be done to filter the data, either by looking at the sequences alone (in particular regions with a large amount of missing data) or using metadata like the sequence quality map provided with each sequence.

Despite the problems with data quality it is possible to get some reasonable results on the different sets of data we have worked with. When looking at two species that are closely related the variation in the results go down.
It might be possible to completely replicate the PSMC model from Li & Durbin [6] if there wasn’t a large increase in time when adding a lot of intervals to the model. Since the only difference between the epochs in that case would be the rates and not the structure it seems like a place that is ripe for a clever optimization.

7.1 Future work

There are several ways to proceed, but the two ideas that first come to mind is further work on admixture and some work on performance.

A different way of modelling admixture events, instead of as varying migration, would be to have instantaneous events when crossing the epoch barriers.

As I have shown in chapter 2 the complexity increases quite a lot with the number of species in the model, number of epochs etc. It would be interesting to see if it is possible to either find some way of reducing the size of the generated graphs or some way that will allow us to skip some of the possible paths through the graph.

Finally an obvious thing to try would be more datasets. I already have access to a set of bears and more great apes, but really any set of closely related species could be used.
Bibliography


[9] Thomas Mailund, Anders E. Halager, Michael Westergaard, Julien Y. Dutheil, Kasper Munch, Lars N. Andersen, Gerton Lunter, Kay Prufer,


