Exploring the Matrilineal Genomic History of Aboriginal Australians through Mitochondrial DNA

Sophie Anne Schiller

April 23, 2021

Thesis submitted for the degree of
Master of Philosophy
in
Statistics
at The University of Adelaide
Faculty of Engineering, Computer and Mathematical Sciences
School of Mathematical Sciences

Contents

Signed Statement

I certify that this work contains no material which has been accepted for the award of any other degree or diploma in my name in any university or other tertiary institution and, to the best of my knowledge and belief, contains no material previously published or written by another person, except where due reference has been made in the text. In addition, I certify that no part of this work will, in the future, be used in a submission in my name for any other degree or diploma in any university or other tertiary institution without the prior approval of the University of Adelaide and where applicable, any partner institution responsible for the joint award of this degree.

I give permission for the digital version of my thesis to be made available on the web, via the University's digital research repository, the Library Search and also through web search engines, unless permission has been granted by the University to restrict access for a period of time.

I acknowledge the support I have received for my research through the provision of an Australian Government Research Training Program Scholarship. This work was also supported with supercomputing resources provided by the Phoenix HPC service at the University of Adelaide.

Signed: .. .. Date: 23/04/2021

Acknowledgements

First and foremost, I would like to thank my supervisors, Associate Professor Gary Glonek and Dr Ben Rohrlach. Your advice and guidance throughout the thesis writing process has been invaluable. Thank you to the team at the Australian Centre of Ancient DNA (ACAD) for the use of their data, and also specifically Raymond Tobler, João Teixeira, Bastien Llamas, and Gludhug Purnomo for the many discussions throughout the model design process. To Jono Tuke, thanks for the feedback on Chapter 2; and to Melissa Humphries, thanks for the many helpful and motivating chats in the tea room, as well as for offering to provide feedback. Thank you to all corresponding authors that provided helpful advice when I had questions about their papers.

Max, Caitlin, and Louise, it's been great working at the same end of the office as you - I have always loved our tea breaks and conversations. To Ashley, thanks for the lunch breaks, the discussions about thesis writing and nothing in particular, and the amazing cakes and biscuits you've brought in over the past six years of uni. Thanks to the 'Frolic Fridays' group of Anthony, Aline, and David: Friday lunches were definitely one of the highlights of my weeks. Thanks also to Jack Maclean, for the well-timed motivational talk towards the end of my thesis.

Tarnya, Jordan, and Stef: it's been great either living with you or regularly catching up with you over the course of my thesis. Your friendship has made my time outside of uni immeasurably better, as have your fantastic tastes in music, cheeseboards, interior decorating, and bicycle storage solutions. Thanks to Kaitlin for always having time for a chat over dessert or coffee. Tyson, thanks for suggesting the occasional procrastination coffee and having impeccable shiraz recommendations.

To my parents, thanks for your love and support over the years. I appreciate everything you've sacrificed to get me to uni in the first place.

Finally, to Anthony: thanks for the love, support, maths puns, work-life

balance chats, fun recipes, and repeated suggestions that I use an ergonomic office chair at home.

Abstract

We investigated the route taken in the first human migration from southeast Asia to southern Australia. Mitochondrial DNA (mtDNA) sequences were analysed using phylogenetic trees, then a simulation study was used to identify the migration route that best described the observed data. The simulation study compared the observed mtDNA data to DNA simulated under nine different scenarios. If the actual migration history resembled one of the scenarios, we would expect the summary statistics of the observed and simulated mtDNA to be similar.

The simulation study did not conclusively select one migration route, or a set of migration routes, that best described the data. To check that this was a consequence of the data and not the modelling assumptions, we considered a range of extensions to the initial simulation study. This included modifications to the time between migration events through the southeast Asian islands, exploration of the effect of different migration levels after the initial settlement of all populations, and then a more detailed haplogroup analysis.

From these extended analyses, we found that increasing the length of time between migration events through the southeast Asian islands resulted in a greater ability to distinguish between migration routes. We also found that we were less able to distinguish between migration routes when identical patterns of ongoing migration were applied to all migration routes. Our further haplogroup analysis used a simulation study to compare three different migration routes. While we could not determine the geographical route taken, we could determine the patterns of ongoing migration that occurred after settlement. The observed mtDNA data was consistent with ongoing, low-level migration between northeastern Australia and New Guinea.

We conjecture that rapid migration through the southeast Asian islands resulted in minimal evidence of the initial migration events in the mtDNA

sequence data. When considering the phylogenetic trees, we noticed low branch support values around the time of settlement of Australia, which further supports our conjecture.

Our findings suggest that small amounts of ongoing migration between north-eastern Australia and New Guinea occurred. Our inability to reliably determine the initial migration route taken also illustrates the limitations of mtDNA analysis, and highlight the importance of using nuclear DNA for future studies in this area.

Chapter 1

Introduction

1.1 Motivation

'Where did humans come from?' has been a question of interest throughout much of human history. While much progress has been made on this topic, many questions remain unanswered. Ongoing research aims to learn more about the evolutionary history of humans, as well as the details of human migration into and throughout all continents of the world.

First, we present a brief discussion on what is already known about human prehistory, which is the time before written human records. Anatomically modern humans, *Homo sapiens*, were not the first humans to inhabit the continents outside of Africa. Neanderthals and Denisovans predate *Homo sapiens* as a species; Neanderthal remains have been found in Europe and eastern Asia, while Denisovan remains have been found in Asia (specifically, in Denisova Cave, Russia) [72]. Outside of Africa, most humans today have Neanderthal ancestry, Denisovan ancestry, or both Neanderthal and Denisovan ancestry due to interbreeding between the different human species [74]. While there are other ancestral human species, some of which also inhabited Eurasia and the southeast Asian islands [18, 1], we do not discuss these in detail here. Our research concerns only the history of modern humans, and this is where we now focus.

The most widely accepted theory for the origin and later migration of modern humans is the Out-of-Africa theory, which hypothesises that modern humans originated in Africa, and subsequent migration events resulted in the peopling of all other continents [44]. First, fossil evidence suggests that the Levant

region (the present-day Middle East) was inhabited 126,000 - 74,000 years ago [70], although early migration into this region did not lead to further migration events. It is estimated that mainland southeast Asia was inhabited between 70,000 [94] and 50,000 years ago [63]. This was followed by the peopling of the southeast Asian islands, and then Australia approximately 50,000 years ago [85]. Later, Europe was inhabited approximately 45,000 years ago [77], and the Americas were inhabited from Eurasia via the Bering land bridge approximately 15,000 years ago [48].

These estimates have been taken from a range of studies in the fields of archaeology and genetics. Due to technological advances, and the corresponding decrease in the cost of DNA sequencing, human prehistory has only recently become an area that can be studied through the analysis of genetic material.

We investigate only a small part of this vast migration history: the migration of modern humans from southeast Asia into what is now Australia, which occurred approximately 50,000 - 65,000 years ago [85, 15]. We aim to further the research of Tobler *et al.* [85], who used mitochondrial DNA (mtDNA) to study the initial peopling of Australia by Aboriginal Australians. They investigated the timing of the peopling of Australia, and also suggested that coastal migration occurred within Australia based on the distribution of mtDNA haplogroups within the country.

We will extend their research by investigating the migration path taken through the southeast Asian islands, as well as the migration path taken within Australia. The DNA samples that we will analyse have been collected from descendants of the first modern humans to inhabit Australia and the southeast Asian islands, and many of these samples were also used in the analysis by Tobler *et al.*.

This topic will be explored through a simulation study, in which we will investigate whether our observed mtDNA samples are consistent with a range of previously published migration routes. We will also perform a phylogenetic analysis, which will describe the genetic relationships between the populations that are relevant to our research question.

1.2 Thesis Structure

The theory required for understanding DNA simulation is described in Chapter 2. This introduces the concept of phylogenetic trees, which are then pre-

sented in the following chapter. We also discuss mitochondrial DNA and coalescent theory.

We apply the theoretical discussion on phylogenetic trees and coalescent theory in Chapter 3, in which we present our phylogenetic analysis in the form of phylogenetic trees and any relevant parameter estimates. Further theoretical details of the programs used to reconstruct these trees and other estimates are also provided.

Before introducing the simulation study and its results, we describe the mathematical theory underpinning different classification and dimension reduction methods in Chapter 4. These methods will later be used to establish the results of the simulation study.

The migration models that form the foundation of the simulation study are defined in Chapter 5. Migration models clearly define all parameters required to simulate DNA, and so we will need to specify geographical routes, migration times, effective population sizes, and any other parameters that describe the demographic history of populations or the type of DNA that we wish to simulate.

All previous chapters are brought together in Chapter 6, where we present the results of the simulation study and discuss them in context. This involves applying the classification methods introduced in Chapter 4 to the summary statistics resulting from the simulations defined in Chapter 5.

Chapter 7 investigates the results presented in Chapter 6 in more detail. We conduct further simulations that explore the effects of relaxing the assumptions made when defining the migration models, and whether any information can be gained from exploring different sub-populations at a finer resolution. We also discuss what we would theoretically expect to see under different migration scenarios.

All code used to analyse data will be available via the sahul_migration_pathways repository at `https://github.com/sophie-schiller`.

Chapter 2

DNA Simulation and Coalescent Theory

In this chapter, we describe the use of genetic simulators to simulate DNA, as well as the theoretical basis of these simulators. First, we describe some key properties of DNA, and then introduce the use of trees to describe genetic relationships. We define the Wright-Fisher model, and show how this model can be used to derive the standard coalescent. We then consider how changes in effective population size over time and migration events can be incorporated as extensions of the standard coalescent. We describe how DNA is simulated given a coalescent tree, and then conclude by discussing the mutation process of DNA.

Genetic simulators are programs that simulate DNA, given some pre-defined population history. This history includes any changes in effective population size, creation or extinction of populations, and migration between populations.

Genetic simulators fall under two general categories: backward-time (*e.g.* coalescent) or forward-time. Backward-time simulators start at the present day or some other specified time, and then model how population changes backward in time. These simulators are based on coalescent theory, and are therefore more efficient than alternative simulators [98]. Forward-time simulators begin with some population in the past, and then model the change in this entire population through to the present day. While this type of simulator is more flexible, it is also less efficient. We use the coalescent simulator Bayesian Serial SimCoal (BayeSSC) [4, 23], due to its flexibility in simulating different population histories, the automatic calculation of summary

statistics for DNA, and the prohibitive amount of time many forward-time simulators take to produce a large number of simulations.

2.1 DNA

DNA commonly refers to autosomal DNA, which is the DNA inherited from both parents. There are also other types of DNA, such as mitochondrial DNA (mtDNA), X-chromosome DNA, and Y-chromosome DNA. In this project, we only use mtDNA. Mitochondrial DNA is shorter than autosomal DNA, having a length of 16569 base pairs (bp) compared to the approximately 2.9×10^9 bp that make up autosomal DNA [67]. Unlike autosomal DNA, mtDNA is circular, so that the last base pair at Position 16569 is next to the first base pair at Position 1.

Each DNA sequence is made up of four nucleotides: adenine, cytosine, guanine, and thymine. These are represented by the letters A, C, G, and T respectively. Each of the four bases is either a pyrimidine or a purine depending on its chemical composition: C and T are pyrimidines, while A and G are purines.

DNA changes over time, and one cause of this is the mutation process. There are many types of mutations that affect DNA in practice; for example, the insertion or deletion of DNA, or single nucleotide polymorphisms (SNPs). A SNP occurs when one base changes to a different base. This is also referred to as a substitution, and an example is given in Figure 2.1.

Some regions in the mitochondrial genome accumulate mutations faster than others, and are called hypervariable regions I and II. We can split the full genome into two main regions: the coding region and the control region. Both hypervariable regions lie within the control region. Since the hypervariable regions accumulate mutations at a greater rate than other regions, the hypervariable regions contain a greater proportion of mutations. Positions within the hypervariable regions are also more likely to be affected by multiple mutations. The hypervariable regions have a greater average mutation rate than the coding region, and also contain multiple 'hotspots' with a considerably greater mutation rate. Furthermore, the higher mutation rate and shorter length of the hypervariable regions means that these regions are more susceptible to back-mutations, where a single site undergoes two mutations. For example, an A could change to a T, and then back to an A; in this case, it appears as though a mutation never occurred. For these reasons,

Figure 2.1: Two DNA sequences with the SNP highlighted in red.

we will exclude the control region and only analyse the coding region in our analyses.

All substitutions are either transitions or transversions. Transitions occur within a purine/pyrimidine family; for example, a change from a C to a T is a transition, as is an A to a G. Conversely, transversions occur when a purine is substituted for a pyrimidine, or vice versa. For example, a change from an A to a C is a transversion. Many simulation programs include transition bias as a parameter, which allows transitions and transversions to occur at different rates for more realistic simulations. The transition bias can be estimated from the sequence data using a program such as ModelGenerator [42].

There are some considerations when using mtDNA to investigate population history. As mtDNA is passed along the maternal line, we can only gain insight into the matrilineal genetic history of the population. While this is a limitation, models based on mitochondrial DNA are also simpler due to the lack of recombination. Recombination is a process that affects nuclear DNA, and occurs when cells divide. DNA splits at some points along the DNA strand, and then joins back together (recombines) in a different way. This process is described visually in Figure 2.2.

2.2 Trees

Given a sample of mtDNA sequences, it is possible to infer a phylogenetic tree. Trees visualise the relatedness of sequences in a DNA sequence alignment, and can also provide information about the timing of events in the population history if they are calibrated. There are many different programs that can create trees from sequence data. Two commonly used programs are BEAST2 [8], which uses Bayesian methods to produce a tree, and IQ-TREE [60], which constructs maximum likelihood trees. These programs are discussed in more detail in Chapter 3.

To illustrate, we consider a simple, rooted phylogenetic tree as shown in Figure 2.3. The three modern sequences, A, B, and C, are at the tips of the

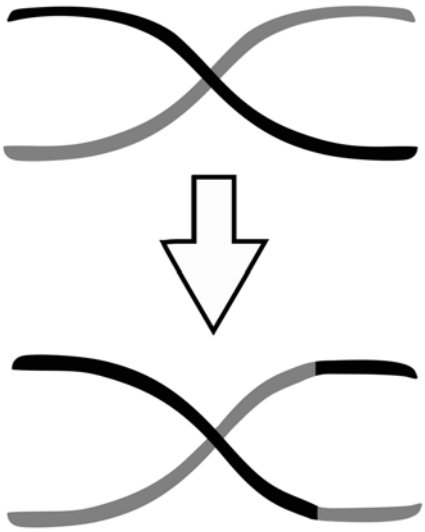


Figure 2.2: Chromosomes before and after recombination.

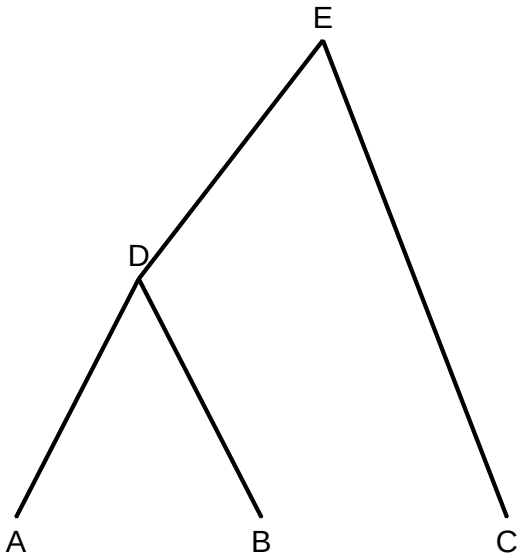


Figure 2.3: A phylogenetic tree for modern sequences A, B, and C.

tree. D is the common ancestor of A and B, and A and B are more closely related than either A and C or B and C. The most recent common ancestor (MRCA) of all samples is represented by E. Sequences at D and E are not sampled, but are inferred by the method used to construct the tree.

Both BEAST2 and IQ-TREE take a DNA sequence alignment as input and then construct a tree. However, this method of tree building is not suitable for DNA simulation. DNA simulation, at least using coalescent simulation programs, works by first building a tree based on a pre-defined population history, distributing mutations along the tree, and then simulating DNA according to the tree topology and distribution of mutations. Since DNA is simulated at the final step of this process, it cannot be used to construct a tree in the first step of the process. It is also unsuitable to use another source of DNA to build the tree, as the other source may reflect a different population history.

Consequently, trees must be constructed based only on the population history. One method for generating trees in this situation is through the use of coalescent theory. Next, we present the Wright-Fisher model, and how it can be used to derive the coalescent.

2.3 The Wright-Fisher Model

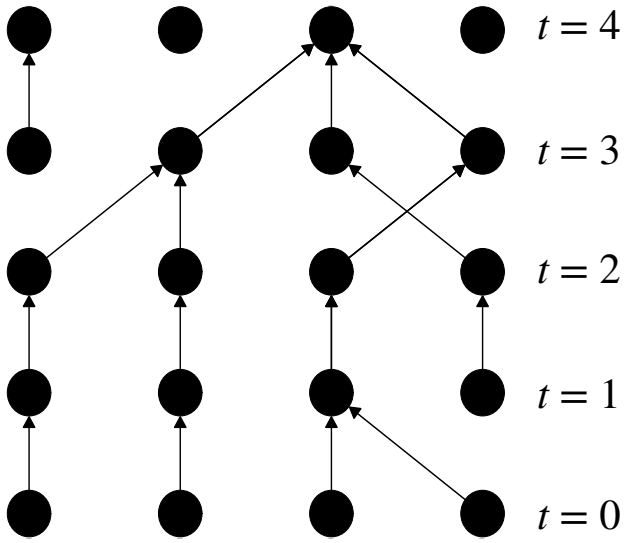
The Wright-Fisher Model, developed by Sewall Wright [95] and Sir Ronald Fisher [25], was one of the first models used to describe how the allele frequencies, and therefore the genetic characteristics of a population, change over time. This model assumes a constant population of size N, with every individual being replaced each generation. The population is not made up of males and females like a human population; instead, the Wright-Fisher model assumes a haploid population. By definition, individuals in a haploid population only carry one copy of the genetic material being passed on [61]. Mitochondrial DNA is one example of haploid inheritance, since it is only passed on through the maternal line.

Another assumption of the Wright-Fisher model is random mating in each generation, which means that an individual passes on their genetic information to any individual in the next generation with equal probability. This assumption is equivalent to each child randomly choosing one parent in the previous generation. Children only choose one parent instead of two due to the assumption of a haploid population.

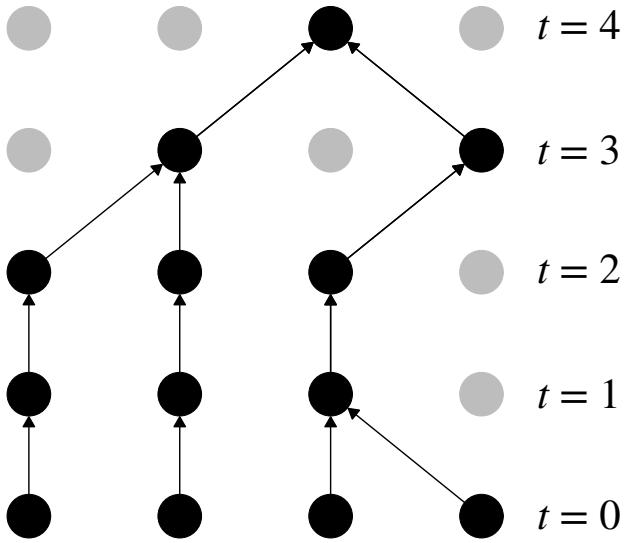
Figure 2.4.1 demonstrates random selection of parents for five generations. Once each child from the current generation ($t = 0$) has selected a parent, each individual from this generation ($t = 1$) must select a parent, and so on until the most recent common ancestor (MRCA) for all individuals in the present day ($t = 0$) is found. In Figure 2.4.1, this occurs four generations in the past ($t = 4$). Figure 2.4.2 only shows the selection of parents for descendants of the MRCA.

Since individuals are replaced in each generation, it is also helpful to consider lineages. In terms of Figure 2.4.2, a lineage is a path from an individual at $t = 0$ to the MRCA. Lineages merge backwards in time because individuals eventually share a common ancestor. Since lineages merge over time, we sometimes mention the number of distinct lineages. At time $t = 0$ there are four distinct lineages, but at time $t = 3$ there are only two distinct lineages.

A coalescent event occurs when the ancestors of two or more individuals select the same parent, *i.e.* when two or more lineages coalesce. For example, the lineages of the two leftmost individuals in Figure 2.4.2 coalesce three generations in the past, when $t = 3$. These coalescent events are modelled in continuous time in the coalescent model.



2.4.1: A realisation of the Wright-Fisher model showing the choice of parent for each individual in each generation.



2.4.2: The same realisation of the Wright-Fisher model as Figure 2.4.1, but individuals without offspring are now coloured grey. The choice of parent is only shown for the ancestors of individuals at $t = 0$. The single black circle in the top row is the MRCA of all four individuals in the bottom row.

Figure 2.4: A realisation of Wright-Fisher model for four individuals ($N = 4$) over five generations. Each generation is labelled according to the number of generations in the past t, with $t = 0$ representing the present day. The choice of parent is indicated by an arrow.

2.4 The Coalescent Model

The coalescent model can be described in different ways, but here the derivation of the standard coalescent model from the Wright-Fisher model is presented. The general Wright-Fisher model is similar to the example given in Figure 2.4 , except there are t generations instead of five generations, and some fixed population size N instead of four individuals. We also introduce the concept of a sample size n. The sample size is the number of lineages at $t = 0$, and the number of individuals whose genetic history is considered. In the example given for the Wright-Fisher model, $N = n = 4$.

First, we consider what it means for exactly two individuals to have a common ancestor, and then generalize this to any pair of individuals having a common ancestor. We can generalize this because an individual's choice of parent is independent of the choice made by any other individuals in the same generation.

Consider two specific individuals in a population of size N. The probability of these individuals randomly selecting the same parent is $1/N$. Therefore the probability of these two individuals *not* choosing the same parent is $1-\frac{1}{N}$. Since the assumptions of the Wright-Fisher model imply independence between generations, the probability of two specific individuals not selecting the same parent for t consecutive generations is

$$\left(1 - \frac{1}{N}\right)^t.$$

At this point, the time t describes the number of generations in the past. We rescale time to coalescent time τ by setting

$$\tau = \frac{t}{N}. \tag{2.1}$$

Let the random variable T_2 be the time taken for two lineages to coalesce. Then

$$\begin{aligned} P(T_2 > t) &= \left(1 - \frac{1}{N}\right)^t, \quad \text{or equivalently,} \\ P(T_2 > N\tau) &= \left(1 - \frac{\tau}{N\tau}\right)^{N\tau}. \end{aligned} \tag{2.2}$$

Notice that the form of Equation 2.2 is very similar to the limit characterisation of the exponential. For fixed τ, we let N go to infinity:

$$P(T_2 > t) \approx \lim_{N\to\infty} \left(1 - \frac{\tau}{N\tau}\right)^{N\tau} \tag{2.3}$$

$$= e^{-\tau}$$

$$\Rightarrow P(T_2 < t) \approx 1 - e^{-\tau} \tag{2.4}$$

$$= 1 - e^{\frac{t}{N}}. \tag{2.5}$$

Equation 2.4 can be identified as an exponential CDF with rate parameter one. After converting to generational time using Equation 2.1, the probability becomes an exponential CDF with rate parameter $1/N$ (from Equation 2.5). By the definition of the expected value of an exponential distribution, $E[T_2] = N$. This means that the expected time for two lineages to coalesce is N generations.

In Equation 2.3, we require N to go to infinity to derive properties of coalescent events. While an infinitely-sized population is unrealistic when modelling populations in practice, any sufficiently large value of N where the number of lineages k is small compared to N yields a close approximation.

The main consequences of a finite population size are multiple coalescent events in the same generation, or more than two lineages being involved in the one coalescent event. Instead of calculating all the possibilities under which this could happen, we directly calculate the probability of one or zero coalescent events happening and subtract this from one. If we define the number of coalescent events in the previous generation as C, then

$$\mathrm{P}(C \geq 2) = 1 - \mathrm{P}(C = 1) - \mathrm{P}(C = 0). \tag{2.6}$$

Suppose that there are k distinct lineages at some point in time. Consider the probability of two of these lineages coalescing in the previous generation. An equivalent scenario is k individuals having exactly $k - 1$ ancestors in the previous generation, *i.e.* the event $C = 1$. This means that exactly two individuals have chosen the same parent, and exactly one coalescent event has occurred.

For one coalescent event to occur, two individuals must select the same parent. For two specific individuals, this occurs with probability $1/N$. Now, we require all remaining individuals to choose different parents. If we consider the next individual, the probability of them selecting the same parent as the other two individuals is $1/N$, and so the probability of not selecting

the same parent is $1 - 1/N$. For the next individual, there are two parents selected already, and a similar argument is used. Finally, all probabilities are multiplied due to individuals selecting parents independently of each other. Therefore the probability of two specific individuals sharing a parent and all other individuals having different parents is

$$\frac{1}{N}\prod_{i=1}^{N-2}\left(1-\frac{i}{N}\right).$$

We also need to accommodate the fact that there are $\binom{k}{2}$ different pairs of individuals that could have the same parent, instead of only two specfic individuals having the same parent. Multiplying by the number of pairs gives

$$P(C=1) = \frac{\binom{k}{2}}{N}\prod_{i=1}^{N-2}\left(1-\frac{i}{N}\right). \tag{2.7}$$

Next, consider the probability of no individuals selecting the same parent in the previous generation, *i.e.* the event $C = 0$. Once the first individual selects a parent, the probability that the next individual selects the same parent is $1/N$, and so the probability that this individual selects a different parent is $1 - 1/N$. At this stage, there is a $2/N$ probability that the next individual selects the same parent as one of the previous individuals. We can extend this argument to find that

$$\mathrm{P}(C=0) = \prod_{i=1}^{N-1}\left(1-\frac{i}{N}\right). \tag{2.8}$$

Substituting Equations 2.7 and 2.8 back into the original expression found for multiple coalescent events in Equation 2.6,

$$\mathrm{P}(C\geq 2) = 1 - \frac{\binom{k}{2}}{N}\prod_{i=1}^{N-2}\left(1-\frac{i}{N}\right) - \prod_{i=1}^{N-1}\left(1-\frac{i}{N}\right). \tag{2.9}$$

The probability of multiple coalescent events in one generation for different population sizes and numbers of lineages is given in Figure 2.5. We can see that the probability of multiple coalescent events is negligible when the population size N is large compared to the number of lineages k.

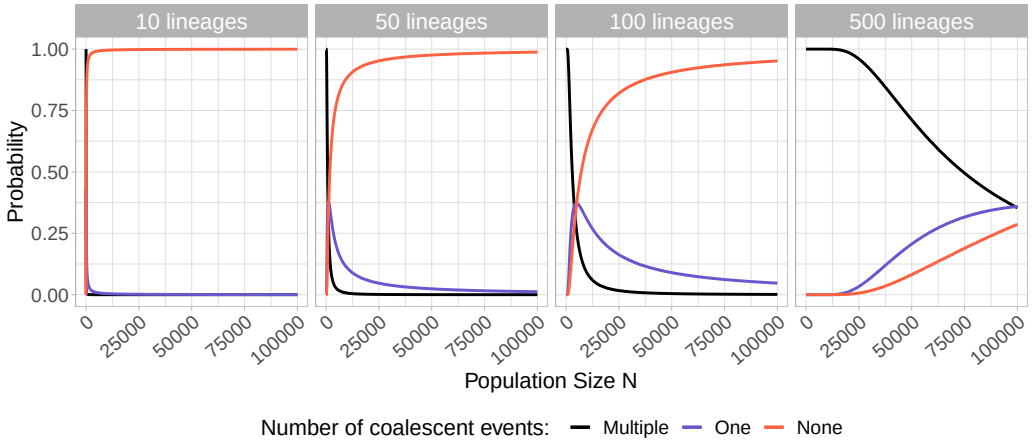


Figure 2.5: The probability of multiple coalescent events occurring in the previous generation, contrasted against the probabilities of exactly one coalescent event occurring and no coalescent events occurring in the previous generation. This was calculated for different numbers of lineages k in different population sizes N.

When deriving rates of coalescent events, we have only considered coalescent events for exactly two lineages. Suppose there are now k lineages in the population; we want to determine the rate at which coalescent events occur between any two lineages. We can think of this as waiting for one of $\binom{k}{2}$ sets of two lineages to coalesce, where the waiting times for all coalescent events are distributed according to an exponential distribution with rate parameter $1/N$ (see Equation 2.5). By the properties of independent and co-occurring exponential distributions [7], coalescent events between any two lineages occur according to an exponential distribution with a total overall rate of $\binom{k}{2}/N$.

Again by the definition of an exponential distribution, a coalescent event is expected to occur after $N/\binom{k}{2}$ generations. Interpreting this expected waiting time in context, we find that as the number of lineages decreases, the expected time until a coalescent event increases. Note also that since the rate of coalescence depends on N, coalescent events are expected to occur at a faster rate in a smaller population than in a larger population. For a more detailed discussion of coalescent theory and the mutation process, the reader is directed to Wakeley's *Coalescent Theory: An Introduction* [91].

2.4.1 Implementation of the coalescent model

Coalescent trees can be constructed in practice using either a continuous implementation or a discrete implementation.

To randomly generate a coalescent tree, a pair of lineages is randomly chosen from the population, and then the time until their coalescence is drawn from an exponential distribution with rate $N/\binom{k}{2}$, assuming generational time is used. The process is repeated until the final two lineages coalesce into one lineage. This single lineage persists backwards in time, but any information further in the past does not affect the relationship between present-day individuals in coalescent theory.

While the standard coalescent can be simulated in continuous-time using the process outlined above, BayeSSC uses a discrete implementation. Each generation is one time step, and at each time step, the probability of two lineages coalescing out of a total k lineages is

$$P(\text{A coalescent event occurs}) = \frac{\binom{k}{2}}{N}. \tag{2.10}$$

In Equation 2.10, N and k are the population size and number of distinct lineages at the current generation [23].

The possibilities of multiple coalescent events or the coalescence of multiple lineages in one generation are both included in the simulation algorithm used by BayeSSC, where the probability of another, further, coalescent event in each generation is calculated after the first coalescent event occurs [23]. THe algorithm continues simulating the current generation until coalescent events have ceased to occur, and only then it begins simulating the next generation.

2.4.2 Changes in effective population size

As seen in the introduction to the standard coalescent model, the rate of coalescence directly depends on N. For sudden changes in population size, *e.g.* a significant decline in population size, the rate of coalescence is immediately adjusted at the time of the population resize.

Alternatively, one may wish to model the exponential growth or decay of a population over time. Suppose a population grows exponentially with rate r forwards in time (*i.e.* the population decays backward in time). We note

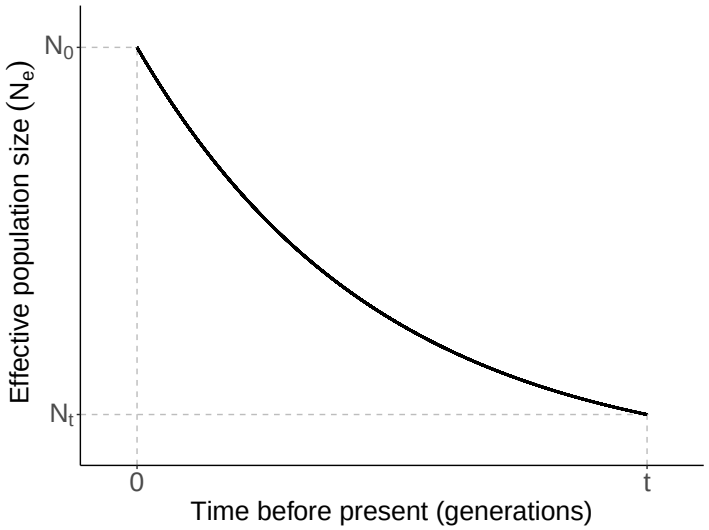


Figure 2.6: A graph showing exponential growth of effective population size as in Equation 2.11, where r is positive. The population size shrinks going backwards in time, which is equivalent to population growth in forwards time.

that while Figure 2.6 displays population growth, all equations in this section apply to population decay as well. Let N_0 denote the effective population size at the time when the exponential growth begins, and N_t the effective population size t units of time in the past. Then population growth can be modelled with the equation

$$N_t = N_0 e^{-rt}, \tag{2.11}$$

where $r \in \mathbb{R}\backslash\{0\}$. This pattern of population growth is described in Figure 2.6.

Since the rate of coalescence depends on N, which was previously fixed in the standard coalescent so that $N_t = N$ for all t, we can simply substitute N_t for N:

$$\begin{aligned} \text{Rate of Coalescence with population growth} &= \frac{\binom{k}{2}}{N_t} \\ &= \frac{\binom{k}{2}}{N_0 e^{-rt}} \\ &= \frac{\binom{k}{2}}{N_0} e^{rt}. \end{aligned}$$

Now the rate of coalescence depends on the current number of generations that have passed since the start of the exponential growth, and so the rate is constantly changing. After one generation, the rate of coalescence has increased by a factor of e^r. Instead of constantly updating the population size and rate of coalescence, it is simpler to simulate a tree using a fixed rate of coalescence first, and then rescale time afterwards according to the exponentially changing rate of coalescence.

Suppose that a tree is simulated using a fixed rate of coalescence, under some timescale $\bar{t}$. The tree from a fixed coalescent rate is the same as a tree constructed when the rate of coalescence is undergoing exponential change, but on the actual timescale t.

Let exponential population growth occur backwards in time from the current time $t = \bar{t} = 0$, and let t be the number of generations in the past. For the tree generated under a fixed rate and the tree generated under an exponential rate to be the same, the timescale of the tree with the fixed rate ($\bar{t}$) must grow exponentially quicker:

$$\begin{aligned} \bar{t} &= \int_0^t e^{rx} dx \\ &= \frac{1}{r}(e^{rt} - 1). \end{aligned} \tag{2.12}$$

To calculate event times on the actual timescale t, we rearrange Equation 2.12 to find

$$t = \frac{1}{r}\log(\bar{t}r + 1).$$

In the context of the population growth seen in Figure 2.6 where r is positive, this means that the tree generated under a fixed rate on timescale $\bar{t}$ would be compressed under the actual timescale of t. Intuitively, as the population size decreases in the past, we expect the rate of coalescence to increase. This means that the expected waiting time for coalescent events decreases, and the tree should be shorter in length.

Time is only rescaled while the effective population size undergoes decay or growth. As soon as the effective population size stabilises at a constant value, the rate of coalescence is as described in Section 2.4 and time is no longer rescaled.

Coalescent events under an exponentially changing effective population size also have a discrete implementation. The effective population size N is up-

dated at each generation according to the growth rate r, and then probabilities of coalescence are calculated as in Equation 2.10, using the updated value of N from Equation 2.11 at each time step.

2.4.3 Creation of new populations

When modelling demographic histories with migration, we often consider people moving between two existing populations in a generation. However, we can consider the creation of new populations along the migration path as migration events as well.

Consider large-scale migration occurring forwards in time. At first, one population is present in one location. After reproducing for some amount of time, some individuals may split from this population and move to a different location. While in reality there may be some individuals that move between populations, we first consider the simpler concept of no ongoing migration. This new population will then continue to evolve until a group of individuals leaves the population and moves to a new area. Suppose that this process repeats until there are D distinct populations that have been created from preceding populations.

To model this scenario using the coalescent, we need to consider the same problem backwards in time. Here, we will start with D populations that evolve independently backwards in time. Instead of individuals leaving a population to make a new population as when time moves forward, the population merges back into the previous population, leaving $D-1$ populations to continue evolving. This process repeats until the second population merges with the one remaining population, which was the starting point of our migration scenario. In this thesis, we only consider the coalescent where the times of population merges are fixed values.

This scenario is not dissimilar to the standard coalescent, as in both cases lineages can only coalesce within a population. However, the standard coalescent is only made up of one population. If there are D independent populations in the present day and no populations ever merge, this scenario is equivalent to building D independent coalescent trees.

As populations merge, the lineages from previously separate populations can potentially coalesce. This is illustrated in Figure 2.7, where populations merge at times t^* and t^{**}.

The discrete implementation of the coalescent model does not change greatly

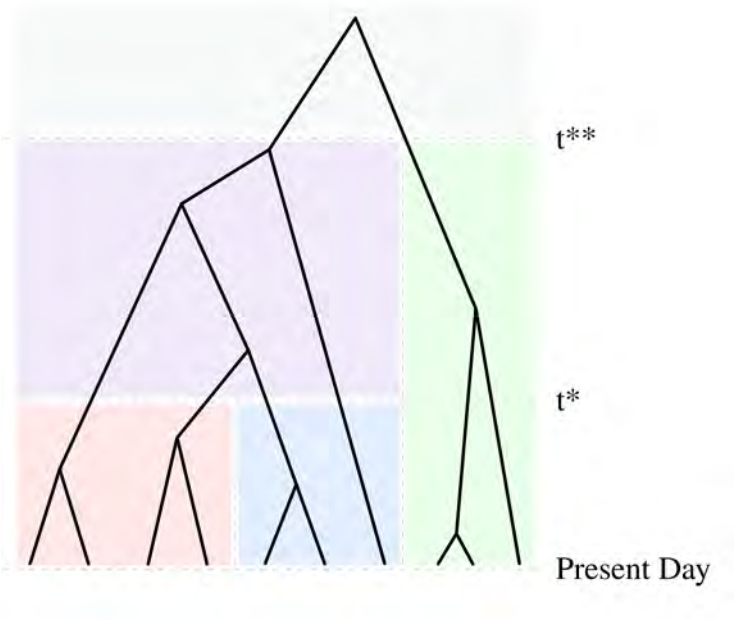


Figure 2.7: A coalescent tree for three populations that merge backwards in time. Individual populations are colour-coded at each point in time. At the present day, there are three distinct populations, coloured red, blue, and green. At time t^*, Population 1 merges with Population 2, creating the merged population in purple. At time t^{**}, the two remaining populations merge, forming one population in grey. Any coalescent events involving two lineages from two different populations must occur after the populations merge, looking backwards in time.

when extended to consider multiple populations, but the probabilities of coalescent events are calculated for each population in each generation. An important point to note regarding the simulation process is that the timing of population merges must be known before simulation.

If there is uncertainty about the times that populations merged (or any other parameter), a prior distribution can be used instead of a fixed value. A prior distribution in this case is a valid probability distribution that encompasses the reasonable range of values for a given parameter. While prior distributions can be used in BayeSSC as part of a Bayesian inference framework to find parameter values, we do not use them for this purpose here.

2.4.4 Post-settlement migration

Instead of an entire population merging into another, sometimes individuals, and therefore lineages, can move between populations. From here, we will refer to small numbers of individuals moving between already-established populations as post-settlement migration. In this section, a migration event refers to a case of post-settlement migration (unless otherwise stated).

The time to a migration event occurs according to an exponential distribution with rate λ_M [61]. This means that when considering events, either a migration event or coalescent event could occur next at any given time. The coalescent process and migration process occur independently of each other, and so we can apply a property of exponential distributions to determine the type of event that occurs [7]. By this property, some type of event (coalescence or migration) will occur with rate $\lambda_M + \lambda_C$, where λ_C is the rate of coalescence. After an event occurs, the type of event can be assigned. The probability of each event occurring is proportional to its rate:

$$P(\text{coalescence} \mid \text{event occurs}) = \frac{\lambda_C}{\lambda_M + \lambda_C}$$
$$P(\text{migration} \mid \text{event occurs}) = \frac{\lambda_M}{\lambda_M + \lambda_C}.$$

Once the type of event is sampled, lineages are chosen at random and the event occurs.

To implement this on a generation-by-generation timescale, we refer back to the Wright-Fisher model. Suppose that there are two populations instead of one. At each generation, there is a probability that an individual chooses a parent in the other population. In a similar way, BayeSSC uses a matrix with

fixed probabilities of migration between populations to implement migration. These migration matrices are square, but not necessarily symmetric. For a migration matrix $M = [M_{ij}]$, M_{ij} is the probability of a lineage migrating from population i to population j *backwards in time*, in one generation.

2.4.5 The mutation process

Using the theory developed in previous sections, we can efficiently simulate a coalescent tree under a range of demographic histories, such as changes in effective population size, migration events, and creation or extinction of different populations. To simulate DNA sequences from these trees, we need to simulate a pattern of mutations that results in differences between present-day DNA sequences, which are found at the tips of the tree. This can be achieved by distributing mutation events along the branches of the coalescent tree. In the coalescent model we only consider point mutations (substitution mutations), *i.e.* a one-letter change of base at a given site, and do not incorporate other types of mutations, such as insertions or deletions. The substitution rate μ defines the mutation process, and is in terms of the number of substitutions that occur at one site in a year. However, it is sometimes given in terms of the number of substitutions that occur throughout an entire sequence in a year or a generation. Here, we assume the substitution rate is given as substitutions per site per year, unless stated otherwise.

Substitutions are distributed according to a Poisson process with rate μ along the tree. The process begins at the MRCA and continues downwards, distributing substitutions independently along each branch. Since the occurrence of substitutions is a Poisson process, the waiting time until a substitution occurs is exponentially distributed, as with coalescent and migration events. If the waiting time for a substitution event is longer than the branch length, no substitutions occur on that particular branch. Once the locations of the substitution events on the tree have been sampled, the substitution is distributed spatially along the DNA sequence. This substitution can be randomly distributed, but that is not a very realistic model. Instead, some regions are allowed to accumulate more substitutions than others by allowing slight variation of the substitution rate between sites.

Rate variation between sites is often parameterised by a discrete gamma distribution defined by a number of rate classes n and a shape parameter α [96]. The discrete gamma distribution is created by partitioning the corresponding continuous gamma distribution into n equiprobable intervals; the value assigned to each interval is found by calculating the mean of the gamma

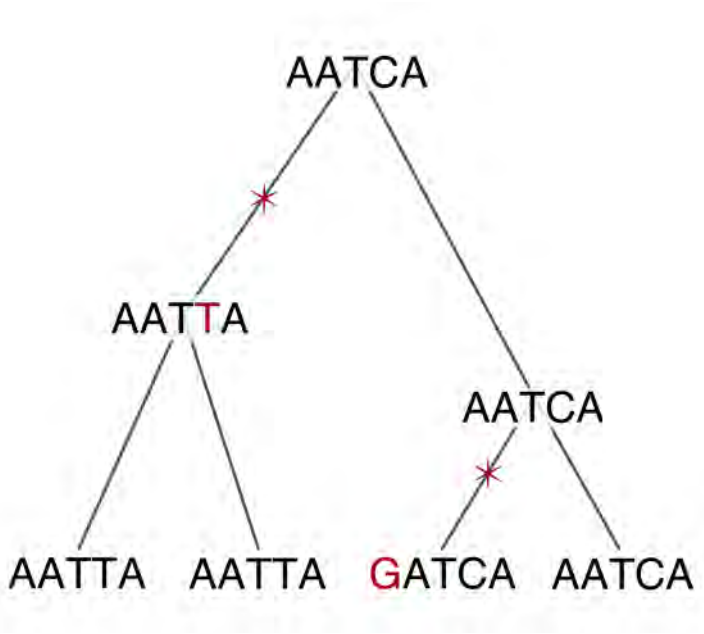


Figure 2.8: An example of how the mutation process is used to simulate present-day DNA sequences. A substitution event (red star) affects all sequences after it, when considered forwards in time. The substitution from the substitution event is shown as a red site in the DNA sequences. The DNA sequences at internal nodes are shown to illustrate the concept, but only the sequences at the tips would be outputs of the simulation program.

distribution within each interval. As n goes to infinity, the discrete gamma distribution approaches the continuous gamma distribution.

A continuous gamma distribution is parameterised by the shape parameter k and the scale parameter θ. A scale parameter is not specified for the discrete gamma distribution because the effect of rate variation is multiplicative; sampling one from the gamma distribution means that the substitution rate remains unchanged. Since the mean of the gamma distribution is forced to be one a scale parameter is not required, as it can be calculated from the shape parameter. This arises from the formula of the expected value of a continuous gamma distribution, where for some random variable $X \sim \text{Gamma}(k, \theta)$, $\mathbb{E}[X] = k\theta$.

Both the shape α and number of rate classes can be determined from DNA sequence data using a program such as ModelGenerator [42].

Simulation of DNA sequences based on the location of substitution events must occur from the MRCA downwards, as descendants inherit any substi-

tutions that have occurred earlier in the tree. This is explained visually in Figure 2.8. To implement the mutation process using a coalescent simulator such as BayeSSC, a random DNA sequence is generated at the MRCA. Substitutions are then simulated according to the distribution of substitutions on the tree. The spatial location of the substitution on the DNA sequence is chosen probabilistically, taking into consideration the rate variation between sites. We also need to consider the type of substitution, *i.e.* which base will be selected - this is accomplished by defining a substitution model. All substitution models referred to in this thesis are defined mathematically in Appendix A.2, but we also present an outline of three substitution models here to discuss limitations and software implementations.

For a substitution to occur at a given site, the current base at the site must change to any of the other three bases. The simplest substitution model is the Jukes-Cantor (JC69) model (see Appendix A.2.1), which selects a different base at random [39].

As stated earlier, transitions and transversions occur at different rates, and so the Jukes-Cantor model is not realistic. The simplest model that allows for different substitution probabilities for transitions and transversions is the Kimura two parameter (K2P) model [43] (see Appendix A.2.2). This substitution model is parameterised by either the transition transversion ratio or the transition bias. The transition transversion ratio κ means that for every transversion that occurs, κ transitions are expected to occur. When discussing transitions and transversions under Section 2.1, we defined the transition bias b as the proportion of substitutions that are transitions. The transition transversion ratio κ can be calculated from the transition bias by applying Equation 2.13,

$$\kappa = \frac{b}{1-b}. \tag{2.13}$$

There are a large number of substitution models, each allowing different sets of substitutions to have different rates. Unlike the two models presented here, some substitution models allow unequal base frequencies, *e.g.* more A's than T's in a sequence. BayeSSC enforces equal base frequencies at the MRCA to simplify the simulation process. One of the more complex models is the General Time Reversible (GTR) model (see Appendix A.2.7), which allows different substitution rates for each type of substitution as well as unequal base frequencies [83].

It is important to note that substitutions occur according to a *finite sites model* in most coalescent simulators. This means that substitutions are al-

lowed to occur at the same site twice, potentially resulting in back-mutations. A back-mutation occurs when one site undergoes two substitutions, where the second substitution returns the site to its ancestral base. These substitutions would not be identified as substitutions when calculating summary statistics. However, mutation rates are very low (in the order of 10^{-8} mutations/site/year), and sequences are 15447 base pairs long due to the length of the coding region of mtDNA. This means that back-mutations rarely occur.

While many studies have been conducted to estimate the substitution rate for human mitochondrial DNA, in this project we will use the substitution rate for the coding region of mtDNA calculated by Fu *et al.* [28], of 1.57×10^{-8} substitutions per site per year. To determine the number of substitutions for the whole coding region per generation, we need to multiply the substitution rate by the length of the coding region (15447 base pairs) and the human female generation time (25 years [24]). This gives a substitution rate of 6.0629×10^{-3} substitutions per mtDNA coding region per generation.

The previous sections contain enough information to simulate a coalescent tree based on the demographic history of a number of populations, and then simulate mtDNA that would have occurred under this history. The next chapter describes the model design process and presents some methods for model selection, where each model describes a different migration route.

Chapter 3

Phylogenetic Methods and Analysis

In this chapter we introduce our data and conduct what is known as a phylogenetic analysis. A phylogenetic analysis explores the relationships between the sampled individuals and the genetic history of the population. This analysis involves the reconstruction of phylogenetic trees, which were first introduced in Section 2.2.

First, we describe the aligned mtDNA sequence data and how this data is filtered to exclude certain sequences and genomic regions. Next, we introduce the theory behind maximum likelihood trees, as implemented in IQ-TREE, before reconstructing a maximum likelihood tree. We then explain how BEAST can be used to produce a phylogenetic tree (referred to as a 'BEAST tree') while simultaneously recovering past effective population size, and present the results of this analysis. We conclude this chapter by discussing the results of these analyses in context.

3.1 Data

Aligned mtDNA sequence data were provided by the researchers at the Australian Centre for Ancient DNA (ACAD), in the School of Biological Sciences at the University of Adelaide. This dataset included the samples collected as part of the Aboriginal Heritage Project (AHP) [85]. The data are uniquely provenanced because, in addition to collecting hair samples, information was collected about the geographical origin of each sample donor's ancestors. The

geographical metadata included where the grandparents of the sample donors lived, and therefore this recorded location predates the European colonisation of Australia and the forced relocation of Aboriginal peoples. Hence, there is a reliable latitude and longitude for all samples within Australia in the form of a pre-European, ancestral country for each sample donor.

The remaining sequences are from individuals with hunter-gatherer ancestry from the islands of southeast Asia. Hunter-gatherers were the first anatomically modern humans to inhabit the islands of southeast Asia. Other migration events into this area soon followed, notably the Austronesian expansion approximately 5 thousand years ago which introduced agriculture to the region [47]. Since we want to investigate the first wave of human migration into the islands of southeast Asia, we analyse mtDNA sequence data from individuals whose ancestors were part of the first migration.

The general location is also recorded for the samples from the southeast Asian islands, but not at the resolution of latitude and longitude. The location metadata is sufficient as we will later assign samples to broad geographical regions instead of specific, individual locations.

We note that all samples are comparatively recent (obtained within the past 100 years), and there are no ancient samples in our dataset.

The raw dataset contains 678 aligned mtDNA sequences in total. The general sampling locations of these sequences are given in Table 3.1. A full table of metadata containing all available location metadata can be found in Appendix B, in Table B.1.

3.1.1 Data filtering

To filter the aligned mtDNA sequence data we need to consider which sequences, and also which sites within each sequence, are best suited for this analysis. Only sequences relevant to the candidate migration routes (described in Section 5.1) are retained. For example, we do not include any mtDNA sequences where the samples were taken from individuals from the Pacific Islands east of Australia (*e.g.* Fiji) because the location is not informative when considering migration from southeast Asia to Australia. We retain all sequences from Borneo, Timor-Leste, the island of New Guinea and other nearby islands, and all Australian sequences with reliable provenance. A number of Australian sequences that had no corresponding latitude or longitude were also excluded from the analysis. A map of the corresponding sampling locations for the retained sequences is given in Figure 3.1.

Sampling Location	n	n (Final Dataset)	Population ID: (Number) Region
Borneo	58	12	(0) SE Asia
The Philippines	33	-	
Timor	15	8	(1) Southern Wallacea
Remote Oceania	20	-	
Santa Cruz Islands	40	-	
New Britain	48	14	(2) New Guinea
Bougainville	69	27	(2) New Guinea
Solomon Islands	110	-	
Australia (no location)	124	-	
Brewarrina	31	20	(3) NE Australia
Cherbourg	23	21	(3) NE Australia
Lake Tyers	14	7	(4) SE Australia
Koonibba	47	10	(5) Southern Australia
Point Pearce	43	12	(5) Southern Australia

Table 3.1: The number of mtDNA sequences from each sampling location, before and after filtering the data. The mtDNA sequences were then grouped geographically into six different populations and assigned a Population ID. Locations of populations are given in Figure 3.1.

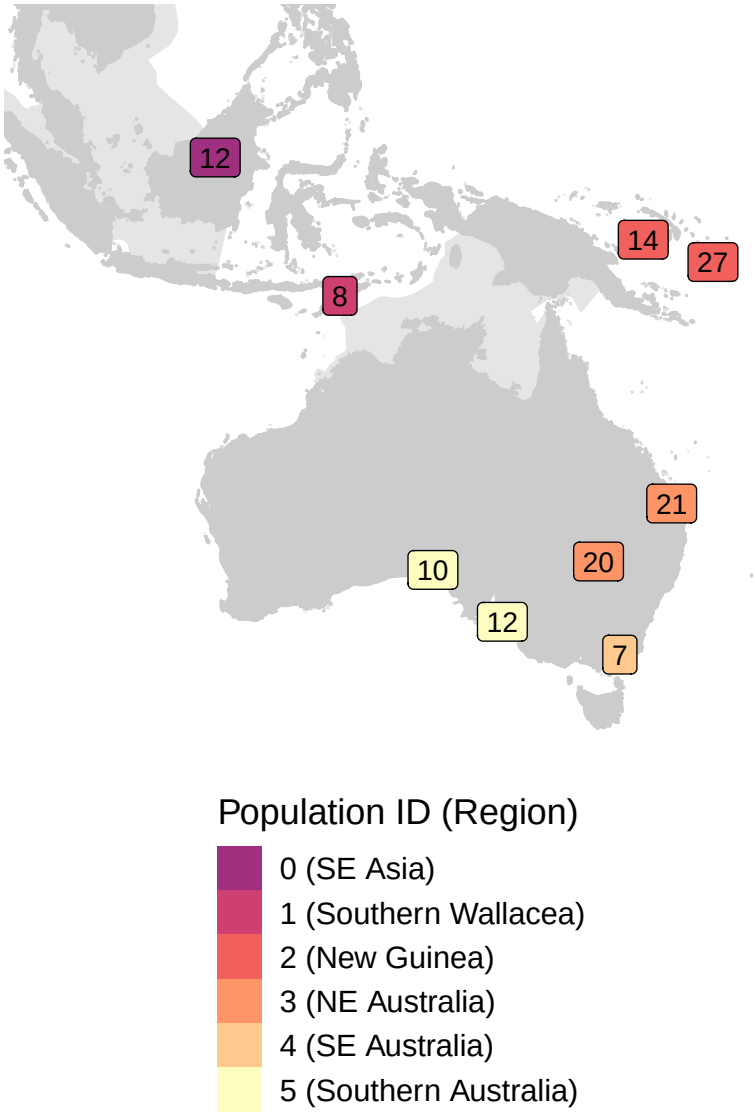


Figure 3.1: Number of sequences from each sampling location for the final dataset. Each colour represents a distinct geographical region; we defined these regions in Table 3.1. The number of sequences is given as a point at each location.

Next, we remove closely genetically related samples within populations, so that the final set of mtDNA sequences better represents the true population diversity. This filtering step is completed with the full mtDNA sequences, *i.e.* including the control region. Recall that the control region contains the hypervariable regions, which have a higher substitution rate than the coding region. Related samples are conservatively defined as sequences from the same population that differ by two or less point mutations.

Finally, we must decide which sites of the aligned mtDNA sequences to include in the analysis. As discussed in Section 2.1, we only select sites that are in the coding region, due to the higher and more variable substitution rate in the control region. The coding region is from position 577 to position 16023 inclusive on the mitochondrial genome [28].

The final filtered dataset is comprised of 131 aligned mtDNA sequences, with the control region excluded for all sequences. These sequences include 43 AHP sequences previously published by Tobler *et al.* [85], 27 sequences from the Aboriginal Heritage Project that have not yet been published, and 61 additional previously published sequences [21, 29, 37], with further details given in Table 3.1. Accession numbers and references for sequence data are given in Table B.2 (see Appendix B).

As described in Section 2.2, phylogenetic trees are used to visualise the genetic relationships between different individuals. In this thesis, we will reconstruct a maximum likelihood tree using IQ-TREE, and then conduct a Bayesian phylogenetic analysis using BEAST. In this Bayesian analysis we will reconstruct phylogenetic trees and the effective population size over time. A theoretical discussion of each method will be given before the results are presented.

3.2 Maximum Likelihood Trees

3.2.1 Likelihood calculation and algorithm details

Consider a phylogenetic tree having the same form as the tree given in Figure 2.3 of Section 2.2. We describe likelihood calculations in detail using an artificial example with the phylogenetic tree and mtDNA alignment in Figure 3.2.

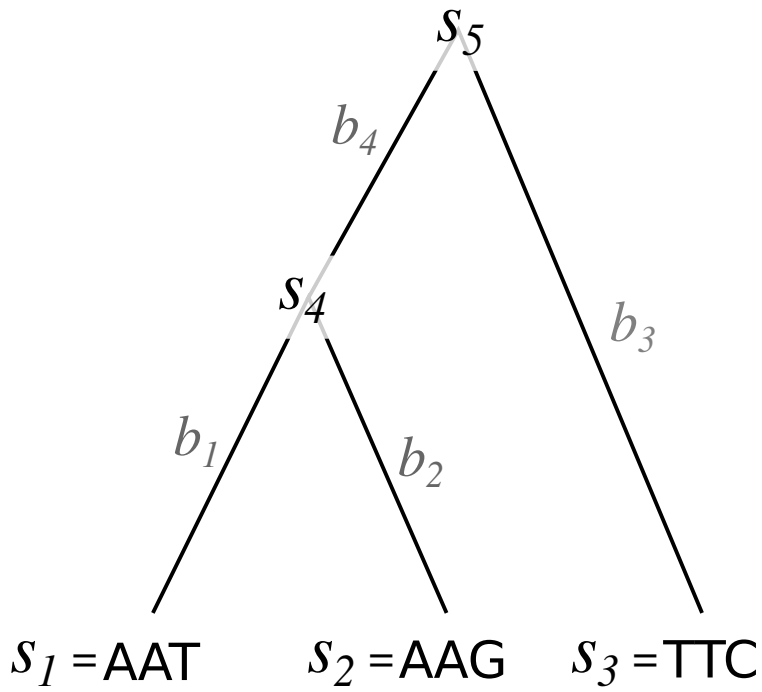


Figure 3.2: An example phylogenetic tree with sequences of length 3bp at the tips. Unknown sequences at internal nodes are labelled s_4 and s_5, and branch lengths are labelled b_1, b_2, b_3, and b_4.

The phylogenetic tree in Figure 3.2 has five nodes, including the tips. There are three sequences at the tips of the tree, denoted s_1, s_2, and s_3, and two unknown sequences at the internal nodes of the tree, denoted s_4 and s_5. We denote the i^{th} site of the j^{th} sequence as s_{ij}, *e.g.* $s_{11} = s_{12}$ =A, s_{13} =T, and s_{23} =G.

To calculate the likelihood of the aligned sequence data given the tree, we will first consider the first site of each sequence. When calculating probabilities from maximum likelihood trees, we can consider the substitution process forward in time or backward in time. These two scenarios are equivalent when the substitution model used is reversible, which is always the case for our analysis. In the following example, we consider substitutions backward in time.

We wish to determine the probability of observing the bases A, A, and T at s_{11}, s_{21}, and s_{31} respectively. The sites s_{41} and s_{51} are unknown, but it is clear that at least one substitution has occurred because not all of the bases

at the tips are the same. From s_{11}, we need to determine the probability that an A changes to a different base or remains the same in time b_1. From s_{21}, we need to determine the probability that an A changes to a different base or remains the same after time b_2. From s_{31}, we need to determine the probability that a T changes to a different base or remains the same after time b_3. Finally, we also need to find the probability that s_{41} either remains the same or changes to a different base in time b_4.

Assuming that substitutions occur according to a Poisson process (as described in Section 2.4.5), we can calculate the probability of observing different types of substitutions after some amount of time from a substitution model. Substitution models define the rate at which each type of substitution occurs (see Appendix A.2 for details of different models).

Since the substitution process occurs independently along branches, we multiply all of the probabilities together. We also apply the law of total probability to account for the unknown sites s_{41} and s_{51}, allowing the sites to be any base in the set $D = \{A, C, G, T\}$. Let the probability that the base s_{ij} changes to or remains as the base s_{kj} in time b as $P_{s_{ij}s_{kj}}(b)$. The likelihood for the first site is then

$$\begin{aligned} L_1 &= \sum_{s_{51}\in D}\sum_{s_{41}\in D} P_{s_{51}s_{41}}(b_4)P_{s_{51}T}(b_3)P_{s_{41}A}(b_2)P_{s_{41}A}(b_1) \\ &= \sum_{s_{51}\in D} P_{s_{51}T}(b_3)\sum_{s_{41}\in D} P_{s_{51}s_{41}}(b_4)P_{s_{41}A}(b_2)P_{s_{41}A}(b_1). \end{aligned}$$

A general formula for the i^{th} site likelihood of the tree in Figure 3.2 is

$$L_i = \sum_{s_{5i}\in D} P_{s_{5i}s_{3i}}(b_3)\sum_{s_{4i}\in D} P_{s_{5i}s_{4i}}(b_4)P_{s_{4i}s_{2i}}(b_2)P_{s_{4i}s_{1i}}(b_1). \tag{3.1}$$

The tree likelihood is calculated by multiplying all site likelihoods. Assuming independence between sites, the general form of the tree likelihood is

$$L_{TREE} = \prod_{i=1}^{n} L_i. \tag{3.2}$$

For our example in Figure 3.2, $n = 3$, but it is easy to see how likelihood calculations can be extended to sequences of any length n.

The likelihoods for each site in the alignment can be quite small, and so calculating the log-likelihood often makes more sense. The log-likelihood of

the tree is found by summing the log-likelihood calculated for each site. If the sequences have length n, then the log-likelihood for the tree is ℓ_{TREE} where

$$\ell_{TREE} = \sum_{i=1}^{n} \log(L_i). \tag{3.3}$$

Here, we assumed a substitution model, fixed branch lengths and a tree topology were all known in advance. In practice, the substitution model, branch lengths, and tree topology must all be determined before the tree log-likelihood can be calculated. We now consider how these may be found.

IQ-TREE allows substitution models to either be specified or estimated from the data. Models may be specified if they have been previously determined using model selection software such as ModelGenerator [42]. Substitution models are estimated from the data through the inbuilt program ModelFinder Plus. This program also finds the best-fitting substitution model which corresponds to the lowest BIC, although it is also possible to select a substitution model by AIC or corrected AIC.

After selecting a substitution model, branch lengths can be estimated. This is an optimisation problem where we wish to identify the branch lengths b_i that yield the maximum value of the likelihood function, stated fully in Equations 3.1 and 3.3. Yang [97] gives a detailed description of estimating branch lengths from sequence data in practice.

In the previous paragraphs, we defined the tree log-likelihood function and described the selection of substitution models and branch lengths for a fixed tree topology. Now, we consider how tree space can be searched to find a tree topology that maximises the log-likelihood. IQ-TREE attempts to identify an optimal tree topology using an algorithm that employs nearest neighbour interchanges (NNIs).

We illustrate the concept of NNI through unrooted trees. So far, we have only considered rooted trees, which have an ancestral node (the MRCA). Unrooted trees do not have this ancestral node, or a clear concept of 'time'. A comparison of rooted and unrooted trees is given in Figure 3.3. Maximum likelihood trees are unrooted by default, but a rooted maximum likelihood tree can be produced by including an outgroup that is not closely related to any of the sequences in the tree reconstruction.

Nearest neighbour interchange works by interchanging 'neighbouring' clades around a particular branch. Around any given branch there are three distinct

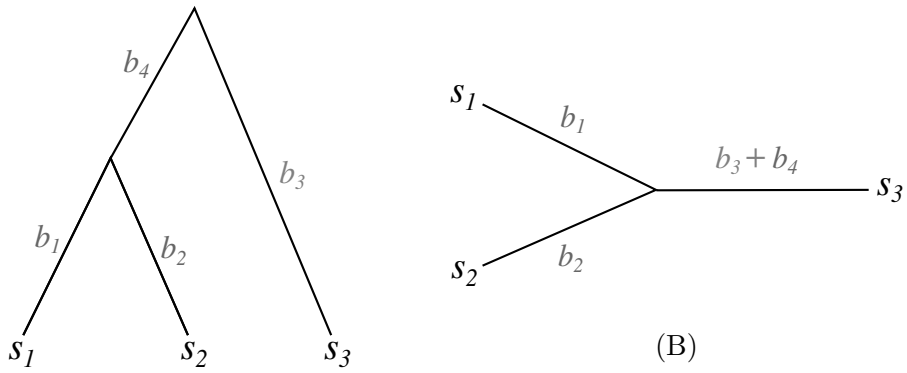


Figure 3.3: (A) Rooted and (B) unrooted forms of the same phylogenetic tree.

topologies, given in Figure 3.4. The branch of interest is in red, and the four subtrees are N_1, N_2, N_3 and N_4.

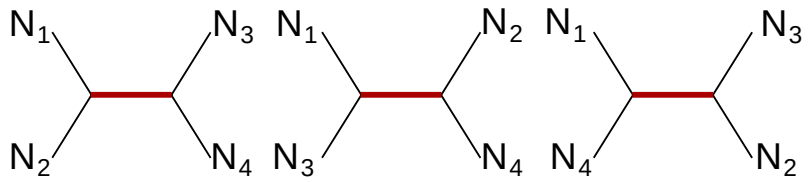


Figure 3.4: Nearest neighbour interchanges around a particular branch (red) of an unrooted tree. N_1, N_2, N_3 and N_4 are subtrees.

Given an initial topology, subsequent applications of NNI are used to find the topology that maximises the log-likelihood. Initial tree topologies may be random, or determined using a simpler algorithm. First, a hill-climbing algorithm by Guindon *et al.* [31] is applied, but this may remain in a local optima instead of finding the global optimum. Stochastic NNI decreases the likelihood of getting stuck in local optima, because it randomly changes the tree topology in a way that may not improve the log-likelihood. This then provides a different starting point for the hill-climbing NNI algorithm. Furthermore, this process is applied to a set of candidate trees, and then the tree with the highest log-likelihood in the final set is returned as the maximum likelihood tree [60].

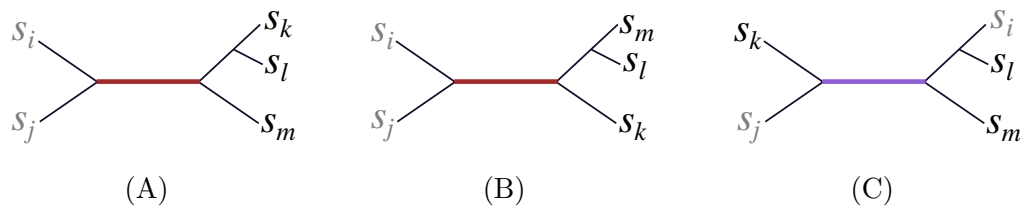


Figure 3.5: Examples of trees that have the same and different subclades as Tree (A). The two different subclades in Tree (A) are different colours (grey and black), and the branch of interest that separates subclades is red in Trees (A) and (B). The branch from Trees (A) and (B) does not appear in Tree (C), and so we use a different colour in Tree (C). The taxa in the clades of Tree (A), ((s_i, s_j), (s_k, s_l, s_m)), are clearly not the same as the taxa in each clade of Tree (C), ((s_j, s_k), (s_i, s_l, s_m)), and there is no branch that we can select in Tree (C) to recover the same subclades as in Tree (A).

3.2.2 Branch support tests

Branch support tests are used to quantify the reliability and fit of a maximum likelihood tree. IQ-TREE supports two main tests: the ultrafast bootstrap and the Shimodaira-Hasegawa approximate likelihood ratio test.

Ultrafast Bootstrap (uBS)

Ultrafast bootstrap employs a resampling technique where each new bootstrap sample is obtained by sampling sites with replacement from the sequence alignment. A maximum likelihood tree is built using only the resampled sites [54]. The uBS process is parameterised by the number of bootstrap replicates, *i.e.* the number of times that sites are resampled. The branch support value is the percentage of trees constructed from the bootstrapped samples that contain the same subclades as the original tree. Examples of trees that contain the same subclades and different subclades are given in Figure 3.5. Note that subclades need to contain the same set of tips, but are not required to have the exact same topology.

Shimodaira-Hasegawa approximate Likelihood Ratio Test

Classical likelihood ratio tests formally test whether a particular branch length in a tree is significantly greater than zero. The approximate likelihood

ratio test (aLRT) tests whether a particular branch significantly contributes to the likelihood by comparing the maximum likelihood to the likelihood of the second-best tree around that branch. The SH-aLRT borrows elements of the Shimodaira-Hasegawa test and applies them to the aLRT [30].

For each branch, the SH-aLRT algorithm considers the three distinct trees found through nearest neighbour interchange (NNI), which slightly modifies the tree topology to create different trees. A full description of NNI is given in Section 3.2.1.

Of the three distinct trees, one tree is the maximum likelihood tree. We denote the maximum likelihood tree T_1, with corresponding log-likelihood ℓ_1. The set of individual site likelihoods are denoted ℓ_1^s. The tree with the second greatest log-likelihood is denoted T_2, with corresponding log-likelihood ℓ_2, and the remaining tree is denoted T_3 with corresponding log-likelihood ℓ_3. We define the complete set of trees as $T_t = \{T_1, T_2, T_3\}$.

The null hypothesis of the SH-aLRT test is that all three trees T_1, T_2, and T_3 resulting from NNI explain the data equally well. This should not be the case if the branch of interest in the maximum likelihood tree is well-supported. The alternative hypothesis of the Shimodaira-Hasegawa (SH) test is: "Some or all of T_1, T_2 and T_3 are not equally good explanations of the data". Combining the general SH test with the aLRT, high branch support suggests that the maximum likelihood tree T_1 is a better explanation for the data than the trees that would result from NNI around the branch of interest.

Let the set of all branches in a tree be $B = \{b_1, b_2, \ldots, b_k\}$. As in Figure 3.2, the j^{th} site of the i^{th} sequence is denoted s_{ij}, for $j = 1, 2, \ldots, n$. To refer to the j^{th} site of all sequences, we use $s_{\bullet j}$. We define the number of bootstrap replicates as $r \in \mathbb{Z}^+$. The SH-aLRT algorithm is formally defined

in Algorithm 1, which calculates the branch support value V.

Algorithm 1: SH-aLRT

Set indicator $v = 0$
for $b_t \in B$ **do**
  Find T' and T'' through NNI around b_t of T_1
  for $t \in \{T', T''\}$ **do**
    for j *in* $\{1, 2, \ldots, n\}$ **do**
      Calculate the log-likelihoods for each site of the sequences $s_{\bullet j}$, denoted ℓ_t^j.
    end
    Calculate the tree log-likelihood, $\ell_t = \sum_j \ell_t^j$.
  end
  Denote the tree with the second greatest log-likelihood T_2, and the remaining tree T_3.
  for i *in* $\{1, 2, \ldots, r\}$ **do**
    for t *in* $\{1, 2, 3\}$ **do**
      Draw n samples from $\{\ell_t^j \mid j = 1, 2, \ldots, \text{n}\}$. We denote the k^{th} sample as ℓ_{t*}^k.
      Calculate $\ell_{t*} = \sum_k \ell_{t*}^k$
      Calculate the centered sum, $C_t = \ell_{t*} - \ell_t$.
    end
    Define the two largest centered sums as C_h and C_m.
    if $2(\ell_1 - \ell_2) > 2(C_h - C_m) + \epsilon$ **then**
      $v = v + 1$
    end
  end
  $V = v/r$
end

Note that the expected value for all centered sums is zero. To understand why branch support is incremented, let us consider what it means for the condition $2(\ell_1 - \ell_2) > 2(C_h - C_m) + \epsilon$ to be rejected. The ϵ in the condition makes it unlikely to increase the branch support value V when T_1 and T_2 have similar likelihoods and nearly identical site log-likelihood values [30]. Similar site log-likelihoods mean that the sum of the resampled log-likelihoods, ℓ_{t*}, should be close to the tree log-likelihood ℓ_t, and therefore the centered sums should be small. If the likelihoods ℓ_1 and ℓ_2 are also similar, then both sides of the condition would be very small. Alternatively, if ℓ_1 is significantly larger than ℓ_2, the condition is likely to be met.

For reliable branch support values, the authors of the SH-aLRT recommend $r \geq 1000$ and $\epsilon = 0.1$ [30].

According to guidelines from IQ-TREE, a branch is well supported if the uBS support is at least 95% and the SH-aLRT support is at least 80%.

3.2.3 Results

A maximum likelihood tree of the filtered sequences was produced using IQ-TREE version 1.6.10 [60]. Five complete or near-complete Neanderthal mitogenomes [11] were included as an outgroup to reliably root the tree; accession numbers for these sequences are given in Appendix B.2. We aligned the Neanderthal sequences to the revised Cambridge reference sequence (rCRS) using BLAST version 2.10.1 [3]. The Neanderthal sequences were removed after the maximum likelihood tree has been reconstructed, as they are not sequences of interest.

IQ-TREE was implemented with seed 808266 and automatic substitution model selection, which selected a Tamura-Nei substitution model with invariant sites, empirical base frequencies, and a FreeRate model with two classes.

The Tamura-Nei substitution model allows different rates for the two types of transitions, *i.e.* transitions between purines and transitions between pyrimidines, and then another different rate for all transversions. The Tamura-Nei model with empirical base frequencies also takes into account the frequency of each base in the sequence data. This substitution model is defined mathematically in Appendix A.2.5. Invariant sites are sites that do not undergo substitution, which is biologically reasonable for regions of the genome that perform critical functions. The proportion of invariant sites is estimated by IQ-TREE. The FreeRate model can be thought of as an extension to the gamma rate variation between sites described earlier in Section 2.4.5; both the FreeRate model and gamma rate variation allow different substitution rates at different sites. More information on the FreeRate model is given by Soubrier *et al.* [78].

To assess internal node support, the ultrafast bootstrap (uBS) [54] and the Shimodaira-Hasegawa approximate Likelihood Ratio Test (SH-aLRT) [30] were both performed with 1000 replicates.

The resulting maximum likelihood tree with branch support values is presented in Figure 3.6. This tree displays the topology only, and so the branch

lengths have no meaning. We display the tree in this way to better visualise the tree topology and provide readable branch support values. The maximum likelihood tree with branch lengths is then presented in Figure 3.7. Both of these trees are labelled according to the major mtDNA haplogroups of the sequences. Different mtDNA haplogroups are defined by different sets of mutations that occur in mtDNA. For further information, we refer the reader to PhyloTree [86], which describes the genetic history of different mtDNA haplogroups.

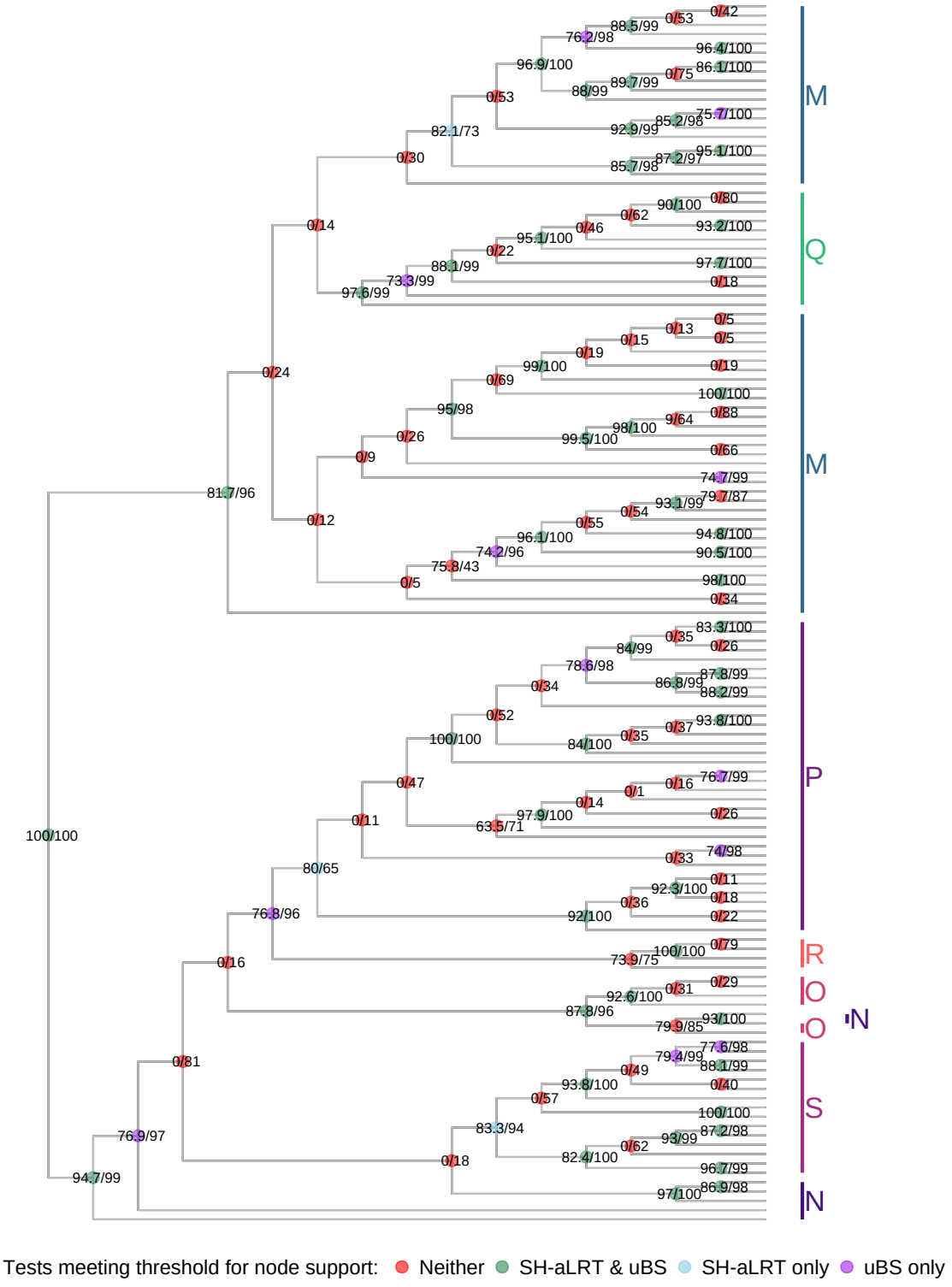


Figure 3.6: The topology of the maximum likelihood tree for mtDNA sequences in the filtered data; the filtering process was described in Section 3.1.1. Support values are given at the end of each branch in the form SH-aLRT(%)/uBS(%), and points are coloured according to the tests that support the branch. Major mtDNA haplogroups are also labelled. This tree assumes a Tamura-Nei substitution model.

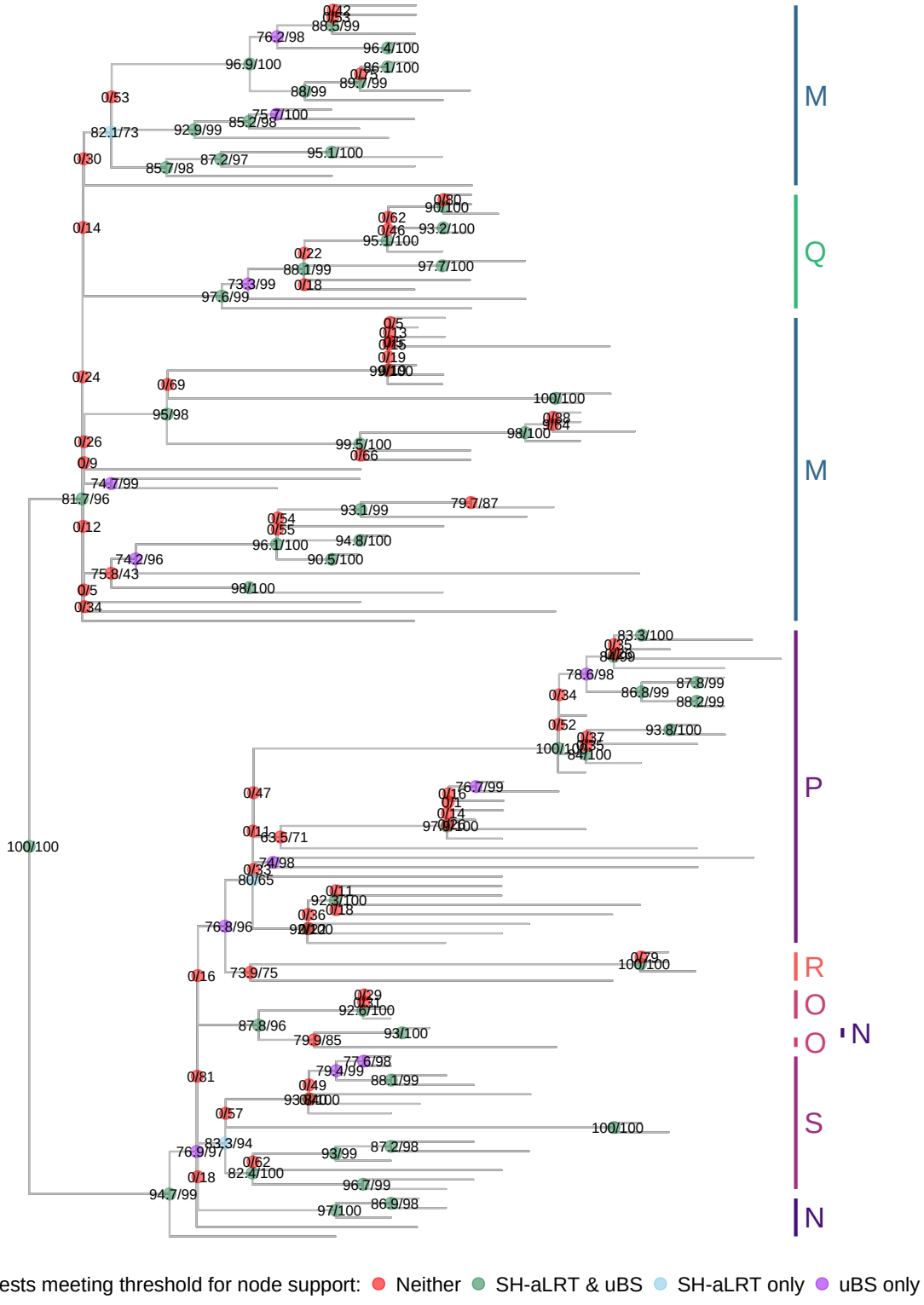


Figure 3.7: The maximum likelihood tree with branch lengths for mtDNA sequences in the filtered data. Support values are given at the end of each branch in the form SH-aLRT(%)/uBS(%), and points are coloured according to the tests that support the branch. Major mtDNA haplogroups are also labelled. This tree assumes a Tamura-Nei substitution model.

3.3 BEAST Trees

3.3.1 Methods

Bayesian evolutionary analysis by sampling trees (BEAST) is another commonly-used program to construct 'calibrated' phylogenetic trees [8]. Calibrated trees can have branch lengths in calendar years, which allows for the estimation of divergence times. BEAST v2.6.0 and all associated packages (BEAUti, TreeAnnotator, LogCombiner) were used for this analysis; all references to BEAST refer to the implementation in BEAST2.

Sampling trees using MCMC

Let D denote the aligned mtDNA sequence data, and T denote the phylogenetic tree and its corresponding parameters: branch lengths, population demographic history, and the substitution model. We can find the probability of the tree given the data by applying Bayes' theorem,

$$P(T \mid D) = \frac{P(D|T)P(T)}{P(D)}. \tag{3.4}$$

BEAST uses Markov chain Monte Carlo (MCMC) to sample trees from the posterior distribution $P(tree \mid data)$ (see Equation 3.4).

We illustrate how trees are sampled in MCMC by defining the Metropolis algorithm in Algorithm 2. The Metropolis algorithm is a special case of the well-known Metropolis-Hastings algorithm, with the assumption of symmetric proposal distributions. This is a simple example given to illustrate how trees could be sampled. In practice, BEAST allows for the use of more recently developed MCMC algorithms that are more efficient or better-suited for post-hoc analysis [19].

In Algorithm 2, the sequence of sampled trees is denoted $\boldsymbol{T}$, and it contains k trees. The initial starting tree is denoted T_1, and is supplied to the algorithm.

We denote the chain length by m.

Algorithm 2: Metropolis MCMC for sampling trees

1 $\boldsymbol{T} = \{T_1\}$;
2 $k = 1$
3 **for** i *in* $\{1, 2, \ldots, m\}$ **do**
4 | Sample T_i from the proposal distribution.
5 | Calculate the log-likelihoods ℓ_i and ℓ_k for the trees T_i and T_k.
6 | Calculate the ratio $R = \ell_i P(T_i)/\ell_k P(T_k)$.
7 | $\boldsymbol{T} = \{\boldsymbol{T}, T_i\}$, $k = k + 1$ with probability $\min\{1, R\}$.
8 **end**

At Step 4, a new tree is proposed based on the proposal distribution, which is defined by operators. Some operators are probability distributions, *e.g.* the operators relating to population sizes, or substitution rate. Other operators change the tree topology in different ways. Each operator also has a weight, and operators with a larger weight change more frequently. To propose a new tree, an operator is chosen according to its weight, and then the tree is modified by changing this operator.

The changes due to the operators are random, and depend only on the current state of the tree and its associated parameters. This behaviour illustrates the stochastic and memoryless properties of a Markov chain: stochasticity due to the randomness of the change in operators, and memorylessness because the next state depends only on the current state, and not on the trees or parameters previously sampled.

The formulation of MCMC is such that the posterior distribution that we are interested in is the stationary distribution of the Markov chain. Since it may take some time to reach the stationary distribution, 'burn-in' must be specified. This is a percentage of initial samples that are discarded.

Monte Carlo methods rely on sampling from the posterior distribution, and so they will not be exactly correct unless the chain has an infinite number of steps. However, given a sufficiently large number of samples, the method will result in a representative sample from the posterior distribution of the parameters of interest. It can be difficult to pre-specify a sufficient number of steps, and so after the sampling process is completed it is important to assess whether or not enough posterior samples have been obtained.

Convergence of MCMC chains

To check that a MCMC chain was run for long enough, we inspect trace plots and effective sample sizes, both of which can be obtained through Tracer [71]. Trace plots are a line plot of the sampled values over time. For convergence, it is recommended that these plots look like 'fuzzy caterpillars', *i.e.* showing no trend and having similar variance along the chain [19]. The amount of burn-in should be determined from the trace plot, because the default 10% burn-in is sometimes not sufficient for the chain to reach convergence.

We can also assess the convergence of multiple MCMC chains using the $\widehat{R}$ statistic, which was originally suggested by Gelman and Rubin and then later improved by Vehtari *et al.* [88]. The $\widehat{R}$ statistic compares the within-chain variance to the total variance of all chains, and also compares the first half of the chain to the second half of the chain. If the set of chains shows poor mixing or lack of convergence, then we expect the posterior of the combined chains to look different to the posterior of each chain individually, which can result in larger variance in the combined chain. Furthermore, if the first half of the chain is dissimilar to the second half of the chain, it is unlikely that the chain has converged. The $\widehat{R}$ statistic is constructed so that $\widehat{R} > 1$, with values closer to one suggesting better mixing of the chain and little evidence for non-convergence. Initially Gelman and Rubin suggested $\widehat{R} < 1.1$ as a threshold for using the posterior samples of MCMC chains, but recently Vehtari *et al.* have suggested a threshold of 1.01 [88].

Since new trees are proposed based on the current tree in an MCMC chain, the samples are not independent. This means that we cannot use the unadjusted sample size for each parameter when inspecting how well the MCMC chain has sampled the posterior distribution. Instead of considering the actual posterior sample size, we may be interested in the relative number of 'independent' samples from the posterior distribution. One measure of this is the effective sample size (ESS), which accounts for the correlation of sampled parameter values. Drummond *et al.* [19] suggest that all parameters should have an ESS above 200 for sampling to be considered adequate.

3.3.2 BEAST specification

The BEAST analysis is specified using the application 'BEAUti', which is part of the BEAST package. We will first present some preliminary results from ModelGenerator, which was used to select a substitution model and

associated parameters. We then describe all parameters required to specify a BEAST analysis through BEAUti.

ModelGenerator [42] was used to select the optimal substitution model, as decided by the BIC (Bayesian Information Criterion) value. This substitution model, as well as other parameters of the substitution process that were estimated from the data, are given in Table 3.2. These parameters will be defined under the 'site model' and 'clock model' parts of the BEAUti specification.

Parameter	Value/Type
Substitution Model	HKY
Parameter κ for HKY model	38.73
Substitution Rate	1.57×10^{-8} [28]
Number of Gamma Rate Classes	7
Gamma Shape Parameter α	0.53
Invariant Sites	0.81

Table 3.2: Parameters determined from the mtDNA data by ModelGenerator.

When using BEAUti to specify a BEAST analysis, the parameters are divided into the sections 'site model', 'clock model', 'priors', and 'MCMC'. We consider each of these sections in turn and describe them in the following paragraphs.

First, we consider the site model, which defines how substitutions occur at each site. We use the HKY substitution model, since it had the lowest BIC value in the ModelGenerator output. BEAST allows for a proportion of invariant sites (sites that do not accumulate substitutions), which was determined to be 0.81 from the data. Different substitution rates for different sites were modeled with a discrete gamma distribution, parameterised by $n = 7$ rate categories and a shape parameter $\alpha = 0.53$.

When specifying the substitution model for the BEAST analysis, we use empirical base frequencies, *i.e.* the base frequencies observed in the mtDNA alignment.

We now consider the prior distributions for all parameters. If we do not have cause to change any prior distributions, the default distributions will be used.

We change the tree prior and associated parameters, which defines the underlying model of the phylogenetic tree. For example, different tree priors allow for different theoretical foundations (the coalescent model or a birth-death process) and different assumptions of population growth (constant population size, exponential growth, or no assumption). The Coalescent Extended Bayesian Skyline tree prior is used to accommodate a complex demographic history. Parametric models are limited to constant or exponential growth of the effective population size through history, which can be too restrictive. The Extended Bayesian Skyline, which will be described in detail in Section 3.5, simultaneously estimates the tree and the effective population size throughout history. The tree prior has an additional parameter, 'Population Model', which allows the analysis of different types of DNA data in a single MCMC analysis. Since we are analysing only mtDNA, this should be set to 0.5.

We now consider the clock model, which defines the time-scale of the tree, and also whether different rates are allowed in different branches of the tree. We set the mean clock rate as the substitution rate of 1.57×10^{-8} substitutions per site per year to reconstruct a tree where branch lengths are in calendar years. When one clock rate is enforced for the entire tree, a 'strict clock' is used, and the single clock rate is set as the mean clock rate. In contrast, a 'relaxed clock' allows different rates for different branches. The mean of the distribution that the clock rates are drawn from is set to the mean clock rate.

An initial BEAST analysis with chain length 1×10^7 was performed assuming a relaxed clock to assess whether a strict clock was adequate, or whether a relaxed clock was required. All other model parameters were as defined in previous paragraphs.

A relaxed log-normal clock draws the different clock rates for different branches from a log-normal distribution. We consider the distribution of S, the standard deviation of the log-normal distribution of clock rates, to assess whether a relaxed clock is needed. The default prior distribution for this parameter is a gamma distribution with 50% of the mass below 0.1. We reject the possibility of a strict clock if the estimate of S is greater than 0.1 and there is no probability mass near zero when inspecting the marginal probability distribution [19].

The point estimate of S was 0.0997, with a 95% highest posterior density (HPD) interval of $(8.92\times10^{-10}, 0.261)$. The 95% HPD interval is the narrowest interval that contains 95% of the posterior density. The point estimate of S is very close to 0.1, so we cannot confidently use a strict clock. The wide

95% HPD interval further contributes to this uncertainty. Hence, we use a relaxed log-normal clock. The mean clock rate was set to the substitution rate of 1.57×10^{-8}, so that the branch lengths of the tree are in calendar years [19].

For the MCMC specification, we use a chain length of 10^8, and implement thinning by retaining one sample every 10^4 samples. This is according to recommendations by Drummond *et al.* [19], who suggest retaining no more than a maximum of 10^4 samples.

The number of chains is not specified in BEAUti, but is instead the number of times that the BEAST analysis is run. We will use three MCMC chains. This provides another simple test for convergence, because if all chains have converged, the marginal densities of the posterior samples should be very similar.

3.3.3 Results

Tracer version 1.7.1 [71] was used to analyse and visualise the posterior samples of three separate BEAST analyses. The three MCMC chains were combined using LogCombiner, which is a part of the BEAST package. All post-processing is described under Section 3.5.2, since we will also discuss parameters directly related to the skyline plot.

The final set of combined posterior samples contained 13294 trees. To visualise the set of sampled trees, we can use the DensiTree software, which superimposes all sampled trees in one figure [9]. The DensiTree results are displayed in Figure C.1, Appendix C.1. It is difficult to determine any measures of branch support by simply inspecting the set of trees, and so an annotated summary tree is used to better describe the set of sampled trees. We use TreeAnnotator, which is part of the BEAST package, to obtain this summary tree [8].

Summary trees are often selected based on the posterior clade probabilities in a tree. For any given clade, the posterior clade probability is the proportion of trees in the posterior sample that contain the clade.

To summarise the set of trees in the posterior sample we use a maximum clade credibility (MCC) tree, which is the tree with the highest product of posterior clade probabilities. Although other types of summary trees are possible, such as the extended majority consensus tree (which produces a tree from the clades with the highest posterior clade probabilities) or the

tree with the highest sum of clade credibilities, we use the MCC tree because it is an actual tree that was sampled, and therefore has positive branch lengths.

After deciding on an appropriate summary tree, we can add annotations to this tree. Common annotations are the posterior clade probabilities for each clade, and either point estimates or 95% HPD intervals for node heights. In our analysis, the node heights are the times before the present day that lineages coalesce, because we have selected a coalescent tree prior and set the mean clock rate to the substitution rate. When using TreeAnnotator to create tree annotations, we also need to specify any burn-in, select a posterior probability limit, and decide how the node heights should be summarised.

The posterior probability limit restricts which nodes in the summary tree are annotated. If the posterior clade credibility is below the posterior probability limit, a point estimate or 95% HPD interval of node height is not calculated for that node. Note that posterior probability limits do not change the topology of the summary tree, and only affect the annotated information. We use a posterior probability limit of 0.1, because clades that appear in 10% or less of the trees in the posterior sample are unlikely to have reliable annotations.

We use common ancestor times (CAT) to construct 95% HPD intervals for node heights. For a given clade, the set of all common ancestor times comprises the MRCA time of the tips in that clade, as calculated for each tree. The 95% HPD interval is then calculated from this set of times. Based on a study by Heled *et al.*[33], using CAT with MCC trees recovers node heights more accurately than any other method available in TreeAnnotator, such as median heights or mean heights.

BEAST trees

We present two summary trees: a MCC tree with node heights defined by common ancestor times (Figure 3.8), and the same MCC tree annotated with the posterior probability of each clade (Figure 3.9).

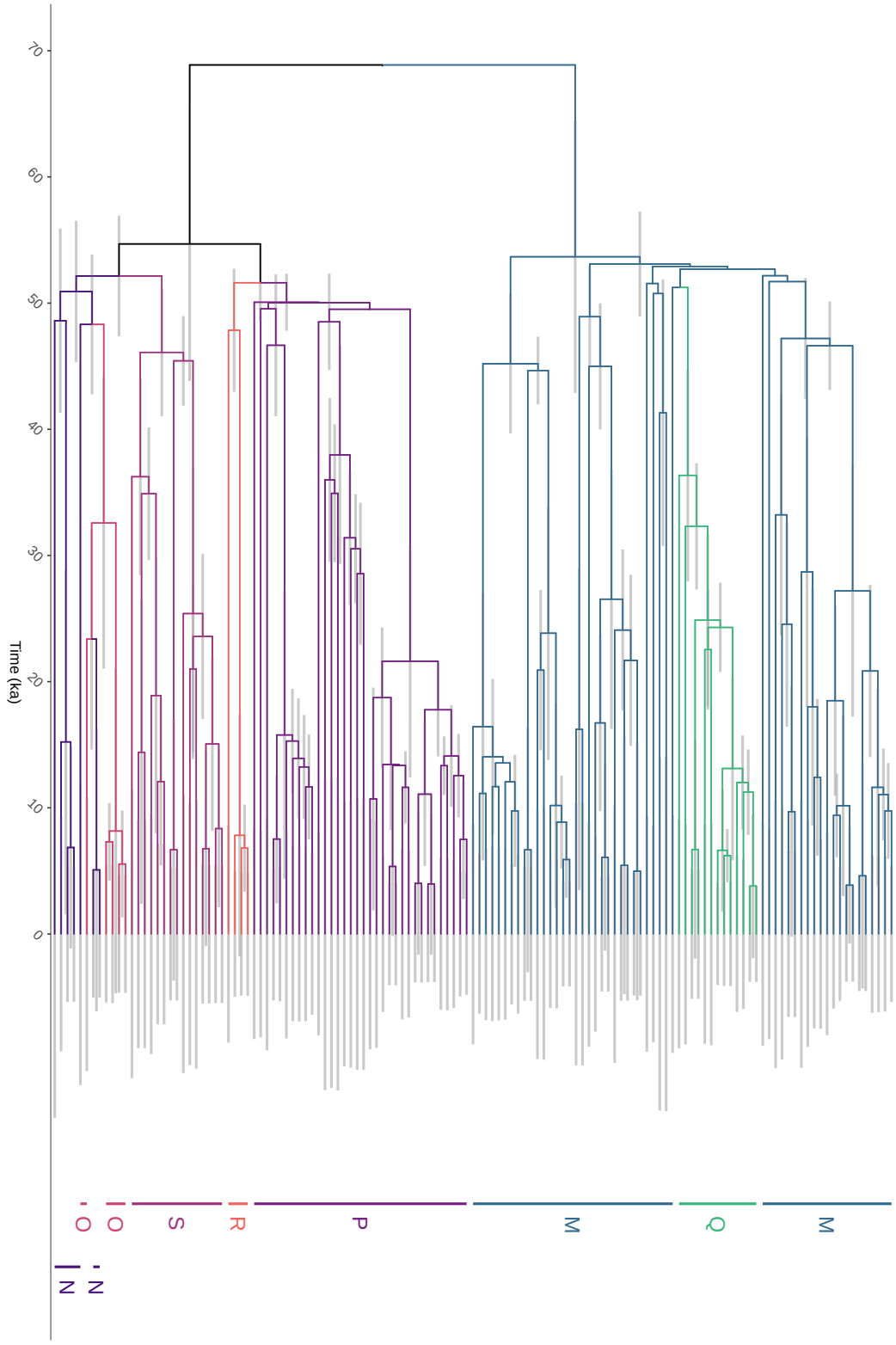


Figure 3.8: MCC tree with 95% HPD node intervals defined by common ancestor times. Each major mtDNA haplogroup has a distinct colour and label.

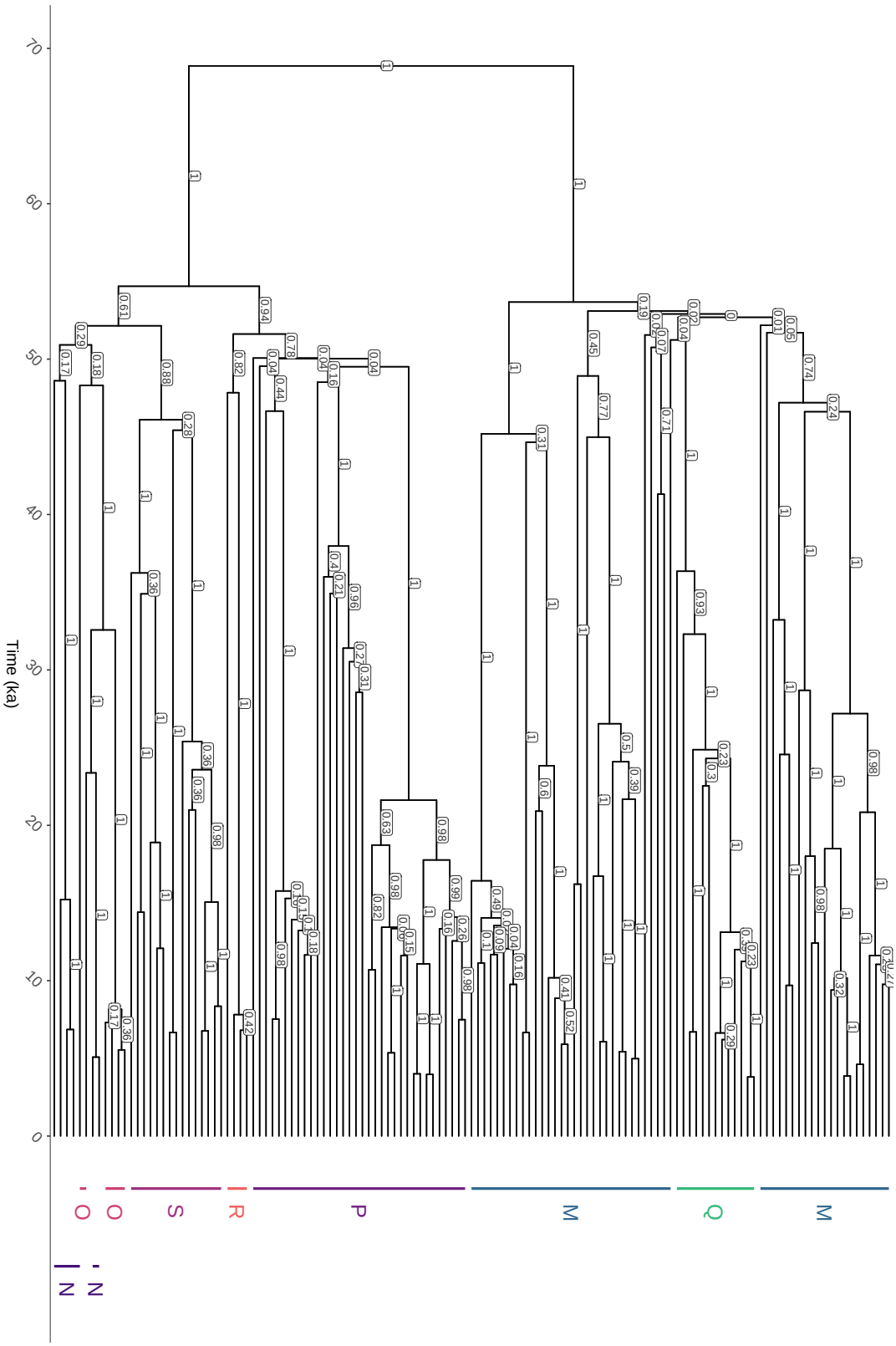


Figure 3.9: MCC tree with node heights according to common ancestor times. Each node is annotated with the corresponding posterior clade probability. Again, all major mtDNA haplogroups have a distinct colour and label.

3.4 Discussion of Tree Results

First, we discuss the maximum likelihood trees in Figures 3.6 and 3.7. We notice that the splits for major mtDNA haplogroups are often supported by either both tests or the SH-aLRT only. Within haplogroups, many branches are not supported by either tests. In particular, many branches have a score of 0 for uBS, meaning that the subclade appeared in no other trees in the sample. Branches with low support are also significantly shorter than well-supported branches.

Now, we consider the BEAST trees in Figures 3.8 and 3.9. Since we are interested in events that occurred thousands of years ago, we measure time using the *kiloannum* (ka), where 1 ka is one thousand years ago. The 95% HPD intervals for the splits of ancestral mtDNA haplogroups M and N are older than 50 ka, suggesting that M and N had split into other haplogroups at the time of the peopling of Sahul. The split between mtDNA haplogroups R and P is also dated to before 50 ka; haplogroups R and P have posterior probabilities of 0.82 and 0.78 respectively. This finding is consistent with the proposal by Tobler *et al.*, which is that the mtDNA haplogroups split soon after entering Australia, taking separate migration paths around the east and west coasts [85].

From Figure 3.9, we notice that monophyletic haplogroups tend to have high posterior probabilities, *e.g.* 0.78 for haplogroup P, 1 for macrohaplogroup M, 0.88 for haplogroup S. 'Monophyletic' means that all tips in a clade belong to the same haplogroup, and there are no other tips belonging to that haplogroup.

The topology of the BEAST trees are consistent with those of the maximum likelihood trees, in that many major mtDNA haplogroups are well-supported. The ancestry of each mtDNA haplogroup is consistent with the reference human mtDNA tree at PhyloTree [86], and the split between macrohaplogroups M and N is clearly defined. We note the short times between coalescent events from approximately 45 ka to 55 ka; these short branch lengths were seen in a similar region in the maximum likelihood tree in Figure 3.7. This again suggests that few substitutions are likely to have occurred in the rapid migration through the southeast Asian islands and into Australia, which is estimated to have occurred in this window. Due to long branch lengths within mtDNA haplogroups in both the maximum likelihood and BEAST trees, we also expect considerable differences between sequences within the same mtDNA haplogroup.

3.5 Extended Bayesian Skyline Plots

3.5.1 Methods

Before introducing Extended Bayesian Skyline Plots (EBSPs), we discuss the simpler classic skyline plots, upon which EBSPs are based. We then describe the many improvements that have been made to classic skyline plots, before defining the EBSP.

The classic skyline plot was introduced in Pybus *et al.* [68]. Given a coalescent tree, these plots produced an estimate of the effective population size over every coalescent interval (the time between two consecutive coalescent events). Figure 3.10(b) shows a classic skyline plot from Strimmer *et al.*. We notice that the 'skyline plot' resembles a city skyline, due to the very noisy, piecewise constant estimate of effective population size over time.

To reduce the stochastic noise present in classic skyline plots, Strimmer *et al.* introduced the generalized skyline plot. Instead of estimating the effective population size over every coalescent interval, multiple coalescent intervals were grouped together to form a smaller number of intervals. This grouping was defined by a parameter ϵ, where the grouped intervals were required to have a length greater than or equal to ϵ substitutions per site. As seen in Figure 3.10, estimating the effective population size over the grouped intervals has a significant smoothing effect on the plot and does not result in a significant loss of information, assuming that an unreasonably large value of ϵ is not chosen. The use of grouped intervals was also found to produce a better estimate of the effective population size over time than the classic skyline plot when there was weak phylogenetic signal in the data [80]. This can be seen intuitively by considering our reconstructed maximum likelihood tree in Figure 3.7: very short intervals between coalescent events occur when there is low branch support and the tree is not fully resolved, and so little information would be lost by grouping these short intervals with at least one other interval.

Both the classic and generalized skyline plots were post-hoc analyses that required a phylogenetic tree as input. Bayesian skyline plots remove the requirement for a finalized phylogenetic tree as input, because the effective population size and the phylogenetic tree are estimated concurrently.

In Bayesian Skyline Plots (BSPs), the number of intervals over which to estimate the effective population size is pre-specified [20]. Both the tree and the trajectory of effective population size over time are estimated through a

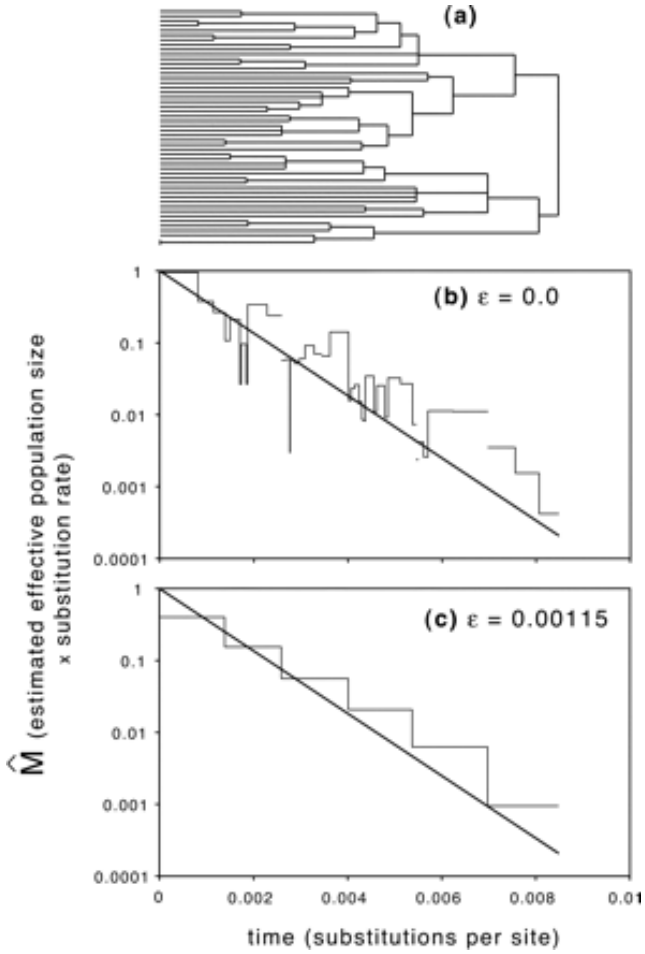


Figure 3.10: Skyline plots for DNA sequences simulated assuming an exponentially growing population: (a), estimated tree; (b), classic skyline plot ($\epsilon = 0$); and (c), generalized skyline plot ($AICc$ estimate of $\epsilon = 0.00115$). The thick line shows the true demographic history.
Originally published by K. Strimmer and O. Pybus, Exploring the Demographic History of DNA Sequences Using the Generalized Skyline Plot, *Molecular Biology and Evolution*, 2001, Vol. 18, Issue 12, pp. 22982305, by permission of Oxford University Press.

Monte-Carlo sampling process. This is beneficial as researchers are no longer required to specify some demographic history (*e.g.* constant, exponential growth, logistic growth) before reconstructing a phylogenetic tree to use as input for the skyline plots.

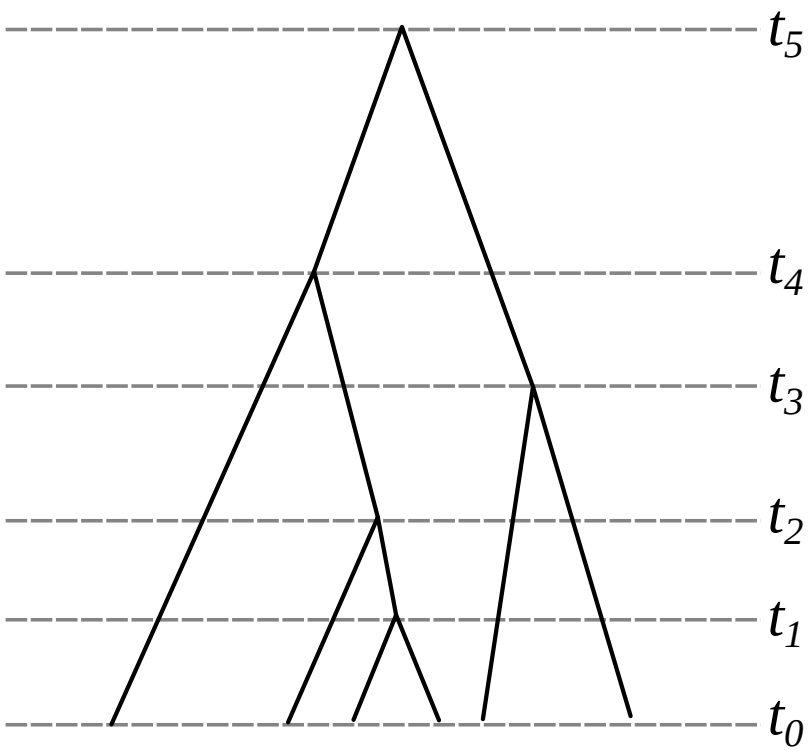


Figure 3.11: A tree with five coalescent events occurring at times t_1, t_2, t_3, t_4. and t_5.

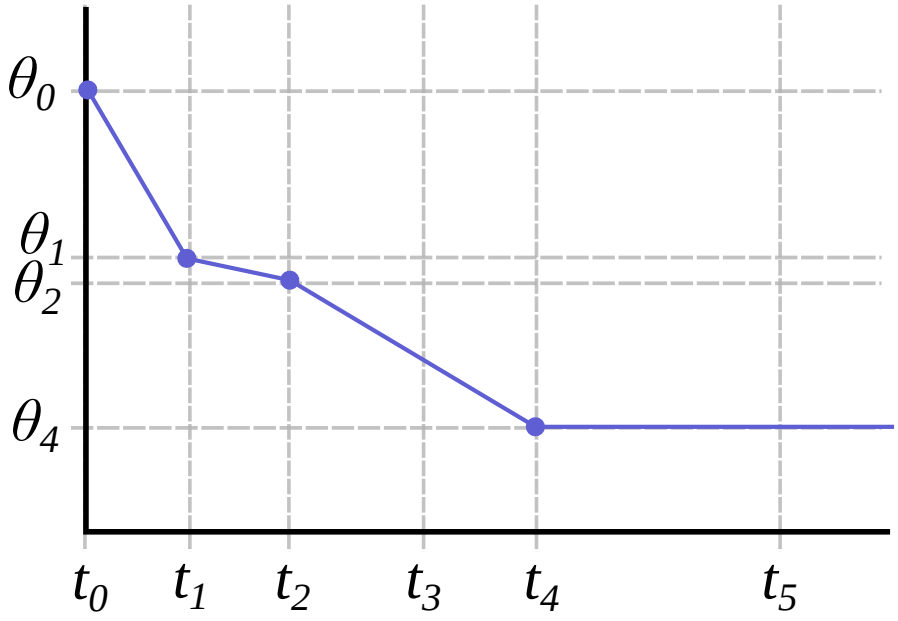


Figure 3.12: One sample of the Extended Bayesian Skyline. Only $\theta_0, \theta_1, \theta_2$, and θ_4 contribute to the effective population size over time, which corresponds to the indicator variables $I_0 = I_1 = I_2 = I_4 = 1$ and $I_3 = I_5 = 0$.

There is no way to definitively know how many intervals should be used in a BSP, so it may be necessary to try a range of values. Extended Bayesian Skyline analysis, as defined by Heled and Drummond [34], improves on previous types of skyline analyses by considering the number of times that the effective population size changes throughout history as a variable to be estimated. The effective population size over time is also constrained to be a piecewise linear function, instead of a piecewise constant function.

We illustrate the concept of EBSPs with a simple example. Suppose that a tree is sampled as in Figure 3.11; a trajectory of the effective population size over time is given in Figure 3.12. The EBSP is defined by a set of effective population sizes, $\boldsymbol{\theta} = \{\theta_0, \theta_1, \theta_2, \theta_3, \theta_4, \theta_5\}$, and a set of indicators, $\mathcal{I} = \{I_0 = 1, I_1, I_2, I_3, I_4, I_5\}$. The coalescent times of the tree in Figure 3.11 are denoted t_i, the variable θ_i is the estimated effective population size at the time t_i, and the variable I_i is an indicator that defines whether the population size θ_i contributes to the estimated effective population size over time.

We note that the indicator variables define the coalescent events that are the endpoints of the intervals over which the effective population size is estimated. As for generalized skyline plots and BSPs, intervals are grouped to obtain a smoother estimate of effective population size over time while discarding minimal signal from the data. The grouped intervals are those that individually have weak signal.

As illustrated in Figure 3.12, the estimated effective population size at the present day is always used to define the EBSP (*i.e.* $I_0 = 1$), and I_i is either 0 or 1 for $i = 1, 2, \ldots, 5$. Both $\boldsymbol{\theta}$ and $\mathcal{I}$ are sampled during the MCMC process according to their proposal distributions, and are not estimated from the tree.

Figure 3.12 shows how the estimated effective population sizes $\boldsymbol{\theta}$ and the indicators $\mathcal{I}$ are used to construct the estimated effective population size over time. The effective population size over time is a piecewise linear function; the effective population size is held constant after the oldest estimate of effective population size. In Figure 3.12, we can see that the effective population size remains at θ_4 after time t_4 because the later indicator variable I_5 is zero. If all indicator variables are zero, the EBSP corresponds to a constant effective population size over time.

In addition to the variables already defined, we denote the distribution of the mean population size as ϕ, the tree as T, the substitution rate as μ, and the sequence data as D.

The posterior distribution of an EBSP analysis in BEAST is then

$$f(\boldsymbol{\theta}, \mathcal{I}, T, \phi, \mu \mid D) \propto f_D(D|T, \mu) f_T(T|\boldsymbol{\theta}, \mathcal{I}) f_{\boldsymbol{\theta}}(\boldsymbol{\theta}, \phi) f_{\mathcal{I}}(\mathcal{I}) f_\mu(\mu). \tag{3.5}$$

The final three terms in Equation 3.5 are the prior probabilities of the variables $\boldsymbol{\theta}, \phi, \mathcal{I}$, and μ, while the first two terms on the RHS are the conditional probabilities of the sequence data D given the tree and substitution rate, and of the tree given the population sizes and indicators [34].

A high posterior probability therefore requires the data to be likely given the tree and substitution rate, and the tree to be likely given the population sizes and indicators that define the effective population size over time. We note that Heled *et al.* also extend the theory of skyline plots to multiple loci, which is not necessarily equivalent to multiple individual sites. A locus refers to the location of a gene on the genome [22].

The path shown in Figure 3.12 would correspond to one MCMC sample from the posterior distribution; different samples will have different sets of indicator variables and different effective population size estimates. This gives us a set of effective population size paths over time. These paths can then be aggregated to produce a 95% highest posterior density (HPD) interval and a median estimate of effective population size over time, which are then presented in the EBSP.

For more details on EBSPs we refer the reader to Heled and Drummond [34]; for more information on how EBSPs and trees are estimated using MCMC sampling methods, we again refer the reader to Drummond *et al.* [19].

3.5.2 Results

Three MCMC chains were initialized with a length 10^8 states. Analysis was performed on a HPC cluster; in ensuring that all chains reached a length of 10^8 states, the final length of all chains exceeded 10^8 states. Chain 1 had a length 198,022,000 states, chain 2 had a length of 193,085,000 states, and chain 3 had a length of 191,402,000 states. Recall from Section 3.3.2 that we specified only one sample is stored every 10^4 states. This means that in all trace plots, as sampling progresses from left to right, that a new sample only occurs every 10^4 states instead of at every state.

While the trace plots and ESSs for all parameters should be inspected, here we only present the trace plots and ESSs for sum(indices.alltrees) and the posterior. 'Sum(indices.alltrees)' is the statistic that defines the number of times

that the trajectory of effective population size changes throughout history, and 'posterior' is the term used to refer to the log posterior probability.

The trace plots in Figure 3.13 suggest that the data do not strongly support one definitive result, given that both two and three changes in effective population size dynamics are frequently accepted in sum(indicators.alltrees). Still inspecting the trace plots for sum(indicators.alltrees), we notice that zero is never accepted when sampling. Furthermore, the 95% HPD for sum(indicators.alltrees) excludes zero, so we can confidently reject the possibility of a constant effective population size through time.

The ESS for all parameters is given in Tables C.1, C.2, C.3, and C.4 of Appendix C.1.2. Nearly all variables have an ESS above 100, with the majority additionally having an ESS above 200. The individual population size and indicator variables sometimes have low ESS values, especially deep in the tree: 15 of these 231 parameters that define the EBSP had an ESS below 100.

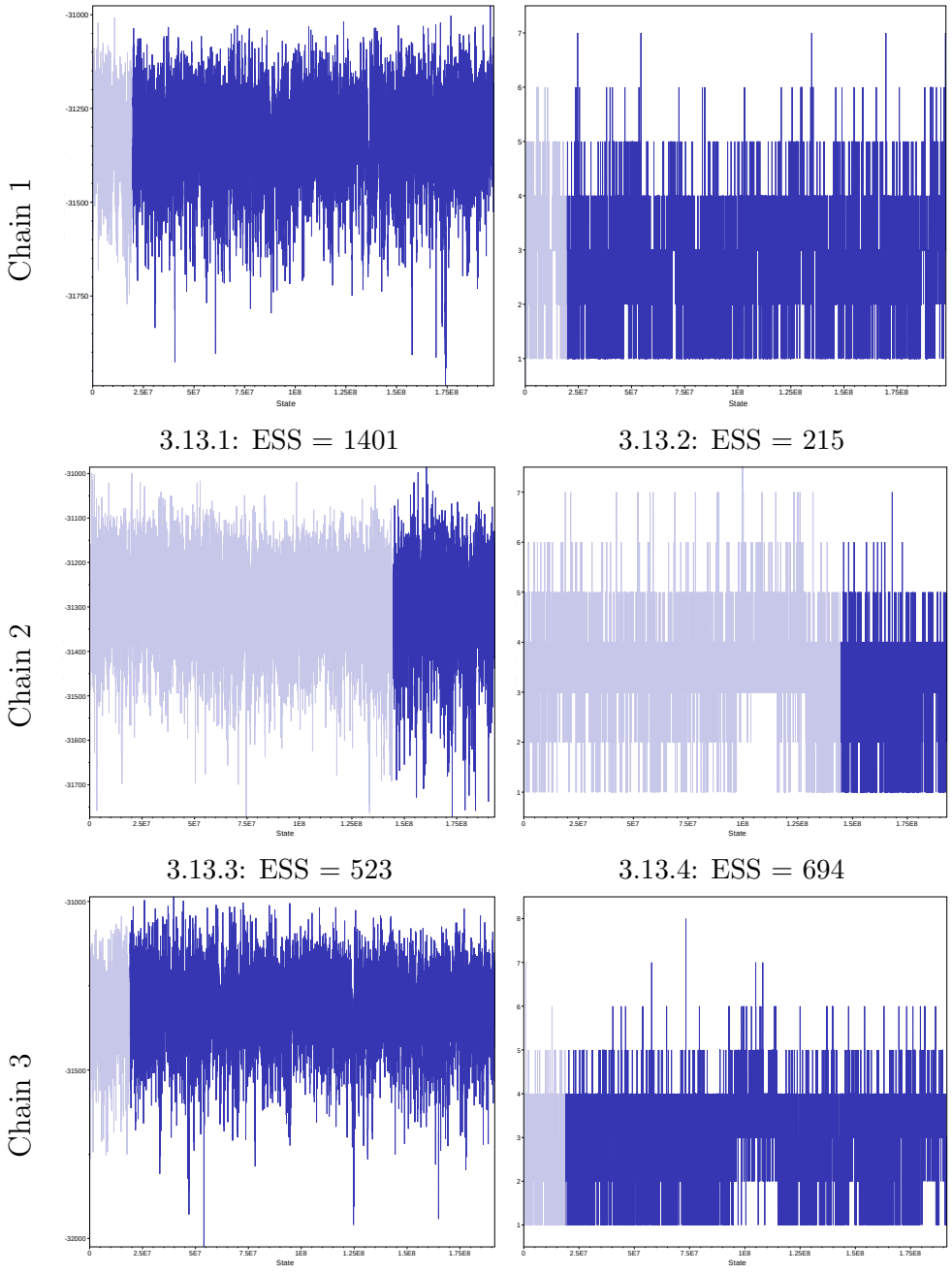
We also note that the estimate of the standard deviation of the clock rate is greater than 0.1, which confirms that a relaxed clock was required for this analysis.

We combine the parts of each MCMC run that have converged to the same stationary distribution using the program 'LogCombiner'. 10% burn-in was discarded from both the first and third MCMC runs, while 75% burn-in was discarded from the second run.

We inspected the combined log file in Tracer, and found that the posterior had an ESS of 1940, while sum(indicators.alltrees) had an ESS of 442. The trace plots for these two parameters are given in Figures 3.14 and 3.15. The trace plots for the posterior shows no trend, which is ideal. The trace plot for sum(indicators.alltrees) is noticeably higher just after states 1.25×10^8 and 2×10^8, but quickly returns to the baseline. This plot generally has no trend, and coupled with an ESS of over 200, it is likely reasonable to use these combined posterior samples to construct an EBSP. To further verify this claim, we calculate $\widehat{R}$ for all parameters of interest using the R package `rstan`.

The resulting values of $\widehat{R}$ are given in Table 3.3. All $\widehat{R}$ values are less than or equal to 1.01, and so as per Vehtari *et al.* [88], $\widehat{R}$ does not suggest that our MCMC chains have not converged.

We still found that some individual population size parameters and indicator parameters were poorly sampled deeper in the tree, and could not increase



3.13.1: ESS = 1401 3.13.2: ESS = 215

3.13.3: ESS = 523 3.13.4: ESS = 694

3.13.5: ESS = 1585 3.13.6: ESS = 139

Figure 3.13: Trace plots for the posterior (left) and sum(indicators.alltrees) (right) of the raw posterior samples. Each row is a different MCMC chain and the ESS for each parameter is given as a subcaption. The burn-in for each chain is indicated by a transparent region in each trace plot.

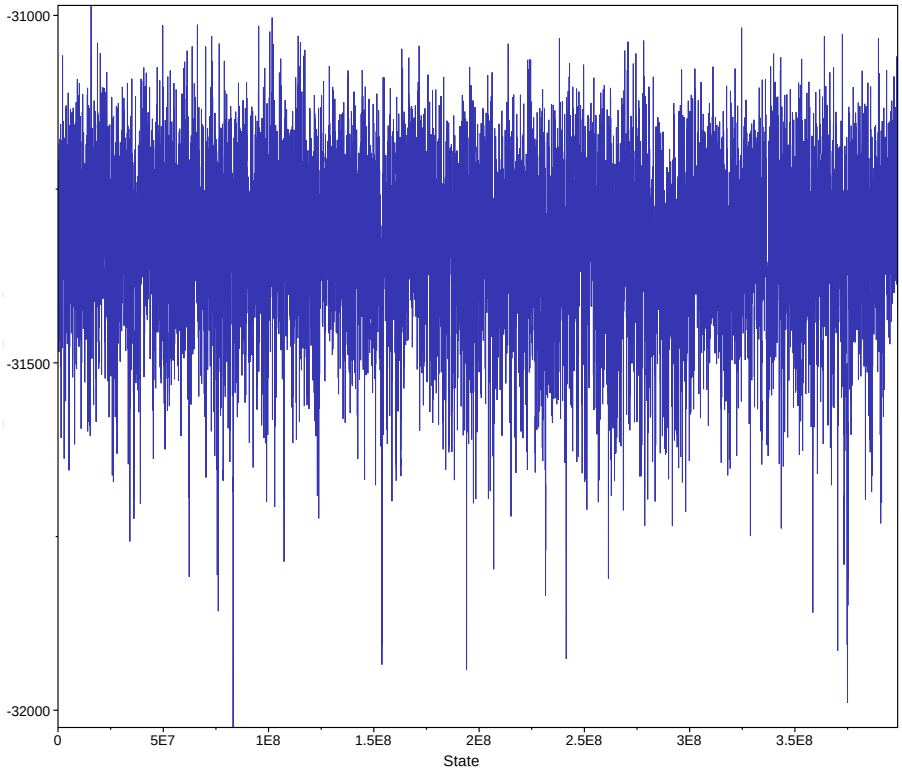


Figure 3.14: Trace plot for the posterior based on the combined posterior samples.

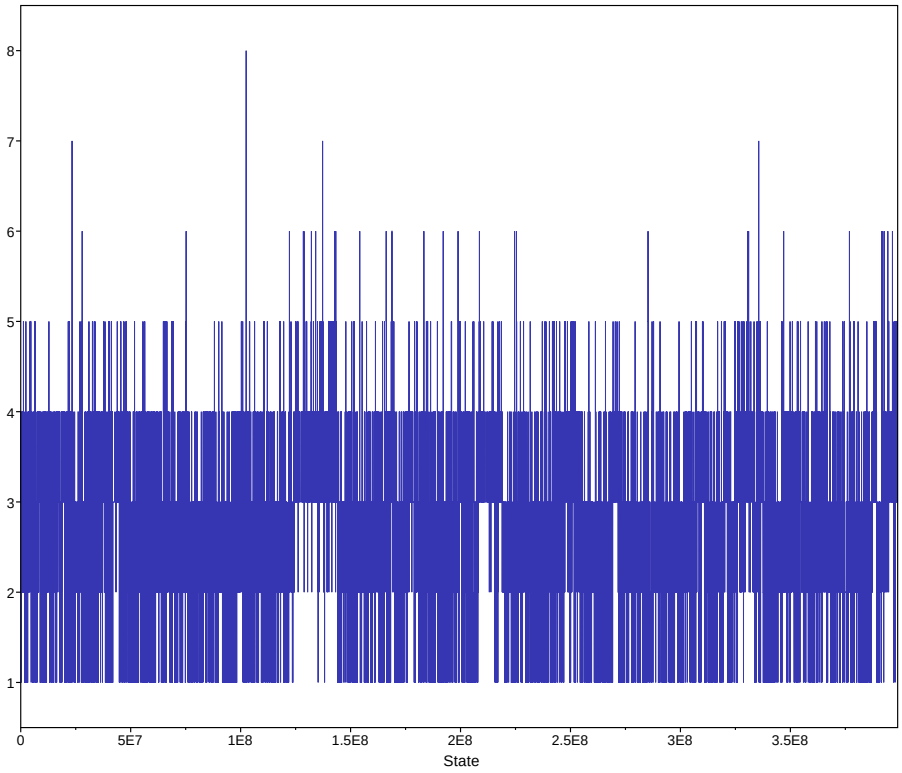


Figure 3.15: Trace plot for the variable sum(indicators.alltrees) based on the combined posterior samples.

Parameter	$\widehat{R}$
Posterior	1.0047
Likelihood	1.0002
Prior	1.0049
Tree Height	1.0010
κ (HKY substitution model)	1.0010
γ (shape parameter)	1.0004
Extended Bayesian Skyline	1.0106
sum(indicators.alltrees)	1.0061
Uncorrelated clock standard deviation	1.0011
Substitution rate mean	1.0005
Substitution rate variance	1.0018

Table 3.3: $\widehat{R}$ values for all main parameters of interest. Values closer to one indicate better mixing within chains. Vehtari *et al.* [88] recommend only using the posterior samples if $\widehat{R} < 1.01$.

the ESS through standard troubleshooting methods such as increasing the relevant operator weights. Performing validation runs will help to verify whether the EBSP from this analysis is a stable result.

The combined samples from the posterior distribution were analysed using an R script based on one written for an EBSP tutorial by J. Heled and T. Vaughan (2015). The resulting EBSP is given in Figure 3.16. Note that linear trends on a log scale correspond to exponential growth or decay. Considering the effective population size forward in time, we see that the effective population size remains constant until approximately 50 ka, when it starts to increase exponentially. The effective population size stabilises again approximately 35-40 ka, and then remains constant until approximately 8 ka, after which it undergoes exponential growth until the present day.

We perform three further MCMC analyses as validation, using the same BEAST specification. These appear to have slightly different marginal densities for sum(indices.alltrees). The trace plots and marginal densities for the posterior and sum(indices.alltrees) are provided in Appendix C.1.3.

After selecting an appropriate amount of burn-in for each of the validation runs, we construct EBSPs to compare to the results from the initial combined MCMC chains. We used 20% burn-in for the first validation run, and 10%

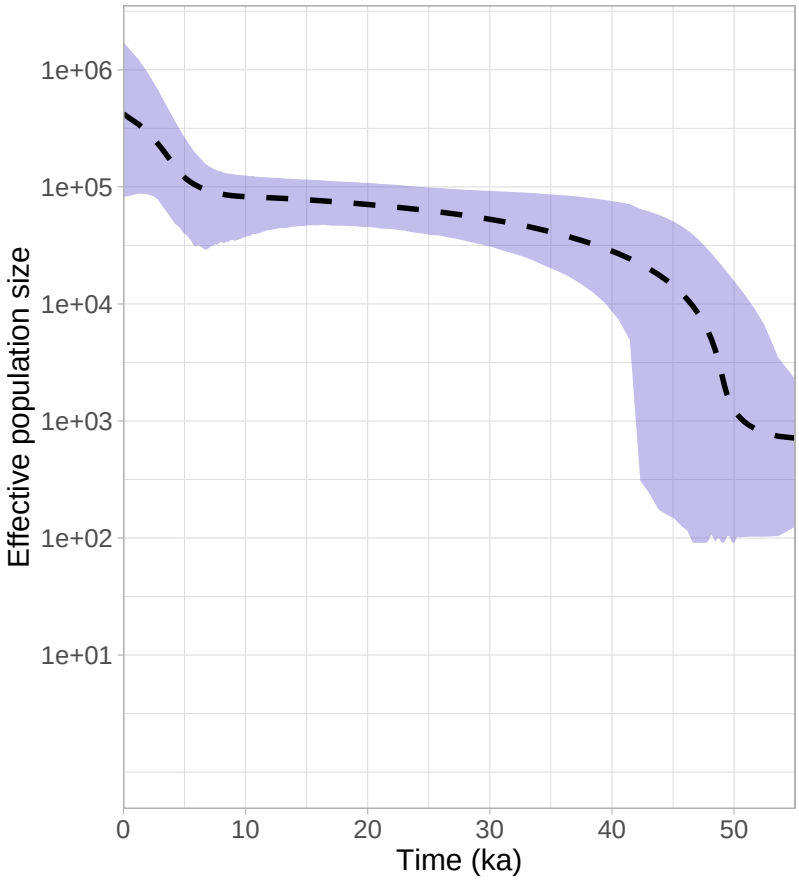


Figure 3.16: The Extended Bayesian Skyline Plot for the three combined BEAST analyses. The 95% HPD interval over time is shown as the filled blue area, while the median estimated effective population size over time is given by a dashed line. Note that the effective population size is presented on a log scale.

burn-in for the remaining two validation runs. The three resulting EBSPs are given in Figure 3.17.

We note that the EBSPs from these three validation runs are consistent with the earlier results, despite the slight difference in marginal densities of some variables. This gives us a higher level of confidence in the estimated effective population sizes shown in Figure 3.16.

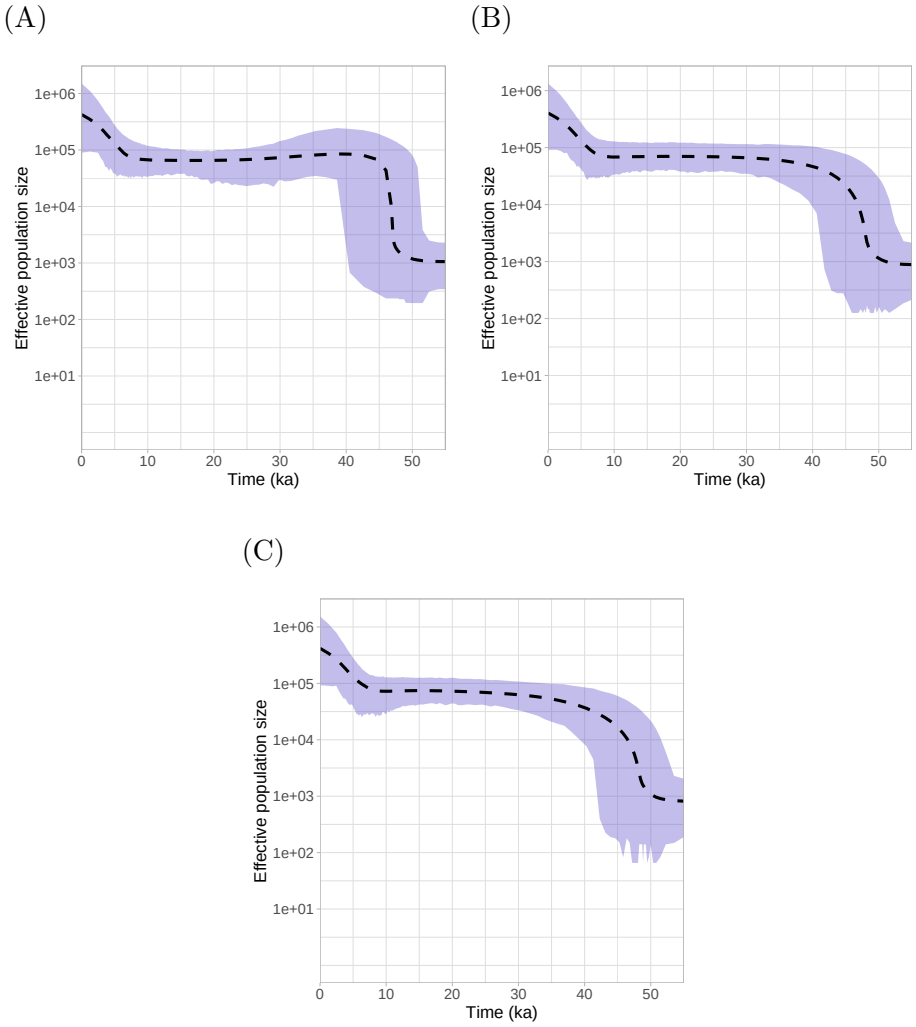


Figure 3.17: Extended Bayesian Skyline Plots for the three validation BEAST analyses. EBSP (A) was produced from the samples in validation chain 1, after 20% burn-in was removed. EBSP (B) was produced from the samples in validation chain 2, after 10% burn-in was removed. Finally, EBSP (C) was produced from the samples in validation chain 3, after 10% burn-in was removed.

In this chapter we presented maximum likelihood trees, BEAST consensus trees, and the Extended Bayesian Skyline Plot for the mtDNA alignment, as well as the theory underpinning each of these methods. We found that major mtDNA haplogroups were well-supported, and also that some events deeper in the maximum likelihood and BEAST consensus trees had low support. This could possibly be the result of rapid migration events leaving little signal to recover in the reconstruction of trees. In the next chapter we describe our simulation study, which will allow us to see which, if any, of the different migration scenarios best describe our data.

Chapter 4

Methods for Dimension Reduction and Classification

The summary statistics as simulated by the coalescent simulation program BayeSSC are high-dimensional; to visualise these statistics, we need to use dimension reduction techniques. We will then apply classification methods to select the migration model that the summary statistics were most likely simulated under. This chapter describes methods for dimension reduction, classification methods, and techniques used to evaluate classifier performance.

4.1 Dimension Reduction

We will use two different methods for dimension reduction, which can be used to visualise high-dimensional data. These methods are principal component analysis (PCA) and uniform manifold approximation and projection (UMAP). PCA is a linear dimension reduction technique, while UMAP is a non-linear dimension reduction technique. We will explore the differences in these two techniques by applying them to the example data shown in Figure 4.1.

Reducing two dimensions to one provides a simple way to visualise how these methods work when applied to higher-dimensional data.

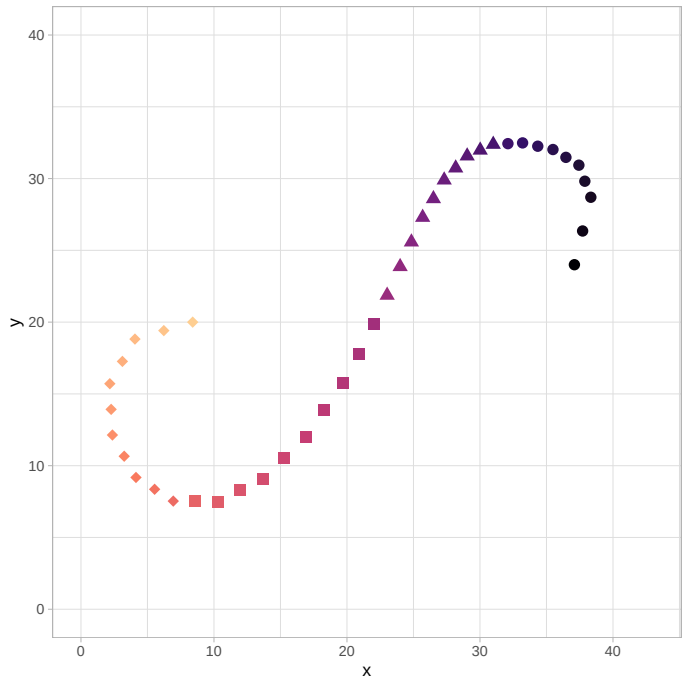


Figure 4.1: Two-dimensional data, where the points lie along a S-shaped curve. To ensure that points are comparable between two- and one-dimensional visualisations of the data, the shape and colour of the points change as the curve is traversed.

4.1.1 Principal component analysis

Principal component analysis (PCA) is a form of dimension reduction that identifies the successive directions of greatest variance in the data; the principal components are then the data projected onto each of these directions. The direction of each subsequent principal component must be orthogonal to all previous principal components. For n-dimensional data, n principal components can be found, assuming that there are more than n observations. Since the number of observations is a controllable parameter in simulation studies, this condition does not affect our use of PCA.

We can use the first k principal components to find a representation of the data in k-dimensional space. This process will be described after the PCA algorithm, which is given in Algorithm 3.

Algorithm 3: PCA

1. Center and scale the data X to get X'.
2. Find the covariance matrix C for X'.
3. Calculate eigenvalues of C, where $\lambda_1 \geq \lambda_2 \geq \ldots \geq \lambda_n$.
4. Calculate the corresponding eigenvectors of C, denoted $\boldsymbol{v}_i$ for $i = 1, 2, \ldots n$. Each eigenvector gives the direction of a principal component.

Step 1 is not required for the PCA algorithm; however, it is advised [38]. PCA is not scale invariant, which means that the results change for different scaling of the variables. Since summary statistics for DNA have different scales, *e.g.* some are proportions while some are counts, we will center each variable by subtracting its mean and scale each variable by dividing by its standard deviation before performing PCA.

Let the columns of the matrix W be the eigenvectors corresponding to the k greatest eigenvalues. The data X can then be projected into k dimensions by finding XW, *i.e.* by postmultiplying by the first k eigenvectors. By using all eigenvectors as columns of W, we can find all principal components of the data.

The eigenvalues of the covariance matrix can also be used to find the proportion of variance explained by each principal component. This is calculated by dividing each eigenvalue by the sum of all eigenvalues. All eigenvalues of a covariance matrix are non-negative, which is a consequence of the structure of a covariance matrix. Hence, the proportion of variance explained is guaranteed to be non-negative. In practice, the cumulative proportion

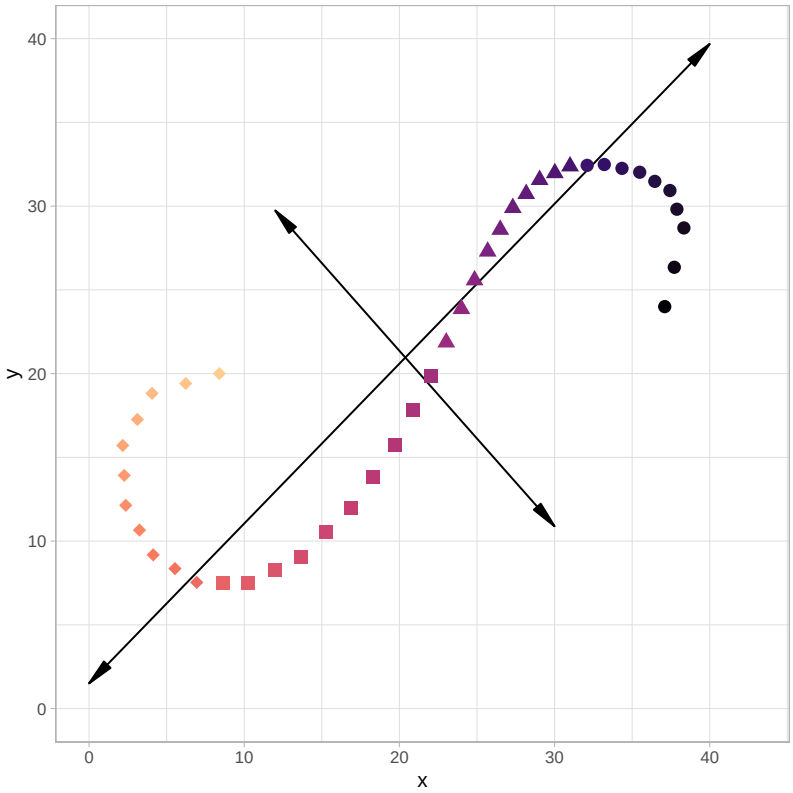


Figure 4.2: The first two principal components for the data from Figure 4.1.

of variance explained is often considered when determining the number of principal components to use.

For more information on PCA, we refer the reader to Jolliffe and Cadima [38].

For the data presented in Figure 4.1, the directions of the first two principal components are given by the arrows in Figure 4.2. The direction of the first principal component corresponds to the largest possible variance in the data; the direction of the second principal component is orthogonal (perpendicular in 2D) to the that of the first principal component. We can see that it takes the direction of maximum variance satisfying this condition.

The first principal component is shown in Figure 4.3; it explains 97.9% of the variance of the data. We see that the structure of the transformed data is similar to the structure in Figure 4.1, but the colours are mixed towards the ends. This is a consequence of using a linear dimension reduction technique.

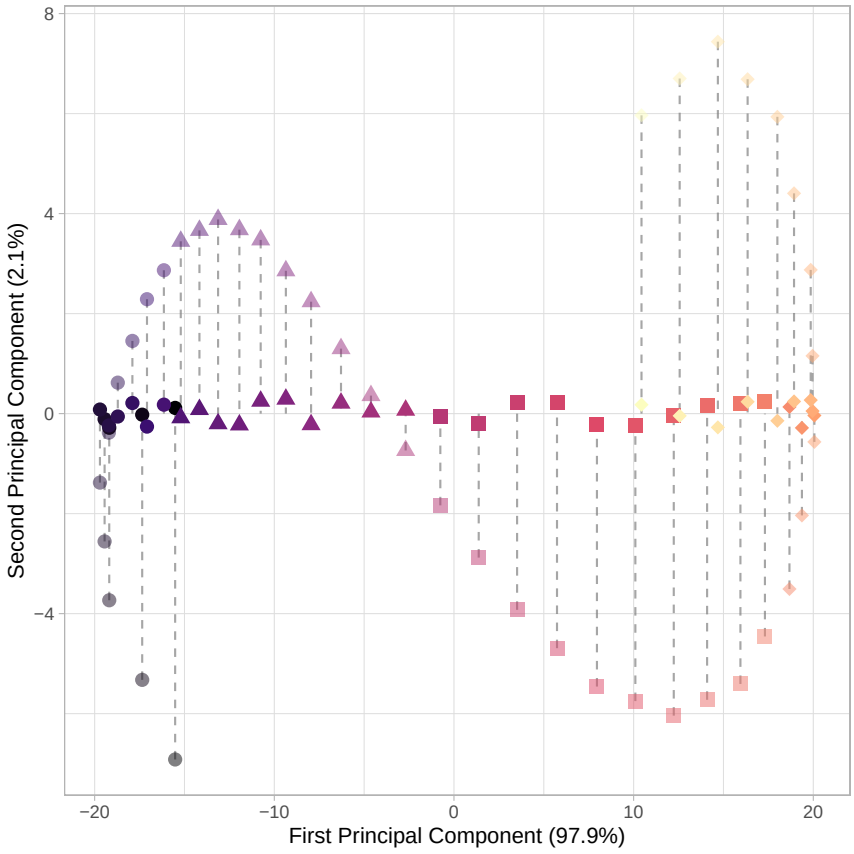


Figure 4.3: The projection of the original data from Figure 4.1 onto the direction of the first principal component. The original data is partially transparent and the projections are indicated by dashed lines. A small amount of vertical jitter was applied to the projected points so that overlapping points could be easily distinguished.

4.1.2 Uniform manifold approximation and projection

Uniform manifold approximation and projection (UMAP) is a recently developed method which has already been widely adopted as a dimension reduction technique. A detailed discussion on manifolds and their use in this method is beyond the scope of this thesis, and we direct the reader to the paper by McInnes *et al.* [51]. Here, we will discuss the implementation of the UMAP algorithm in the Python package `umap-learn` [52].

Unlike PCA, UMAP requires some parameters to be defined before it can be used for dimension reduction. Some key parameters of the UMAP implementation in `umap-learn` are described below.

- `n_components` is the desired dimension of the lower-dimensional space.
- `n_neighbours` controls the 'connectedness' of the points in the high dimensional space. Larger `n_neighbours` results in more points being connected, which better preserves the overall structure of the data. Conversely, smaller `n_neighbours` better preserves the local structure around each point, but the overall structure of the data might become very distorted.
- `min_dist` is the minimum distance required between points in the lower-dimensional space. Forcing a high minimum distance can result in no discernible trend, as points are repelled from each other.

One benefit of UMAP is that it provides results of a similar quality as other commonly used non-linear dimension reduction techniques, but has a considerably shorter run time. This was achieved through the use of stochastic methods (for details, see McInnes *et al.* [52]). While the algorithm is quicker than other methods, this stochasticity means that it is important to perform multiple replicates to ensure that the results are consistent. It is also important to record the `random_state` parameter so that any results are reproducible; this is the random seed used in the algorithm.

In practice, there are no guidelines for the choice of `n_neighbours` and `min_dist`. Instead, a grid search can be conducted across multiple values for `n_neighbours` and `min_dist`.

UMAP is applied to the data in Figure 4.1 for different `n_neighbours` and `min_dist`; the results are given in Figure 4.4. As with the PCA example, we project this data into one-dimensional space; *i.e.* `n_components` $= 1$. Unlike PCA, where the cumulative proportion of variance explained can be

calculated, there is no way to determine the optimal number of components in UMAP.

The 'unwrapping' demonstrated in the bottom-right triangle of Figure 4.4, especially the centre panel and the bottom-right panel, clearly demonstrates the difference between a linear dimension reduction method (PCA) and a non-linear dimension reduction method (UMAP). As seen in Figure 4.3, linear dimension reduction methods are not capable of this type of data transformation.

Another point that we highlight in the UMAP results is that distance between clusters in the lower-dimensional space is not necessarily meaningful. In the three panels in the upper-left triangle of Figure 4.4, there are two main clusters that do not correspond to any features in the original data. This is a consequence of the small number of nearest neighbours used, coupled with the low minimum distance. For example, when 10 nearest neighbours are specified in the function call, the clustering only appears for the smallest value of `min_dist`, 0.1.

While it is simple to tell that the data in Figure 4.1 had non-linear structure, and therefore required a non-linear dimension reduction technique to visualise it in a lower-dimensional space, it is not immediately obvious which method should be used when analysing unknown high-dimensional data. We will later apply both PCA and UMAP to the high-dimensional summary statistics. Since UMAP requires more specification than PCA, the resulting visualisations should be considered with caution. As we saw in Figure 4.4, UMAP can provide a potentially useful low-dimensional visualisation of the data.

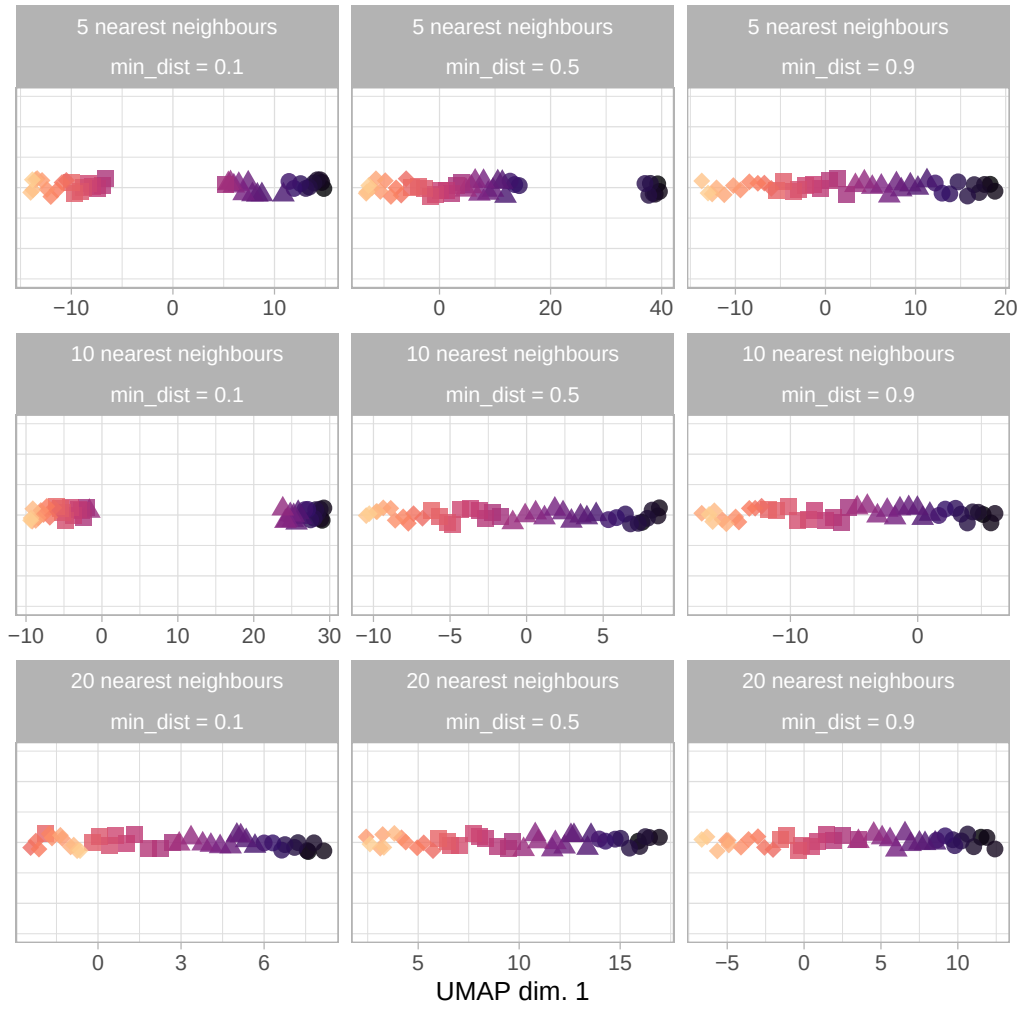


Figure 4.4: Applying UMAP to the data from Figure 4.1. A range of values for `n_neighbours` and `min_dist` were compared. Once again, vertical jitter and transparency are used to display overlapping points.

4.2 Classification: An Example

A classification problem arises when we have an observation of some numerical variables and wish to determine which 'class' this observation belongs to. Class is a categorical variable, and the number of classes is always known in advance. Classification is a form of supervised learning, because the known classes of some observations are used to determine rules to separate the classes.

To illustrate the concept of classification, we present a problem using a simple dataset. This data was simulated by sampling from normal distributions as parameterised in Table 4.1.

Suppose that you collect three particular types of shells from a local beach. Each of these shells has their length and width measured, as described in Table 4.2. There are 75 pairs of measurements (length and width) for each shell type.

Shell Type	Shell Measurement	Mean μ (cm)	Standard Deviation σ (cm)
Type 1	Length	6	1
Type 1	Width	6	1
Type 2	Length	4	3
Type 2	Width	8	0.5
Type 3	Length	3	1
Type 3	Width	4	2

Table 4.1: Mean and standard deviations of normal distributions behind the length and width of each type of shell.

Now suppose that a friend finds a shell at the same beach to add to your collection, but will only give you the shell if you can guess the type of shell. They give you the length and width of the shell as a hint. This is shown in the final row of Table 4.2, where the length and width of the shell are known but the type is unknown. We want to classify this shell as some particular type, based on what we already know about the measurements of shells.

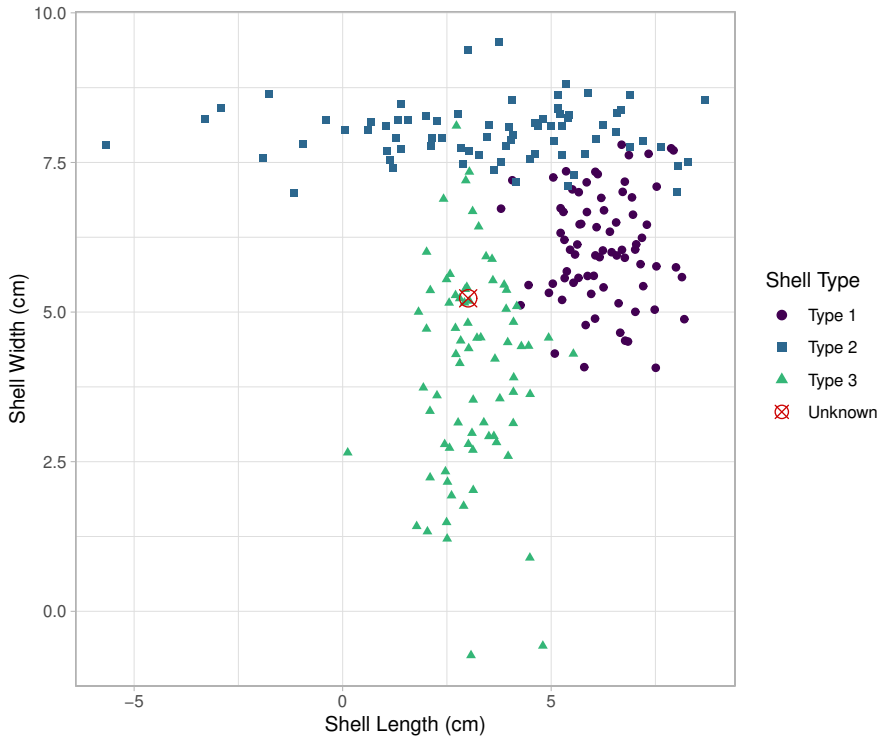


Figure 4.5: A scatter plot of shell width against shell length for this toy dataset, coloured by shell type. The measurements of the shell with unknown type are represented by a star.

Shell Length (cm)	Shell Width (cm)	Shell Type
6.94	6.92	Type 1
5.63	6.13	Type 1
6.08	7.89	Type 2
2.13	7.90	Type 2
2.98	5.42	Type 3
⋮	⋮	⋮
2.81	4.15	Type 3
3.01	5.23	???

Table 4.2: Shell lengths and widths for different types of shells, labelled Type 1, Type 2, or Type 3. The final row shows the measurements of the shell that we wish to classify as one of the three types.

This is a classification problem with three classes: Type 1, Type 2, and Type 3. The variables are shell length and shell width, and each shell is one observation. The measurements of all shells that have already been collected would be the data used to train any classifiers.

Now, we consider the classification problem in two parts:

1. Training the classifier on data where the class is known;
2. Classifying data where the class is unknown.

First, we use observations (shell length and width) with class labels (shell types) to train a classifier. This step involves estimating parameters from data that define the classifier. Data used at this step is referred to as *training data*.

Once a classifier has been trained, it can take numerical variables as input (shell length and width) and then predict a class (shell type). Note that we trained the classifier on shells of Type 1, Type 2, and Type 3; this means that the classifier will only predict one of these three types. Based on the distributions of shell measurements of each type in Figure 4.5, it seems likely that the new shell is a shell from Type 3.

Depending on the classifier used, a soft or hard classification may be produced. Soft classification results in a list of numbers between zero and one that indicate the degree of class membership for each class. These can sometimes be interpreted as the probability of an observation belonging to each class. Alternatively, hard classification gives a single class label.

In the following sections we describe how the concept of classification extends to migration models and summary statistics, and explore classification methods that could be used to classify the new shell.

4.3 Using Classification Methods to Identify Migration Routes

The remaining sections of my thesis aim to use classification methods to investigate potential migration models that describe the path taken by Aboriginal Australians from southeast Asia to Sahul. A migration model comprises a geographical migration route, as well as information describing the demographic history and how DNA changes through time. Each of the migration

models describe a different geographical path, which will be fully explained in Chapter 5.

Framing this classification problem with the same terminology as the shell example, we can consider each migration model as a class, each simulation as one observation, and every summary statistic calculated from simulated DNA as a predictor variable. Since there are nine candidate migration models there are nine classes, and therefore this is a multiclass classification problem. The shell classification problem described in the previous section is also a multiclass classification problem. When there are only two classes, we say that it is a binary classification problem.

The following steps describe how classification can be used to select a migration model.

1. Simulate DNA under each of the migration models.
2. Find the simulated summary statistics by calculating summary statistics for the simulated DNA.
3. Find the observed summary statistics by calculating summary statistics on the actual collected DNA.
4. Use the *simulated* summary statistics and known migration models as predictor variables and class labels to train a classifier.
5. Use the *observed* summary statistics as input for the trained classifier, which will then predict some migration model.

These steps are slightly different to the process described previously in the shell example. We draw attention to Steps 4 and 5 of the above procedure: the classifier is trained on simulated summary statistics, and then used to predict the class of the observed summary statistics.

In the shell example, we train the classifier on the measurements of collected shells, and then use the classifier to predict the type of another actual shell. Unfortunately in this case we have exactly one set of summary statistics from our observed data, and we do not know the migration path that resulted in the collected DNA or summary statistics. This is analogous to having one set of length and width measurements from a shell, and not even knowing the type of shell that was measured.

A classifier cannot be trained on a single observation of real data, and the data that we have does not describe what would have happened under the 'incorrect' migration models. There is no method that allows us to observe real mtDNA data that evolved under different migration models, as the initial

migration from southeast Asia to Sahul only occurred once and time travel does not exist. The next best option is to simulate the changes in DNA that would likely have occurred under different migration models.

This means that any classifier used to predict a migration model based on real mtDNA data is answering the following question:

> "If the real mtDNA sequences were output from BayeSSC, under which migration model were they most likely simulated?"

There is also the limitation inherent to all classifiers that the selected class is not guaranteed to be correct. This is true even when we are using the same type of data for training and prediction. We return to the shell example: suppose that the shell that my friend found was of Type 4, and Type 4 shells are a similar size to Type 3 shells. The classifier would likely classify the shell as Type 3, but this would be an incorrect result. Likewise, the migration model predicted for the real mtDNA data can only be one of the migration models on which the classifier was trained.

4.4 Classification Methods

As described in Section 4.3, classification methods can be used to select a migration model based on summary statistics of DNA.

Multinomial logistic regression, support vector machines, and neural networks can all be applied to multiclass classification problems, and can therefore be used to select the most likely migration model based on the observed mtDNA summary statistics. In all cases, training data is used to estimate the parameters before the trained classifier can be used to make predictions for new data.

4.4.1 Multinomial logistic regression

Multinomial logistic regression (MLR) is often a first approach for multiclass classification. MLR calculates the probability of an observation belonging to each of the K classes. This calculation of class probabilities means that MLR is a type of soft classification, since each class has a degree of membership. If a hard classification is desired, the observation is assigned to the most probable class.

Suppose that the data $X_{n\times p}$ is made up of n observations of p variables. We refer to one observation as $\boldsymbol{x}_i$, for $i = 1, 2, \ldots, n$. The corresponding class of each observation $\boldsymbol{x}_i$ is the categorical variable y_i, where $y_i \in \{1, 2, \ldots, K\}$. If we take the reference class to be class K, there are $K - 1$ logistic regressions,

$$\ln\left(\frac{P(y_i = k|\boldsymbol{x}_i)}{P(y_i = K|\boldsymbol{x}_i)}\right) = c_k + \boldsymbol{x}_i\boldsymbol{\beta}_k, \qquad k = 1, 2, \ldots, K-1. \tag{4.1}$$

In Equation 4.1, the data $\boldsymbol{x}_i$ are a $1 \times p$ row vector, while the regression coefficients $\boldsymbol{\beta}_k$ are a $p \times 1$ column vector. A different set of regression coefficients is estimated for each logistic regression.

Exponentiating both sides, we can find the probability of the observation belonging to class k,

$$P(y_i = k|\boldsymbol{x}_i) = P(y_i = K|\boldsymbol{x}_i)e^{c_k + \boldsymbol{x}_i\boldsymbol{\beta}_k}, \qquad k = 1, 2, \ldots, K-1. \tag{4.2}$$

An implicit assumption of MLR is that each observation $\boldsymbol{x}_i$ belongs to one of the K classes, *i.e.* that

$$\sum_{k=1}^{K} P(y_i = k|\boldsymbol{x}_i) = 1 \text{ for all } i. \tag{4.3}$$

Using Equation 4.3, we can find the probability of belonging to each class expressed in terms of the regression coefficients and the data.

$$
\begin{aligned}
P(y_i = K|\boldsymbol{x}_i) &= 1 - \sum_{k=1}^{K-1} P(y_i = k|\boldsymbol{x}_i) \\
&= 1 - \sum_{k=1}^{K-1} P(y_i = K|\boldsymbol{x}_i) e^{c_k + \boldsymbol{x}_i \boldsymbol{\beta}_k} \\
\Rightarrow P(y_i = K|\boldsymbol{x}_i) &= 1 - P(y_i = K|\boldsymbol{x}_i) \sum_{k=1}^{K-1} e^{c_k + \boldsymbol{x}_i \boldsymbol{\beta}_k} \\
\Rightarrow 1 &= P(y_i = K|\boldsymbol{x}_i) + P(y_i = K|\boldsymbol{x}_i) \sum_{k=1}^{K-1} e^{c_k + \boldsymbol{x}_i \boldsymbol{\beta}_k} \\
1 &= P(y_i = K|\boldsymbol{x}_i) \left(1 + \sum_{k=1}^{K-1} e^{c_k + \boldsymbol{x}_i \boldsymbol{\beta}_k} \right) \\
\Rightarrow P(y_i = K|\boldsymbol{x}_i) &= \frac{1}{1 + \sum_{k=1}^{K-1} e^{c_k + \boldsymbol{x}_i \boldsymbol{\beta}_k}}. \qquad (4.4)
\end{aligned}
$$

Substituting Equation 4.4 into Equation 4.2, we can find the class probabilities

$$
P(y_i = k|\boldsymbol{x}_i) = \begin{cases} \frac{e^{c_k + \boldsymbol{x}_i \boldsymbol{\beta}_k}}{1 + \sum_{k=1}^{K-1} e^{c_k + \boldsymbol{x}_i \boldsymbol{\beta}_k}}, & k = 1, 2, \ldots, K-1, \\ \frac{1}{1 + \sum_{k=1}^{K-1} e^{c_k + \boldsymbol{x}_i \boldsymbol{\beta}_k}} & k = K. \end{cases} \qquad (4.5)
$$

Determining the probability of class membership relies on knowing the parameter estimates $\boldsymbol{\beta}_k$. These estimates are determined using maximum likelihood estimation. Maximum likelihood estimates are the parameters that maximise the likelihood of observing the data. Using the probabilities from Equations 4.4 and 4.5, the likelihood of observing the data given the parameter values is given by

$$
L(\boldsymbol{x}; \boldsymbol{\beta}) = \prod_{i=1}^{n} \left[\frac{e^{c_k + \boldsymbol{x}_i \boldsymbol{\beta}_k}}{1 + \sum_{k=1}^{K-1} e^{c_k + \boldsymbol{x}_i \boldsymbol{\beta}_k}} \right]^{\sum_{k=1}^{K-1} C_{ik}} \left[\frac{1}{1 + \sum_{k=1}^{K-1} e^{c_k + \boldsymbol{x}_i \boldsymbol{\beta}_k}} \right]^{C_{iK}}, \qquad (4.6)
$$

where C is an indicator variable such that

$$
C_{ik} = \begin{cases} 1 & \text{if } y_i = k, \\ 0 & \text{if } y_i \neq k. \end{cases}
$$

In practice the log-likelihood is maximised instead, which yields the same parameter values that would be obtained from maximising the likelihood. The log-likelihood is found by taking the logarithm of Equation 4.6. Optimisation methods can be used to calculate maximum likelihood estimates.

Further discussion on the theory of multinomial logistic regression is given by Friedman, Hastie, and Tibshirani [26].

As with all regression models, the predictor variables to include in Equation 4.1 must be chosen with some thought. There is also no requirement that the predictor variables are from the raw data. We may wish to make indicator variables to include categorical data, or to derive other variables from the raw data. Variables should be selected through a combination of subject matter expertise and variable selection algorithms.

We consider two algorithms for variable selection: forwards selection based on the Akaike Information Criterion (AIC), and the least absolute shrinkage and selection operator (LASSO).

Information Criteria

There are two main information criteria used when assessing model fit: Akaike Information Criterion (AIC) and Bayesian Information Criterion (BIC)

AIC can be used to compare nested models, which makes it ideal for selecting the regression model in MLR. The predictive accuracy of regression models depends on the variables included in the model. While adding predictor variables will increase the accuracy of the model on the training data, this increases the risk of overfitting and the model may not predict well for new data. AIC penalises the inclusion of too many parameters while rewarding a high likelihood, and is defined as

$$AIC = 2p - 2\log(\hat{L}). \tag{4.7}$$

The value of the log-likelihood evaluated at the maximum likelihood estimates of the parameters is denoted $\log(\hat{L})$, and there are p parameters [2]. In a model selection framework, preferred models have lower values of AIC. In our analyses, the model with the smallest AIC is chosen.

BIC also penalizes the inclusion of free parameters p.

$$BIC = \log(n)p - 2\log(\hat{L}), \tag{4.8}$$

where n is the number of observations in the data [75]. Due to the different coefficient of p, BIC penalizes complex models more than AIC except when there is a very small amount of data [26]. Similarly to AIC, models with smaller BIC are preferable, and in our analyses we will select the model with the smallest BIC.

Both AIC and BIC are used when comparing substitution models in ModelGenerator.

To find the absolute minimum AIC, we would need to calculate the AIC for all models under consideration. Calculating the AIC for all possible regression models is infeasible, and so forward selection is used to restrict the combinations of predictor variables.

Forward selection

Forward selection is used to determine which predictor variables should be included in a regression model. At first, no predictor variables are included in the model. Each variable is then added one at a time, testing all possible ways to add one new variable to the model. Some performance metric is calculated for each possible model. The predictor variable with the best performance metric is then included in the regression model, and the process is repeated [26].

Predictor variables are no longer added when the performance metric ceases to improve upon adding another variable. AIC and BIC are both common performance metrics used to compare models in forward selection.

There is the caveat that forward selection is not guaranteed to select the best subset of predictor variables [26]. For this reason we will also use LASSO for variable selection, and compare the results for the two algorithms.

LASSO

The least absolute shrinkage and selection operator (LASSO) penalizes the size of the regression coefficients, shrinking them towards zero [27]. Consider performing LASSO on a MLR model with K classes and n observations. We define the indicator variable y_{ik} as

$$y_{ik} = \begin{cases} 1 \text{ if observation } i \text{ is from class } k; \\ 0 \text{ if observation } i \text{ is not from class } k. \end{cases} \tag{4.9}$$

For fixed λ, LASSO maximizes the penalised log-likelihood

$$\frac{1}{N}\sum_{i=1}^{n}\left[\sum_{k=1}^{K} y_{ik}(c_k + x_i^T\boldsymbol{\beta}_k) - \log\left(\sum_{k=1}^{K} e^{c_k + x_i^T\boldsymbol{\beta}_k}\right)\right] - \lambda\sum_{k=1}^{K}|\boldsymbol{\beta}_k|. \qquad (4.10)$$

The penalty applied to the regression coefficients $\boldsymbol{\beta}_k$ increases as λ increases. An appropriate value for λ is chosen through cross-validation; the general cross-validation process is described in Section 4.5.1. The misclassification error (proportion of misclassified points) is calculated using cross-validation for a range of λ values. The chosen λ value is the one with the lowest cross-validated misclassification error.

Using MLR for prediction

Once the predictor variables and corresponding coefficients have been determined, we can use MLR to predict the most likely class of new data.

Suppose that we have m observations of p predictor variables, $X_{m\times p}$. This could be completely new data that we wish to know the class of, but it is also reasonable to predict the classes of observations in the training data. In fact, this can be an important validation process.

Recall that $\boldsymbol{x}_i$ denotes the i^{th} row of matrix X. The predicted classes Y^* are determined by assigning each observation to the class with the highest probability, *i.e.*

$$y_i^* = \underset{k}{\text{argmax}}\ P(y_i^* = k|\boldsymbol{x}_i) \quad \text{for } i = 1, 2, \ldots, m.$$

4.4.2 Support vector machines

Support vector machines (SVMs) are a classification method that relies on finding an optimal boundary, or a *separating hyperplane*, between classes. A hyperplane is the generalisation of a line to higher dimensions; *e.g.* a line is a hyperplane in $\mathbb{R}^2$, and a plane is a hyperplane in $\mathbb{R}^3$. It is called the optimal separating hyperplane because the hyperplane is chosen to best separate the different classes.

First, consider a binary classification problem. The target variable y can take values ± 1 to represent two different classes. These two classes are linearly separable if a 'straight line' can be drawn to separate the two classes.

More formally, for some data $\boldsymbol{x}$, classes are linearly separable when for each observation i, $\boldsymbol{w}$ and b can be found such that

$$\begin{aligned} \boldsymbol{w} \cdot \boldsymbol{x}_i + b > 0 \quad & \text{for } y_i = 1, \quad \text{and} \\ \boldsymbol{w} \cdot \boldsymbol{x}_i + b \leq 0 \quad & \text{for } y_i = -1. \end{aligned}$$

An example of linearly separable classes is given in Figure 4.6. While there are many possible hyperplanes that separate the two classes here, support vector machines find the hyperplane that maximises the width of the margin. This means that the hyperplane depends only on the points lying on the margin, *i.e.* the support vectors. These vectors are used to determine the weights $\boldsymbol{w}$ and the bias b that define the hyperplane $\boldsymbol{w} \cdot \boldsymbol{x} + b = 0$.

This scenario describes the basic idea behind linear SVMs, except instead of a two dimensional hyperplane (*i.e.* a line) separating the classes, there is a p-dimensional hyperplane separating the classes, where p is the number of variables in the data.

Classes are linearly separable in the example in Figure 4.6, but this is rarely the case for actual data. However, a transformation can always be applied to the data to create linearly separable classes. Mapping all data points into the required higher-dimensional space can be computationally infeasible, especially if a space of high dimension is needed. Conveniently, the formula for calculating the weights only depends on the dot product of points in the higher-dimensional space, and we can calculate this without projecting the points into the higher-dimensional space by using kernels. A frequently used kernel that allows non-linear transformations is the Gaussian or radial basis function (RBF) kernel,

$$K(\boldsymbol{u}, \boldsymbol{v}) = \exp(-\gamma ||\boldsymbol{u} - \boldsymbol{v}||^2).$$

This takes two points $\boldsymbol{u}$ and $\boldsymbol{v}$ of the untransformed data and returns a scalar. The Gaussian kernel is defined by the hyperparameter $\gamma \in \mathbb{R}^+$. Hyperparameters are any classifier parameters that must be defined before other classifier parameters are estimated from the data.

While applying different kernel functions can theoretically always separate classes perfectly, this is rarely a good idea. If the classes are separated perfectly, the equation of the hyperplane may be highly specific to the data used, and so the SVM would not predict well for new data. A way to counteract this is to allow some points to be misclassified. This is allowed in soft-margin SVMs, which penalize the misclassified points by some cost $C \in \mathbb{R}^+$. In a

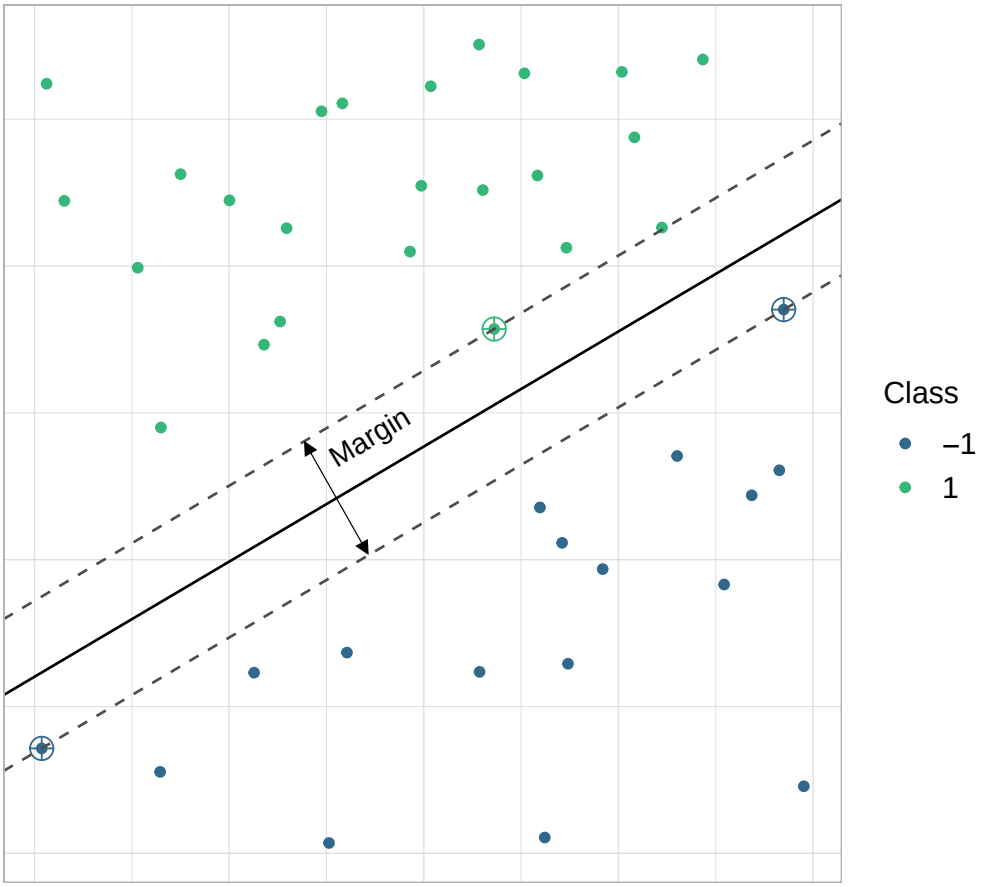


Figure 4.6: A support vector machine used to separate linearly separable data. The black line represents the optimal separating hyperplane, while the grey lines show the maximum width of the margin. Support vectors are indicated by a larger circle around the data point; note that these points are the ones closest to the hyperplane.

soft-margin classifier, the value of the hyperparameter C must also be chosen.

To determine the hyperparameter values that are best suited to the classification problem, all hyperparameters that define a SVM are optimised together using a grid search over a range of values [12]. SVMs are defined using each value or pair of hyperparameters.

After hyperparameters have been specified, the equation of the hyperplane can be found through an optimisation problem, which seeks to maximise the width of the margin while also penalizing misclassified points.

Using SVMs for prediction

The method for assigning a class to a new data point is the same for both hard- and soft-margin SVMs. Recall that the hyperplane is defined so that it best linearly separates the two classes in the higher-dimensional space. This means that once the equation of the hyperplane has been determined, the class of a new point $\boldsymbol{s}$ is given by $\text{sign}(\boldsymbol{w} \cdot \boldsymbol{s} + b)$, where

$$\text{sign}(x) = \begin{cases} 1 & \text{if } x > 0 \\ -1 & \text{if } x < 0 \\ 0 & \text{if } x = 0. \end{cases}$$

The value $\boldsymbol{w} \cdot \boldsymbol{s} + b$ is the corresponding decision value for the point $\boldsymbol{s}$. Decision values do not necessarily correspond to any distance from the hyperplane or other classes, and so they are not meaningful to use as a soft classification. Since SVMs only return a class, they are an example of hard classification.

We emphasise that there is a difference between hard and soft classification and hard-margin and soft-margin SVMs. Hard and soft classification describe whether a classifier assigns class probabilities or merely classes; SVMs provide only a hard classification, since the decision values are meaningless when it comes to describing a degree of class membership. Only the sign of the decision value defines the class of a point. Hard- and soft-margin classifiers are terms used to describe whether the hyperplane must perfectly separate classes or not.

So far, we have only considered SVMs in the context of binary classification. SVMs are capable of multiclass classification by breaking the problem down into pairwise binary classification. For K classes, $\binom{K}{2}$ classifiers are trained,

and the new point is assigned to the most frequent class allocation from the output of the binary classifiers. In the case where there is more than one class tied for the most frequent class allocation, the class that appears first in the vector of classes is chosen [12].

We direct the reader to the paper by Vapnik and Cortes [17] for a more detailed theoretical discussion, and the LibSVM manual [12] for further computational details.

4.4.3 Neural networks

Neural networks are a flexible class of models that can be applied to classification and regression problems. Here, we describe how neural networks can be applied to a multiclass classification problem.

We direct the reader to Figure 4.7 while explaining the architecture of a neural network. Neural networks are made up of layers; in Figure 4.7, each layer has a different colour. First, there is the input layer (shown in blue), which contains the data. Next, there are a number of hidden layers. In Figure 4.7 there are two hidden layers, ℓ_1 and ℓ_2, shown in different shades of grey. Finally, there is an output layer (shown in green), which contains the values required to make a classification decision.

At a more detailed level, each of these layers contains some number of nodes, represented by individual shapes in Figure 4.7. Nodes in the input layer are squares, nodes in the hidden layers are circles, and the node in the output layer is a diamond. For the input layer, each node is a feature or input variable; *i.e.* x_1, x_2, and x_3 are three different variables, and therefore there are three nodes in the input layer.

Each hidden layer has a pre-defined number of nodes. In Figure 4.7, both hidden layers are made up of four nodes, but it is not required that all hidden layers contain the same number of nodes. The number of nodes in these layers should be determined by trial and error. We use validation accuracy as a metric to determine the number of nodes, although other metrics may also be used.

Finally, we consider the number of nodes in the output layer. In regression or binary classification, only one output node is needed, as in Figure 4.7. For multiclass classification the number of nodes should correspond to the number of classes in the classification problem.

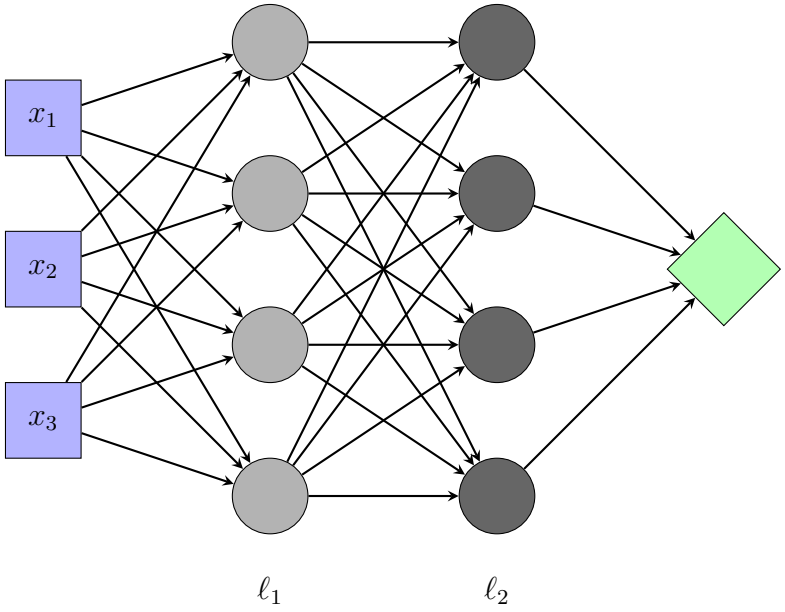


Figure 4.7: A neural network with three input variables x_1, x_2, and x_3. There are two hidden layers ℓ_1 and ℓ_2 that both have four nodes, and the output layer has one node.

We now describe how data is transformed between layers. The data undergoes a similar transformation at each node in the neural network, except for the input nodes. One such node is illustrated in Figure 4.8. The outputs from all nodes in the previous layer are denoted x_1, x_2, and x_3. These may be outputs from the input layer or a previous hidden layer. First, a linear combination is taken of the inputs to this particular node, *i.e.* if there are n nodes in the previous layer, then

$$z = \sum_{i=1}^{n} w_i x_i + w_{n+1},$$

where w_i are the weights for this particular node. The term w_{n+1} is often defined as the bias term. Weights are not required to be the same within or between layers, and so there can be a large number of weights throughout the entire network. After taking this linear combination, a nonlinear activation function σ is applied. The activation function must be nonlinear, otherwise adding layers does not boost the power of the network. This is because networks with multiple layers can be collapsed to a simple linear combination of the input variables if only linear activation functions are used.

Table 4.3 presents the most commonly used activation functions. The output

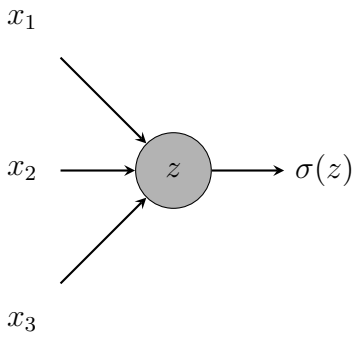


Figure 4.8: One node from a hidden layer of a neural network, where $z = w_1x_1 + w_2x_2 + w_3x_3 + w_4$ for weights w_i.

Function	Formula
Rectified Linear Unit (ReLU) [57]	$\max\{0, x\}$
Sigmoid [40]	$\frac{e^x}{1+e^x}$
Hyperbolic tangent (tanh) [40]	$\frac{e^{2x}-1}{e^{2x}+1}$

Table 4.3: Common activation functions for nodes in hidden layers.

of a node is the number obtained from taking the linear combination of the previous layer and then applying the activation function, as described in Figure 4.8. This continues until the output layer, where the data is transformed as in previous layers but a different activation function may be used. The activation function for the output layer will depend on the type of problem. Since we are using neural networks for multiclass classification, the softmax function should be used as the activation function in the output layer. The softmax output corresponding to the k^{th} class (output node) for the i^{th} observation is

$$\text{softmax}(z)_{ik} = p_{ik} = \frac{e^{z_k}}{\sum_{k=1}^{K} e^{z_k}}.$$

After the data reaches the output layer, the output of the neural network is compared to the known classes from the training data using a loss function.

If an observation is of class k, the k^{th} node of the output layer should be one, while all other nodes should be zero.

For multiclass classification, the most appropriate loss function is categorical cross entropy [14], given by

$$L = -\sum_{k=1}^{K} y_{ik} \ln(p_{ik}). \tag{4.11}$$

For the i^{th} observation, p_{ik} is the softmax output corresponding to the k^{th} class and y_{ik} is a class indicator, *i.e.*

$$y_{i,k} = \begin{cases} 1 & \text{if observation } i \text{ belongs to class } k, \\ 0 & \text{if observation } i \text{ does not belong to class } k. \end{cases}$$

It can be checked that $L \geq 0$. In the event that the i^{th} observation is perfectly classified as class k (*i.e.* $p_{ik} = 1$), and the observation is actually from class k, there is a zero contribution to the categorical cross entropy because $\log(1) = 0$. Hence, small values of L occur when $p_{ik} \approx 1$ for the correct class k.

In a forward pass through the network, the output of all nodes is calculated, as well as the loss function. Once the loss function is known, the weights are shifted in the direction that minimises the loss function. While there can be many thousands of weights in a neural network, backpropagation makes this update process feasible. Backpropagation uses variations on stochastic gradient descent to minimise the loss function, and applies the chain rule to efficiently determine in which direction the weights should be adjusted. The details of this process can be found in Rumelhart *et al.* [73], while *Deep Learning with Python* [14] provides a practical resource for building neural networks.

Using neural networks for prediction

Predicting the class of a new observation i requires one forward pass through the network to calculate the softmax outputs p_{ik}.

To obtain a hard classification y_i^* for each observation, we take

$$y_i^* = \underset{k}{\text{argmax}}\ p_{ik}.$$

4.5 Assessing Classifier Performance

4.5.1 Validation methods

Validation methods are used to evaluate the performance of a classifier. They are also necessary because they identify possible overfitting, which occurs when a classifier performs well on the data that were used to determine the parameter values of the classifier, but then generalizes poorly to previously unseen data.

Train/test splits

When fitting and assessing the performance of different classifiers, we partition the data into three different sets: training data, validation data, and testing data. This partitioning method is discussed in detail by Friedman *et al.* in *The Elements of Statistical Learning* [26].

The training data is the data used in the training process of the classifier. In the context of the classifiers presented earlier in this chapter, training data is used to determine the coefficients in MLR, the equation of the hyperplane in SVMs, and the loss function in neural networks. All observations that comprise the training data must have a corresponding class label.

Validation data is used to determine the predictive performance of the classifier under different parameterisations. Examples of this include variable selection in MLR, kernel hyperparameters for SVMs, or determining the number of layers for neural networks. All observations that comprise the validation data must also have a class label, although these class labels are not directly used in training the classifier. The predictive performance of the classifier is assessed by comparing the predicted class labels to the true class labels through some metric.

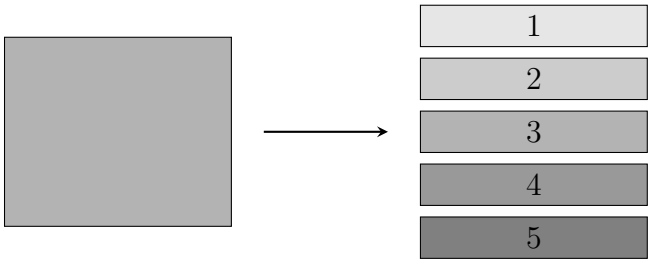
To determine how well the classifier performs on new data, it is necessary to set aside some data that will not be used in the training process. This data is referred to as test data. As for validation data, all observations that comprise the test data must have a class label, but again this class is not used as input for the classifier. The classifier will predict a class for each observation in the test data, which can be compared to the true class of the observation.

For efficiency, we can apply cross-validation instead of using fixed partitions for training and validation data. Separate test data would still be used.

Cross-validation

Cross-validation repeatedly uses different splits of training and validation data to get a general picture of how an algorithm performs. In full, it is called k-fold cross-validation, where k is the number of folds. The cross-validation process is described below.

1. Randomly split the data into k equally-sized 'folds' (illustrated here with $k = 5$ folds).



2. Set one of these folds aside as the validation data, and combine the remaining $k - 1$ folds and use as training data.

3. Implement the classification algorithm with the given training and validation data.
4. Calculate some metric to evaluate the results.
5. Repeat from Step 2 for each of the k folds.

The metric can be averaged across all folds once the cross-validation process is complete.

In this thesis, cross-validation is used to determine the value of hyperparameters. These hyperparameters include the value of λ in the LASSO and γ and C parameters for SVMs. In this application, cross-validation is only applied to the training data, splitting it into multiple training and validation sets. This means that there is always separate test data that was not used during the training process.

4.5.2 Accuracy

Accuracy is the percentage of correctly classified cases. Table 4.4 shows the actual and predicted classes for an example data set with five observations.The three different classes are taken from the shell example presented in Section 4.2.

Actual Class	Predicted Class
Type 1	Type 1
Type 2	Type 2
Type 2	Type 3
Type 1	Type 1
Type 3	Type 3

Table 4.4: Actual and predicted classes for an example data set with five observations.

In this case, four observations out of the five were classified correctly, and so the accuracy would be 80%.

The accuracy for a single classifier can vary depending on the data used to calculate the accuracy. In this thesis we consider three types of accuracy: training accuracy, validation accuracy, and test accuracy. We explained how data can be partitioned into training, validation, and test data in Section 4.5.1. To calculate the training accuracy, the classifier is used to predict a class for each observation in the training data. The training accuracy is the percentage of the predicted classes that match the true, known, classes. Similarly, the validation accuracy is based on the validation data and the test accuracy is based on the test data.

Cross-validated accuracy can be reported instead of validation accuracy. This arises from using accuracy as the metric to evaluate the results (see step 4 of the cross-validation process in Section 4.5.1).

The accuracy as calculated on the training data is likely to overestimate the performance of the classifier, since the data was used in estimating any parameters of the classifier. If the parameters are too specific to the training data, overfitting may occur. Overfitting occurs when a classifier performs well on the training data, but performs poorly when used to classify test data.

Validation accuracy is usually lower than training accuracy, since the classifier is not directly trained on the validation data. Because the validation data is used to select hyperparameters, the validation accuracy may still slightly overestimate the performance of the classifier.

Finally, test accuracy provides an unbiased estimate of classifier performance. This value best describes how the classifier will perform on previously unseen data, and is usually lower than both the training and validation accuracy.

There are some common criticisms of using accuracy as a performance metric. Consider the shell example again: if 90% of the shells in the test data were Type 3, the classifier would be 90% accurate if it blindly predicted all shells to be Type 3. Despite the high accuracy, this is clearly not a useful classification rule. To decrease the chances of this scenario occurring in our analysis, we will ensure that all classes are evenly represented in the training, validation, and test data. We will also use confusion matrices to analyse patterns of misclassification, which would clearly indicate observations being assigned to only some classes.

In this chapter we introduced methods for dimension reduction, described how the selection of a migration model can be considered as a classification problem, and then described three different classification methods. We also describe how accuracy is used to assess classifier performance. These dimension reduction and classification methods will be used in Chapter 6 to analyse the results of the simulation study.

Chapter 5

Simulation Study Design

Previous studies have investigated Aboriginal migration to Australia using a variety of different methods, such as concepts from network theory and the analysis of bathymetric data [41, 62]. A definitive consensus on the migration route taken has not yet been reached; some studies [41] suggest a northern route through the southeast Asian islands to New Guinea, while other studies [62] find support for a southern route.

Aboriginal Australian mtDNA sequences have previously been analysed to gain understanding of the peopling of Australia [50, 56]. We use uniquely provenanced Aboriginal Australian mtDNA sequence data to conduct a simulation study that investigates migration from southeast Asia into Australia. In this chapter, we describe how the migration models that define the simulation process are constructed. The coalescent simulator BayeSSC is used for simulations; further information on coalescent theory and its implementation in BayeSSC can be found in Chapter 2.

5.1 Construction of Migration Models

We define a migration route to be the geographical route taken from one population location to another, beginning in southeast Asia and ending in a region of southern Australia. Population locations used in this study include southeast Asia, southern Wallacea, New Guinea and the surrounding areas, northeastern Australia, southeastern Australia, and central southern Australia.

To obtain a set of candidate migration routes from southeast Asia to Australia, we combined migration routes that have been previously suggested in the literature. The resulting candidate migration routes are presented Figure 5.1.

For migration routes from southeast Asia to New Guinea, we consider Birdsell's northern and southern routes [6] (see Figure 5.2). Birdsell's suggested routes through the islands of southeast Asia are then joined to migration routes within Australia, which were proposed by Birdsell [5], Tindale [84], and Bowdler [10]. Each of these combinations results in a full migration route from southeast Asia to southern Australia. Migration routes are named for the author of the work that suggested the corresponding within-Australia migration path, and then numbered if there is more than one route based on the author's work.

The two additional migration routes referred to as 'Northern Sunda' and 'Southern Sunda' were suggested through discussion with J. Teixeira and G. Purnomo (personal communication, December 05, 2018). These two routes take the coastal migration suggested by Bowdler [10], and combine it with a single entry point to Australia from New Guinea. The eight candidate migration routes discussed so far encompass most reasonable migration routes between the population locations defined in this study.

We also include a ninth migration route, which we will term the 'aggregated model'. In the aggregated model, populations in the same general region are combined. These regions are denoted by black boxes in Figure 5.1, while the individual populations are still marked by black dots for consistency. No structure is assumed within the blocks; this migration route describes a general route from southeast Asia, through the islands of southeast Asia, expansion into the northern part of Australia and then an eventual migration into southern Australia.

To define a complete migration model, it is necessary to specify other parameters in addition to the migration route. Other required parameters are the time between migration events, the effective population size throughout history, any further migration between populations once population locations are inhabited, and DNA-specific parameters which pertain to the underlying biological processes. We discuss each of these parameters in turn, beginning with the time between migration events.

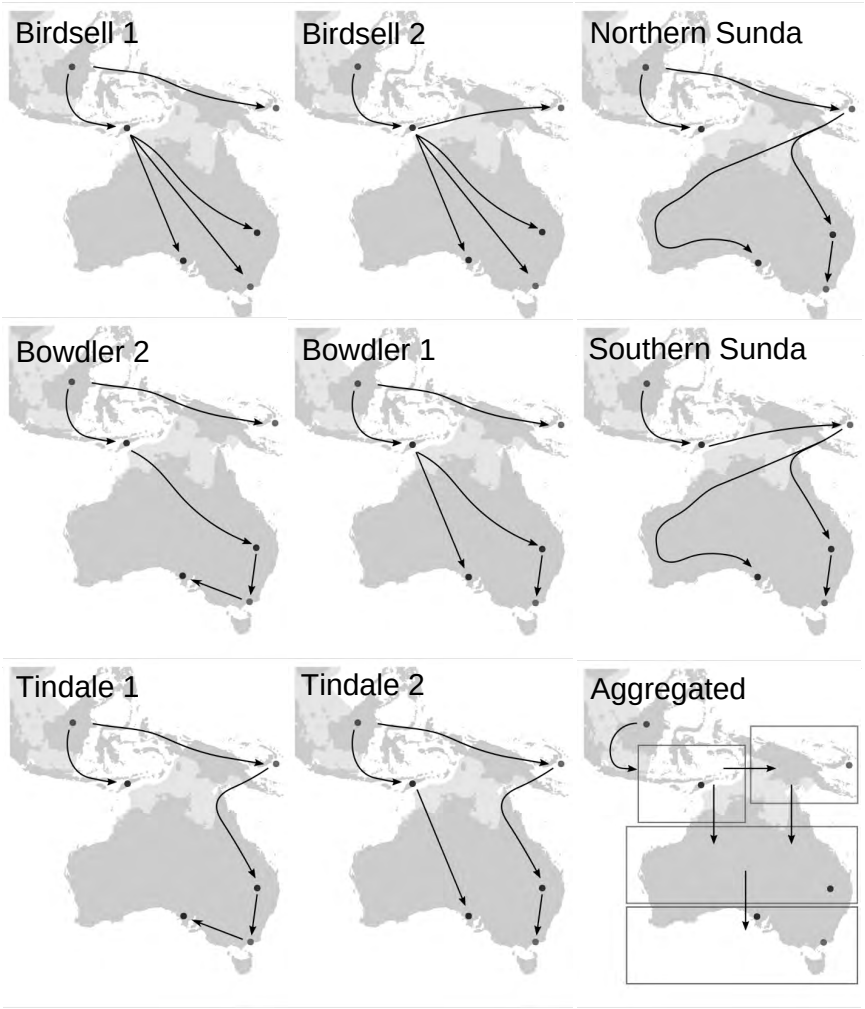


Figure 5.1: All nine candidate migration routes, with major migration events represented by solid arrows. The name of the migration model is given in the top left corner of each diagram.

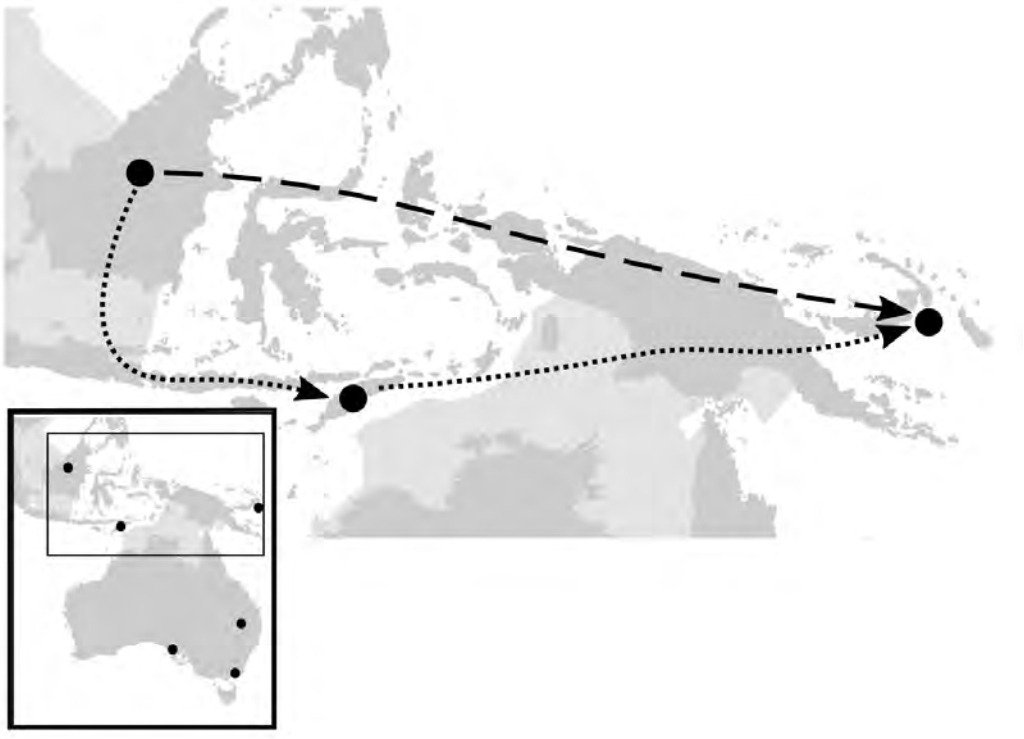


Figure 5.2: Birdsell's northern and southern routes from southeast Asia to the island of New Guinea. The dashed line represents the northern route, while the finer dotted line represents the southern route. A complete map of Australia is shown inset, with a rectangle around the region in the main image.

5.1.1 Time between migration events

Times are often measured using the kiloannum (ka), where one kiloannum is one thousand years before present. When times need to be converted to generation time, *e.g.* when defining the migration model in a BayeSSC input file, we assume a generation time of 25 years [24].

The timing of migration events is determined by the estimated timing of the initial colonisation of Sahul, and also the expected amount of time to move between population locations. The timing of the colonisation of Sahul remains uncertain, and so we allow a range of times between 50 ka and 65 ka. The endpoints of the interval are from Tobler *et al.* [85] and Clarkson *et al.* [15] respectively.

Based on archaeological estimates from the Warratyi rock shelter, we assume that humans were present in southern Australia by at least 45 ka [85]. Depending on the migration route, these dates require migration events towards southern Australia to occur every 2500 - 5000 years. We constrain population movement throughout the islands of southeast Asia by the estimated times that anatomically modern humans were present in southeast Asia. A wide interval with a high upper bound of 70 ka [94] and a lower bound of 50-55 ka [63] is considered. Migration events through the islands of southeast Asia cannot occur instantly or take a negative amount of time, so we consider a reasonably quick migration through this area to take approximately 500 years. This means that the two migration events through the islands of southeast Asia have an inter-event time of 250 years. Alternatively, there are 5000 years between 65 ka and 70 ka, so the maximum reasonable inter-event time for these migration events is 2500 years.

Prior distributions are probability distributions that describe prior knowledge about a parameter. They are traditionally used within a Bayesian framework for parameter estimation, but they can also be used in BayeSSC to account for uncertainty in parameter values. For each simulation, the value of a parameter is randomly sampled from the prior distribution. The timing of the colonisation of Sahul is modelled using a uniform prior distribution defined by the endpoints 50 ka and 65 ka.

5.1.2 Effective population size

Population size is intuitively a census population size, or a count of the number of people in the population. In the coalescent model, effective pop-

ulation size is used instead. It is important to understand the difference between these two types of population size; although census population size influences effective population size, the two are not interchangeable.

Recall that in Chapter 2 we derived the coalescent from the Wright-Fisher model. The effective population size, N_e, is the size of a Wright-Fisher population required for the genetic variability of the Wright-Fisher population to match the genetic variability of the population of interest [61]. Much like the census population size, the effective population size can change over time. Since mtDNA records the genetic history of the maternal line, we use the female effective population size, N_{e_f}.

One method for determining effective population size is through the use of skyline plots. Skyline plots were first proposed in 2000 by Pybus *et al.* [68], and produce an estimate of effective population size over time based on a coalescent tree. Early approaches gave noisy estimates of effective population size, and many methods have been developed in an effort to provide smoother and more accurate estimates. Two commonly used methods are Extended Bayesian Skyline Plots (EBSPs) and the GMRF Skyride.

The Bayesian skyline uses MCMC to determine the effective population size from the sequence data, and uses multiple change-point models to provide a smoother estimate than earlier methods [20]. GMRF Skyride uses a Gaussian Markov random field (GMRF) smoothing prior to produce a smooth estimate of effective population size through time. The GMRF Skyride has been shown to perform at least as well as Bayesian skyline methods, with both methods adequately recovering known effective population sizes [55]. Bayesian skyline plots and GMRF Skyride are both implemented in BEAST2 [8].

Although we performed an EBSP analysis in Section 3.5.2, these values are not suitable for defining our simulations. The female effective population size calculated in our EBSP analysis was for the entire population of Australia, New Guinea and southeast Asia. Due to the structure of populations in this region, where very low levels of migration are observed between populations once a population location has been inhabited [85], we expect the estimated effective population size to be biased upwards [90]. Consequently, we base our values of female effective population size on those estimated by Lippold *et al.* [46], as our sample sizes are too small to construct further EBSPs for each population location.

Lippold *et al.* used Bayesian skyline plots to reconstruct the female effective population size of New Guinea and the southeast Asian islands throughout history. Point estimates were taken from Figure 4 in Lippold *et al.*, which is

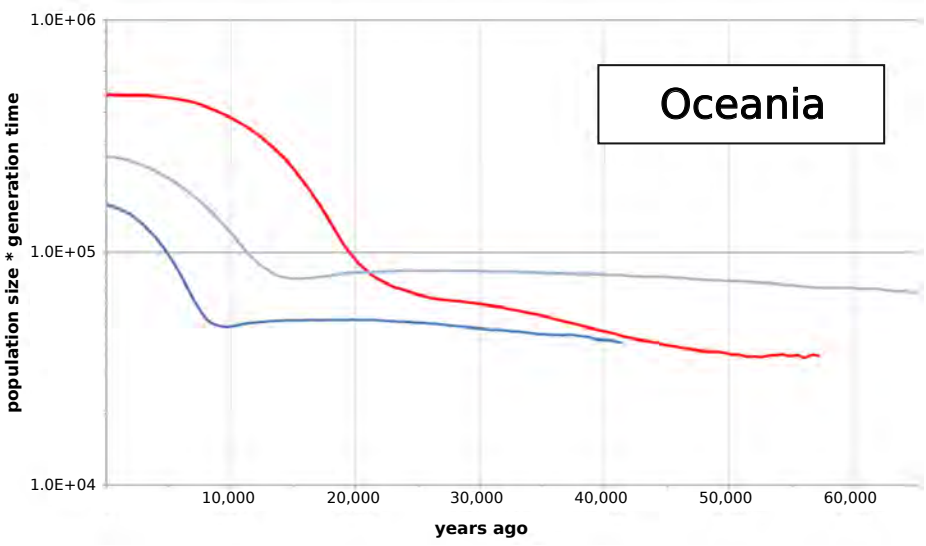


Figure 5.3: Bayesian skyline plot from Lippold *et al.* [46]. The red line shows the estimated N_{e_f} over time (calculated from mtDNA data), while the two blue lines are estimates of N_e from non-recombining Y (NRY) chromosome data. The darker blue line is based on a 'fast' mutation rate estimate for the NRY data, while the lighter blue line is based on a 'slow' mutation rate estimate. This figure was originally published in *BMC Investigative Genetics*.

presented here as Figure 5.3.

The effective population size from the skyline plot will be used for all six population locations (southeast Asia, southern Wallacea, New Guinea and the surrounding areas, northeastern Australia, southeastern Australia, and central southern Australia), as there are no studies that present reliable effective population size estimates for individual Aboriginal populations.

Exponential growth and decay correspond to linear trends on the log scale. Considered backward in time, the Bayesian skyline plot in Figure 5.4 suggests an initial constant effective population size, which decreases exponentially from approximately 8 ka to 50 ka. The two different slopes of the piecewise-linear approximation suggest two different rates of population growth. Table 5.1 gives the time and effective population size at each of the estimated change points, and then Table 5.2 gives the corresponding exponential growth rates. When calculating exponential growth rates, we assume a human female generation time of 25 years [24]. We also use the equation for exponential decay backward in time,

$$N_t = N_0 e^{-rt}, \tag{5.1}$$

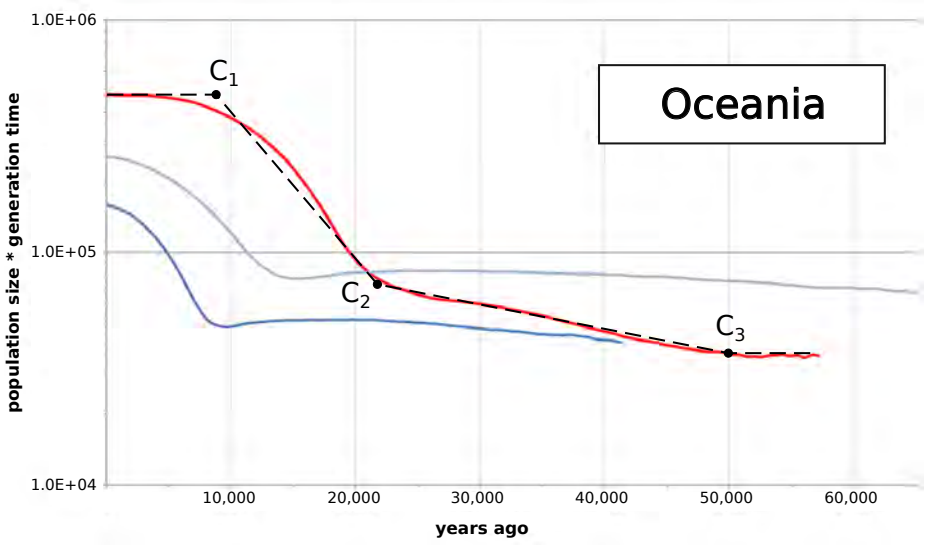


Figure 5.4: The Bayesian skyline plot from Lippold *et al.*, annotated with the piecewise-linear approximation of effective population size over time (black dashed line). Points where the population growth rate changes are represented with a point and labelled as C_i, with i increasing backward in time.

where N_0 is the initial (current) effective size of the population, t is the number of generations that have passed (backward in time), and N_t is the effective population size t generations ago. Generations are used here instead of years because BayeSSC uses generations as the timescale when simulating DNA. Equation 5.1 can be rearranged to

$$r = \frac{\log N_0 - \log N_t}{t} \tag{5.2}$$

to calculate the rate of population growth.

Change Point	Time (years ago)	Population Size * Generation Time	N_{e_f}
C_1	8500	10^4	10047
C_2	21800	$10^{4.6}$	1592
C_3	50000	$10^{4.27}$	744

Table 5.1: Values taken from the piecewise linear approximation to the Bayesian Skyline plot. The Skyline plot with the linear approximation is given in Figure 5.4. To calculate N_{e_f} from population size * generation time, we divide by the human female generation time of 25 years [24].

Interval	Interval Length (years)	Interval Length (generations)	r
$C_1 - C_2$	13300	532	3.4629×10^{-3}
$C_2 - C_3$	28200	1128	6.744×10^{-4}

Table 5.2: Interval lengths in years and generations, and the corresponding exponential growth rate over each interval. This rate was calculated using Equation 5.2 and the interval length in generations.

5.1.3 Post-settlement migration rates

So far, we have defined estimates for migration times and effective population sizes. To adequately describe the population demographic history, we still need to specify the rate at which lineages move between populations over time. We refer to this low-level ongoing migration as *post-settlement migration*, as it can only occur once people have first inhabited, or settled at, a particular population location.

As introduced in Section 2.4.4, we can describe migration between populations using a square migration matrix $M = [m_{ij}]$, where m_{ij} is the probability of an active lineage moving from population i to population j in one generation, going backwards in time. Active lineages are the lineages that are present in the tree at a particular point in time. The diagonal elements m_{ii} are meaningless, since active lineages cannot migrate to a population that they are already in. The BayeSSC manual recommends setting diagonal elements to zero to improve computational efficiency, even though the values do not affect the output of the simulations [4].

To define post-settlement migration, we need to define the elements of a 6×6 migration matrix. Matrix elements can be either single values or a prior distribution. A colour-coded migration matrix and map of corresponding population locations is given in Figure 5.5.

First, we consider populations between which ongoing low-level migration may be reasonable. For example, it does not make sense to have ongoing migration between southeast Asia and southern Australia, due to the large distance, extensive water crossings, and other populations between these two locations. We allow migration between the following population locations:

- Southern Wallacea/New Guinea,
- New Guinea/NE Australia, and
- NE Australia/SE Australia/Southern Australia.

These possible levels of migration are shown as coloured elements of the migration matrix in Figure 5.5.

Despite the comparatively close geographical distance between southeast Asia and southern Wallacea, we do not allow ongoing migration between these two populations because it would require regular crossing of Wallace's Line. Wallace's Line divides the islands of southeast Asia, with the Philip-

$$M = \begin{array}{c|cccccc} & 1 & 2 & 3 & 4 & 5 & 6 \\ \hline 1 & m_{11} & m_{12} & m_{13} & m_{14} & m_{15} & m_{16} \\ 2 & m_{21} & m_{22} & m_{23} & m_{24} & m_{25} & m_{26} \\ 3 & m_{31} & m_{32} & m_{33} & m_{34} & m_{35} & m_{36} \\ 4 & m_{41} & m_{42} & m_{43} & m_{44} & m_{45} & m_{46} \\ 5 & m_{51} & m_{52} & m_{53} & m_{54} & m_{55} & m_{56} \\ 6 & m_{61} & m_{62} & m_{63} & m_{64} & m_{65} & m_{66} \end{array}$$

Figure 5.5: Top: A map indicating the population location corresponding to each row/column of the migration matrix.
Bottom: The migration matrix, with each level of migration having a different colour. Darker colours represent a higher probability of migration on average, and no colour indicates no migration between populations. Diagonal entries are shaded, because migration from a population to itself is meaningless.

pines and Borneo on the western side of the line, and the islands of Sulawesi and Timor-Leste on the eastern side of the line. Alfred Wallace first noticed a difference in flora and fauna species on either side of this line [92]; this difference is a consequence of the deep sea levels and strong currents along Wallace's line. Ongoing post-settlement migration is therefore extremely unlikely.

Furthermore, we only allow migration to occur if a migration corridor is present in the migration route. This allows us to investigate the presence or absence of gene flow between New Guinea and Australia. For example, Bowdler's models suggest that Australia was inhabited from the southern Wallacean islands and not via New Guinea, and so in this case there would be no migration between New Guinea and northeastern Australia at all. This assumption about the presence and absence of migration rates is later explored as an extension. Figure 5.6 displays the post-settlement migration for each migration route.

While some studies suggest pronounced regionalism or female philopatry [85], *i.e.* low or non-existent migration between populations, there have been no studies that give numeric values for migration rates in the region of interest based on mtDNA. We use migration rates estimated from nuclear DNA as an approximation.

We find estimates of the effective number of migrants in each generation in Malaspinas *et al.* [50], presented here in Table 5.3. To calculate the probability of migration from the effective number of migrants between population locations, we divide the effective number of migrants by the total effective population size of the population that migrants are leaving. Point estimates for the effective population size of each population are presented in Table 5.4.

Dividing the values in Table 5.3 by the corresponding effective population sizes in Table 5.4, we get point estimates and confidence intervals for m, the probability of an active lineage moving from one population to another in one generation. These are given in Table 5.5.

Most intervals in Table 5.5 are asymmetric, but the point estimates and confidence intervals for each pair of populations are similar. We assume symmetric migration between populations, *e.g.* so that the probability of a lineage moving from New Guinea to NE Australia is the same probability as a lineage moving from NE Australia to New Guinea. This assumption reduces the dimensionality of the migration models.

Define m as the probability of an active lineage moving from one population

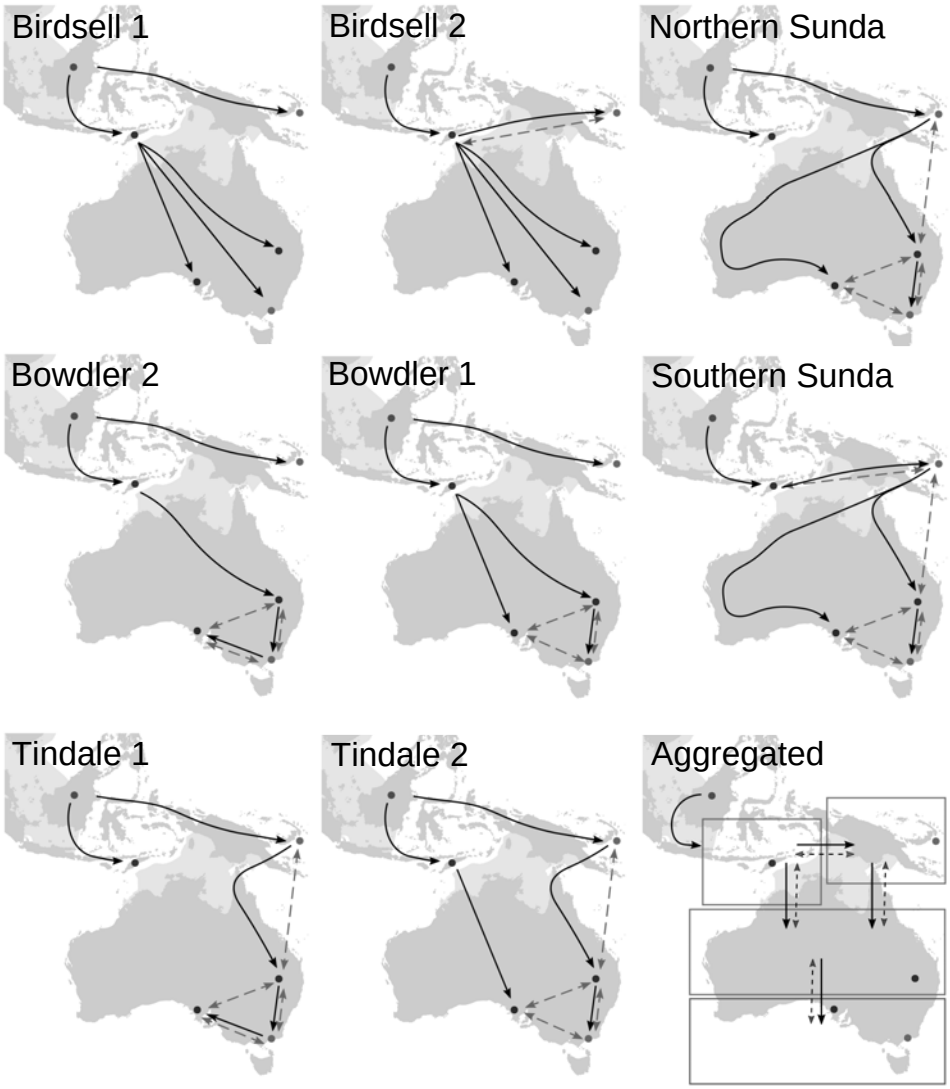


Figure 5.6: Migration patterns for each migration route. Major migration events are represented by the solid arrows as in Figure 5.1; post-settlement migration is represented by dashed arrows. The name of the corresponding migration model is given in the top left corner of each diagram.

Populations	Effective number of migrants per generation (point estimate)	95% CI
New Guinea → East Asia	0.50	(0.02, 0.76)
NE Australia → New Guinea	0.4088	(0.0009, 20.3506)
New Guinea → NE Australia	0.0151	(0.0005, 1.7293)
NE Australia → SW Australia	0.0114	(0.0005, 11.2492)
SW Australia → NE Australia	0.0031	(0.0005, 6.1587)

Table 5.3: The expected effective number of migrants moving between populations in one generation (backwards in time). The original table was given forwards in time, so directions were reinterpreted to be presented backwards in time. All values are from Malaspinas *et al.* [50]. No values were given for migration from East Asia to Papua New Guinea.

Region	$\hat{N}_e$
New Guinea	3586
SW Australia	9779
NE Australia	34214

Table 5.4: Point estimates of effective population sizes, $\hat{N}_e$ [50].

Populations	Probability of migration in one generation (point estimate)	95% CI
New Guinea → East Asia	1.394×10^{-4}	$(5.577\times10^{-6}, 2.119\times10^{-4})$
NE Australia → New Guinea	1.195×10^{-5}	$(2.631\times10^{-8}, 5.948\times10^{-4})$
New Guinea → NE Australia	4.211×10^{-6}	$(1.394\times10^{-7}, 4.822\times10^{-4})$
NE Australia → SW Australia	3.332×10^{-7}	$(1.461\times10^{-8}, 3.288\times10^{-4})$
SW Australia → NE Australia	3.170×10^{-7}	$(5.113\times10^{-8}, 6.298\times10^{-4})$

Table 5.5: The probability of an active lineage moving between populations in one generation (backward in time). All values are given to four significant figures.

Populations	Probability of migration in one generation (point estimate)	95% CI
New Guinea ↔ East Asia	8.080×10^{-6}	$(5.577\times10^{-6}, 2.119\times10^{-4})$
NE Australia ↔ New Guinea	1.195×10^{-5}	$(2.631\times10^{-8}, 5.948\times10^{-4})$
SW Australia ↔ NE Australia	3.251×10^{-7}	$(1.461\times10^{-8}, 6.298\times10^{-4})$

Table 5.6: Symmetric point estimates and intervals for the probability of migration between two populations in one generation. All values are given to four significant figures.

to another in one generation. Three 'types' of post-settlement migration are considered, as illustrated in Figure 5.5. For each of the three types of post-settlement migration, we want to create a single point estimate and a single interval for m. To create a single point estimate from the point estimates of two populations, we calculate the mean of the point estimates. To create a single interval for m between two populations, we take the most conservative values of the 95% confidence intervals for each population. This means that the endpoints of the new interval will be the smallest lower bound of the pair and the largest upper bound of the pair. The new intervals and point estimates are given in Table 5.6.

We model migration rates using a prior distribution. The corresponding author of Malaspinas *et al.* suggested that we do not use the confidence intervals as strict bounds, and instead choose a larger range (L. Excoffier, personal communication, June 13, 2019). Consequently, we decide not to use a uniform prior distributions for these parameters, as this type of prior does not allow values outside of a given range to be selected. We use gamma prior distributions for the intervals in Table 5.6, as the gamma distribution is more flexible and because it has no support on the negative real numbers.

The gamma distribution of a migration rate, m_{ij}, is defined by two parameters: a shape parameter k and a scale parameter θ. Hence is impossible to exactly fit a gamma distribution to the lower bound, upper bound, and point estimate of m. We choose to set the mode of the gamma distribution equal to the point estimate, because if simulations were ran without uncertainty on migration rates, this point estimate alone would be used as an estimate of migration rates.

We determine the parameters of the gamma distribution based on the point estimate and the upper bound of the confidence interval for the migration

Populations	Gamma parameters (shape k, scale θ)	Migration matrix
New Guinea ↔ East Asia	(22.580, 6.461×10^{-6})	m_{23}, m_{32}
NE Australia ↔ New Guinea	(1.052, 1.567×10^{-4})	m_{34}, m_{43}
SW Australia ↔ NE Australia	(1.002, 1.705×10^{-4})	m_{45}, m_{54}, m_{46}, m_{64}, m_{56}, m_{65}

Table 5.7: The parameters of the gamma distribution fitted to the intervals and point estimate of m_{ij} for each population. The corresponding elements of the migration matrix are also given (see Figure 5.5 for more details).

rate. The upper bound was used to determine the parameters because after determining the parameters, the 2.5th percentile was reasonably close to the lower bound of the confidence interval. Conversely, we also tested using the lower bound of the interval to estimate the parameters of the gamma distribution. The 97.5th percentile of this distribution was found to be very different to the upper bound of the confidence interval; hence, using the point estimate and the lower bound of the interval would have resulted in poorly-fitting gamma distributions. Fitting to the lower and upper bound of the interval did not result in a meaningful value for the mode or the mean of the distribution.

Parameters of the gamma distribution were calculated using Nelder-Mead optimisation through the `optim` function in R [69]. The optimisation function minimises the absolute distance between the mode of the gamma distribution and the point estimate, and also the 97.5th percentile of the gamma distribution and the upper bound of the confidence interval. The mode of a gamma distribution parameterised by shape k and scale θ is $(k-1)\theta$.

The parameters of a gamma distribution are estimated for each pair of populations in Table 5.6. These parameters, as well as the elements of the migration matrix in Figure 5.5 that the prior distributions correspond to, are presented in Table 5.7.

For the purposes of simulation, the population demographic history is completely described by the sample sizes, migration routes, timing between migration events, effective population sizes, and rates of post-settlement migration. To fully describe the simulation process of DNA, we also need to define the length of the simulated DNA and other parameters that describe how DNA evolves over time.

Population ID	Population Location	Sample Size
0	Southeast Asia	12
1	Southern Wallacea	8
2	New Guinea	41
3	NE Australia	41
4	SE Australia	7
5	Central southern Australia	22

Table 5.8: Sample sizes for each population location.

5.1.4 Further parameters for DNA simulation

First, we need to consider the number of DNA sequences to be generated for each population location. This will be the number of observed mtDNA samples that we have at each population location, as determined in Section 3.1.1. The sample sizes are presented in Table 5.8.

Next, we consider how long the simulated DNA needs to be. Once again, this should match the observed mtDNA samples, which have a length of 15447 base pairs (the length of the coding region of mtDNA).

A substitution rate is required to define how quickly substitutions accumulate; we use the substitution rate of 1.57×10^{-8} substitutions per site per year from Fu *et al.* [28].

We will also need to select a substitution model to describe how DNA accumulates substitutions over time; for this we use the software package ModelGenerator [42]. ModelGenerator compares a comprehensive range of substitution models by AIC [2], AIC2, and BIC [75]. AIC2 was created for ModelGenerator based on empirical findings [42]. These three criteria calculate the goodness of fit of a model with n estimated parameters using the equation $-2 \times \text{loglikelihood} + kn$. For AIC, $k = 2$; for AIC2, $k = 5$; and for BIC $k = log(n)$. Hence AIC penalizes complexity the least, while BIC penalizes complexity the most.

ModelGenerator also estimates the proportion of invariant sites (denoted by the flag `+I`) , and whether a gamma distribution should be used to allow different substitution rates at different sites (denoted by the flag `+G`).

Ranking	AIC	AIC2	BIC
1	`TVM+I+G` AIC = 86780.08 loglik = -42026.04	`TVM+I+G` AIC2 = 90872.08 loglik = -42026.04	`HKY+I+G` BIC = 97294.23 loglik = -42034.22
2	`GTR+I+G` AIC = 86785.48 loglik = -42027.74	`HKY+I+G` AIC2 = 90873.44 loglik = -42034.22	`K81uf+I+G` BIC = 97300.83 loglik = -42032.66
3	`K81uf+I+G` AIC = 86789.32 loglik = -42032.66	`K81uf+I+G` AIC2 = 90875.31 loglik = -42032.66	`HKY+I+G` BIC = 97303.92 loglik = -42034.20

Table 5.9: The three highest-ranked substitution models according to AIC, AIC2, and BIC. The `+I` flag represents invariant sites, while the `+G` flag represents the inclusion of gamma rate variation. The `uf` appended to the K81 substitution model indicates unequal base frequencies.

The three highest-ranked substitution models for each criterion from the ModelGenerator output are given in Table 5.9. Following the advice of Luo *et al.* [49], we use BIC to select a substitution model. The HKY substitution model cannot be implemented directly in BayeSSC; we approximate it using the Kimura two parameter (K2P) substitution model, which is equivalent to the HKY model with equal base frequencies enforced. The K2P substitution model allows transitions and transversions to occur at different rates, and is parameterised by the transition bias. For the definitions of all substitution models in Table 5.9, please see Appendix A.2.

In Section 2.4.5 we introduced the concept of different sites having different substitution rates. This rate variation between sites is modeled by a discrete gamma distribution, which has some number of classes n and a shape parameter α. The ModelGenerator output best supported seven rate classes and estimated α to be 0.53.

The transition/transversion ratio is estimated to be 18.44 by ModelGenerator, which is equivalent to a transition bias of 0.9485 (to four decimal places).

All DNA-specific parameters other than sample sizes for each population location are presented in Table 5.10 for completeness. For more a complete discussion of DNA-specific parameters, please refer to Chapter 2, specifically Sections 2.1 and 2.4.5.

Parameter	Value
Sequence Length	15447 bp
Substitution Rate	1.57×10^{-8} substitutions/site/year [28]
Substitution Model	Kimura two-parameter
Transition bias	0.9385
Gamma shape parameter α	0.53
Number of rate classes	7

Table 5.10: Other parameters required to define a full migration model in BayeSSC.

5.2 Model Summary

In this chapter, we have defined all necessary input parameters for DNA simulation using BayeSSC. We constructed potential migration routes from previous studies on migration in this region. Considering two population genetics studies, we assume that the colonisation of Sahul occurred between 50 ka and 65 ka; the timings between migration events are then defined based on the estimated time that humans were present in southeast Asia [94, 63] and archaeological evidence for when humans reached southern Australia [85]. Estimates for post-settlement migration (small amounts of migration between population locations) were calculated from a parameter estimation study using nuclear DNA [50], as no rates based on mtDNA could be found for the region of interest.

The female effective population size throughout history was determined from the Bayesian skyline analysis of Lippold *et al.* [46], because the estimates from our EBSP analysis in Section 3.5.2 were calculated at a much coarser geographical scale.

Finally, we discussed DNA-specific parameters such as the substitution model and gamma rate variation between sites, and described how ModelGenerator was used to estimate these parameters.

The models designed here will be used to simulate DNA using BayeSSC; Chapter 6 presents the results of applying the classification methods described in Chapter 4 to the summary statistics of the simulated DNA.

Chapter 6

Simulation Study Results

In this chapter, we present the results of the simulation study designed in Chapter 5. We begin by describing our simulated data and the method used to summarise the observed mtDNA alignment, followed by two-dimensional visualisations of these observed and simulated data. We then describe the process used to train each of the classifiers, and compare how the classifiers perform on simulated data. Next, we apply all classifiers to our observed data. Finally, we conduct additional validation analyses to check the stability of our results, and then summarise the findings of our simulation study.

6.1 Simulation of mtDNA

For all simulations we used Bayesian Serial Simcoal (BayeSSC) to simulate DNA sequence alignments, and to automatically calculate the associated summary statistics. We performed 15,000 simulations for each migration model, which has been successfully used in the past by Llamas *et al.* [48], who also used BayeSSC to perform a simulation study. Performing 15,000 simulations for each migration model resulted in a total of 135,000 realisations of sequence alignments, from which the 126 summary statistics of interest were calculated. The summary statistics that were used in this study are described in Appendix A.

Data is recorded as in Table 6.1 for each analysis. Each row corresponds to one simulation, and the columns are the associated within-population and between-population summary statistics. For each set of summary statistics, we also record the migration model that the corresponding DNA sequence

Within-population summary statistics			Between-population summary statistics			
$\hat{S}_0$	...	D_6	$\hat{k}_{01}$	...	F_{ST45}	Migration Model
58	...	-2.08	31.17	...	0.62	Birdsell 1
64	...	-1.69	57.19	...	0.42	Birdsell 2
⋮	⋮	⋮	⋮	⋮	⋮	⋮
89	...	-1.59	67.79	...	0.13	Tindale 2

Table 6.1: The $135,000 \times 126$ matrix of simulated summary statistics, with a 127^{th} column giving the associated migration model. Each row corresponds to a simulation, while each column corresponds to a different summary statistic. The numbers given are taken from the actual summary statistics calculated for the simulated DNA alignments.

alignment was simulated under. The migration models are labelled 'Birdsell 1', 'Birdsell 2', 'Tindale 1', 'Tindale 2', 'Bowdler 1', 'Bowdler 2', 'Northern Sunda', 'Southern Sunda', and 'Aggregated'. These migration models were described in detail in Chapter 5, and are shown here in Figure 6.1.

We discarded all measurements of nucleotide diversity before training any classifiers, as it is a scalar multiple of the average number of pairwise differences $\hat{k}$ (see Appendix A.1.1).

From this point, we refer to the summary statistics of the simulated DNA sequence alignment as 'simulated summary statistics', and the summary statistics of the observed mtDNA alignment as 'observed summary statistics'.

Custom R [69] scripts were used to calculate the observed summary statistics, using the R packages ape [65] and pegas [64].

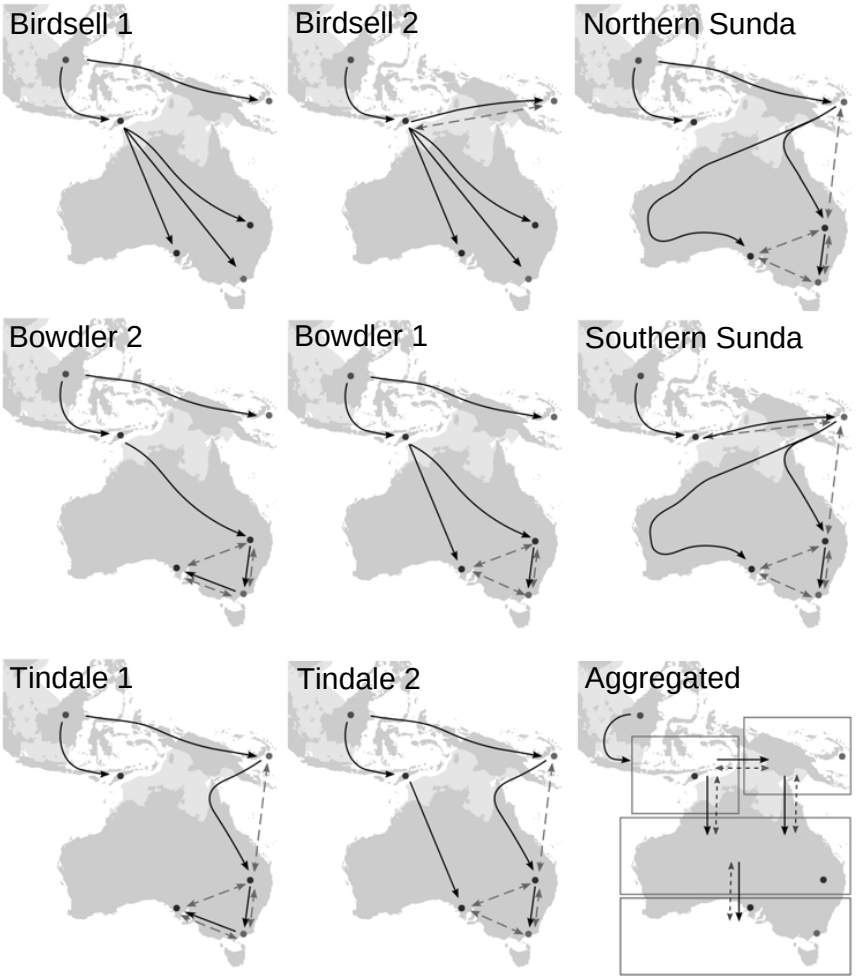


Figure 6.1: Migration patterns for each migration route. Major migration events are represented by solid arrows and post-settlement migration is represented by dashed arrows. The name of the corresponding migration model is given in the top left corner of each diagram. The boxes in the aggregated model show the distinct geographical regions that are considered (*i.e.* the two southern Australian populations are aggregated).

6.2 Visualising Observed and Simulated Summary Statistics

We used both principal component analysis (PCA) and uniform manifold approximation and projection (UMAP) to visualise the distribution of the simulated summary statistics and the relative location of the observed summary statistics. Recall that PCA is a linear dimension reduction method that identifies orthogonal directions of maximum variance in the data, while UMAP is a recently developed algorithm for non-linear dimension reduction that aims to capture non-linear relationships in a lower-dimensional representation.

Figure 6.2 shows the PCA of the observed and simulated summary statistics, while Figure 6.4 shows the UMAP dimension reduction of the same data. Recall that UMAP requires some parameters to be pre-selected. We chose to use two components (*i.e.* two dimensions) for easy visualisation, and then tested all possible combinations of either 5, 10, 20, 50 or 100 nearest neighbours and a minimum distance of 0.1, 0.3, 0.5, 0.7, or 0.9. All combinations of parameters produced similar results, and the full set of plots are given in Appendix C.2.1. For the UMAP dimension reduction in Figure 6.4, the `random_state` parameter was set to 71817.

Simulated summary statistics are coloured according to the migration model that they were simulated under; the observed summary statistics are given by a black star. All points are slightly transparent to better visualise how migration models differ or overlap. Due to the large number of points, we also present facetted plots in Figures 6.3 and 6.5, which split Figures 6.2 and 6.4 by migration model.

We notice that in both Figures 6.2 and 6.4 the aggregated model is clearly distinguishable from all other migration models, while the remaining migration models appear to overlap significantly. In both the PCA and the UMAP dimension reduction, the observed summary statistics (given by a black star) lies within the range of simulated summary statistics. It appears to be extremely unlikely that the observed summary statistics are best explained by the aggregated model.

Other than the noticeably different summary statistics from the aggregated model, Figure 6.3 confirms that there is no visible difference between the PCA of simulated summary statistics from different migration models.

By further examining the facetted UMAP dimension reduction in Figure 6.5,

we notice some differences between the simulated summary statistics from different migration models. The Birdsell models appear to have less points in the leftmost region of the point cluster, and the Bowdler models appear to have less points in the top-left region of the point cluster. Despite these slight differences in the dimension reduction of different simulated summary statistics, the observed summary statistics still lie within the set of simulated summary statistics for all models except the aggregated model, and so we cannot rule out any further migration models.

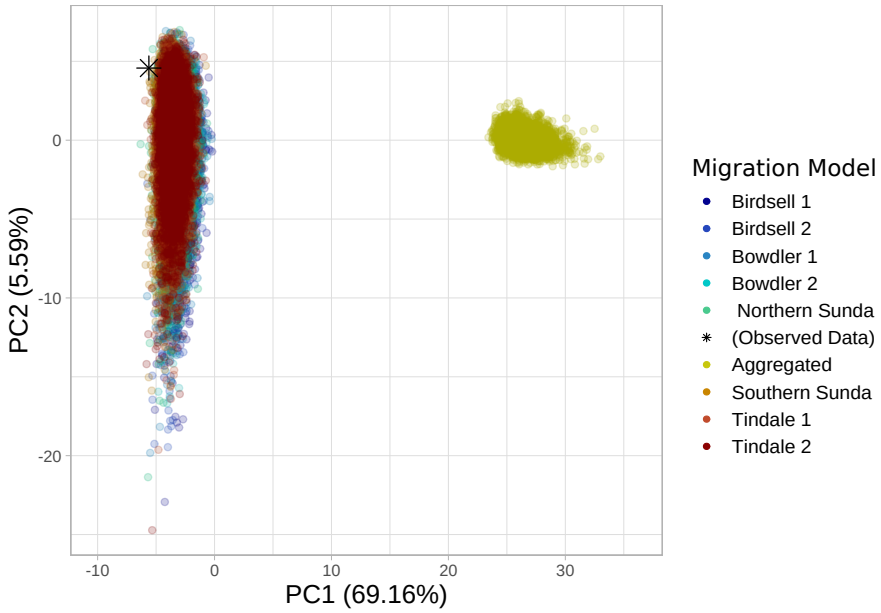


Figure 6.2: PCA of the simulated and observed summary statistics. The percentage of variance explained by each principal component is given in the axis labels.

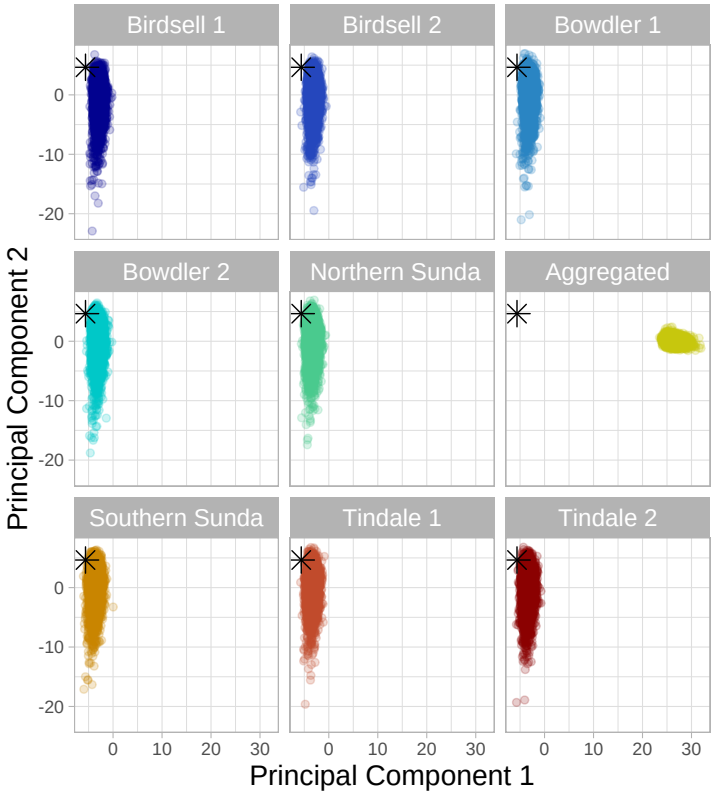


Figure 6.3: The PCA from Figure 6.2 facetted by migration model. The observed summary statistics are given as a black star in each panel.

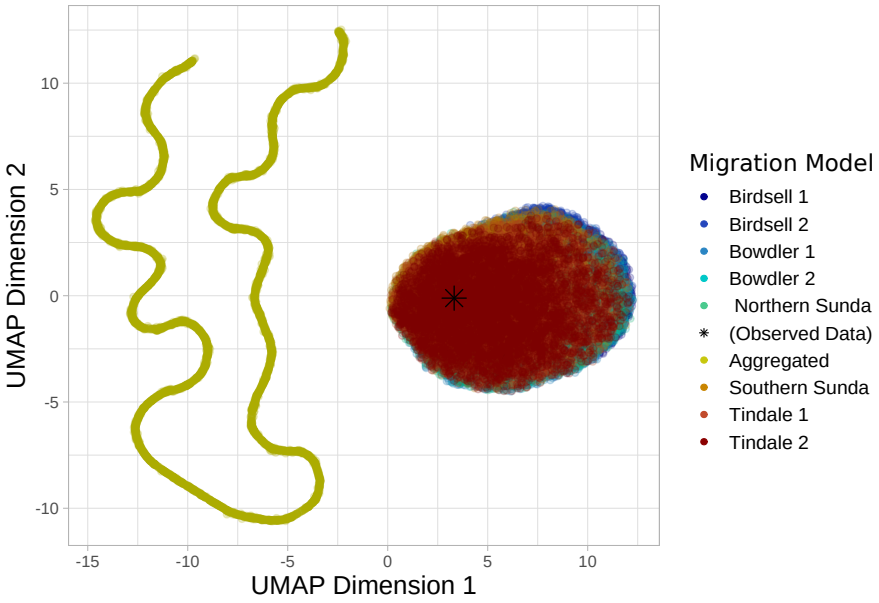


Figure 6.4: UMAP dimension reduction of the simulated summary statistics, with the observed summary statistics projected into the same space. We ran the UMAP algorithm using 100 nearest neighbours, a minimum distance of 0.5, and a random state of 71817.

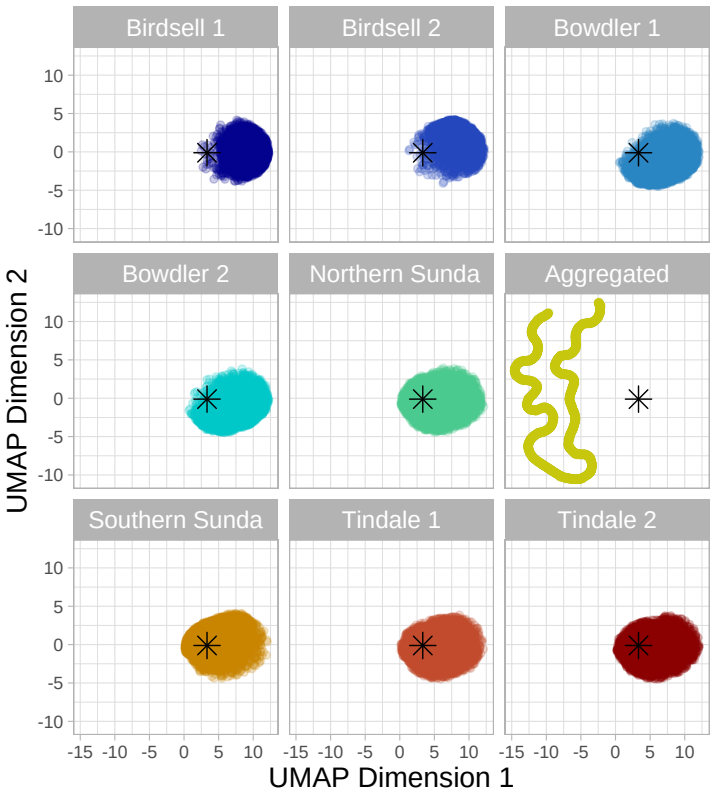


Figure 6.5: The UMAP dimension reduction from Figure 6.4 facetted by migration model. The observed summary statistics are given as a black star in each panel.

6.3 Training and Evaluating Classification Methods

The aim of our simulation study is to explore which, if any, of the migration models adequately explain the observed summary statistics. If the observed summary statistics closely resemble those simulated under one or more of the candidate migration models, as determined using classification methods, we can conclude that the observed DNA is likely a product of a similar demographic history.

This is a multiclass classification problem which therefore requires the use of multiclass classification methods, such as multinomial logistic regression (MLR), support vector machines (SVMs) and neural networks. We use multiple methods due to the complexity of the problem: consistent results from all classifiers would provide stronger evidence for our results. Using both linear and non-linear classification methods also allows us to determine whether non-linear classification methods are better suited to this classification problem.

6.3.1 Multinomial logistic regression

When applying MLR to a multiclass classification problem with k classes, $k-1$ logistic regression models are fitted to the data before the probability of an observation belonging to each class can be predicted.

Before applying MLR, we center and scale each column of summary statistics so that they have a mean of zero and a standard deviation of one. This is performed according to recommendations by the authors of the nnet package [89], which we used to fit our MLR. The maximum number of iterations performed to determine the coefficients of each model was increased to 10^5 to allow the model estimates to converge, and the maximum number of weights was increased to 10^7 so that the total number of predictor variables added was not truncated.

We partitioned the simulated summary statistics so that 70% were used as training data and 30% were reserved as test data. This partitioning was performed within classes so that there were equal proportions of each class in the training and test data.

To select appropriate predictor variables for MLR we used both forward

selection and LASSO, which were described in Section 4.4.1.

Forward selection

Recall that forward selection begins by using only an intercept term, *i.e.* no summary statistics, to predict the migration model. One statistic at a time is then added as a predictor variable until the model fit, as measured by AIC, ceases to improve.

The final regression model from forward selection had 55 predictor variables, and resulted in 50.6% accuracy on test data. The training accuracy was 50.9%, and so the low test accuracy is not a consequence of overfitting. We note that for nine classes, random guessing would result in a classifier that was 11.11% accurate. While this classifier does not have high accuracy, it at least performs better than random guessing.

Predictor variables that were included in the final regression model were mostly F_{ST} , segregating sites, pairwise differences, and the number of private alleles, as well as two pooled haplotype diversity statistics and three Tajima's D statistics. A full list of predictor variables and coefficients is given in Appendix C.2.2.

LASSO

Recall from Section 4.4.1 that LASSO penalizes the size of regression coefficients, and results in regression coefficients being shrunk towards zero. We implemented LASSO with the glmnet package [27]. The same training and test data partitions were used as in forward selection.

In addition to calculating the regression coefficients, the parameter that controls the level of penalization (λ) must also be selected in LASSO. We use the recommended default λ sequence provided by glmnet, which explores 100 values between a near-zero value and the highest possible λ. The λ values close to zero result in very little penalization and therefore minimal reduction in predictor variables, while the maximum possible λ value would result in no predictor variables.

We perform five-fold cross-validation to determine the best value of λ. After the cross-validation procedure is complete, two candidate λ values are returned. The value of λ that minimises the cross-validated error is given, along with the value of λ corresponding to the simplest model within one

Variable	Value	Training Accuracy	Test Accuracy
λ_{min}	3.77×10^{-5}	50.5%	50.4%
λ_{1se}	9.56×10^{-5}	50.5%	50.4%

Table 6.2: The λ values that minimise cross-validated error (λ_{min}), and that correspond to the most parsimonious model within one standard error (λ_{1se}). The classification accuracies on the training and test data are also given.

Migration Model	number of predictors (λ_{min})	number of predictors (λ_{1se})
Birdsell 1	63	53
Birdsell 2	66	57
Bowdler 1	58	44
Bowdler 2	56	49
Northern Sunda	51	38
Southern Sunda	63	51
Tindale 1	48	47
Tindale 2	51	38
Aggregated	6	7

Table 6.3: The number of predictor variables in the optimal regression model for each migration model (*i.e.* each class) for the cases $\lambda = \lambda_{min}$ and $\lambda = \lambda_{1se}$. The regression models using λ_{min} nearly always have at least as many predictor variables as the regression models using λ_{1se}.

standard error. We denote these λ_{min} and λ_{1se} respectively. The variable λ_{1se} is based on the 'one standard error' rule proposed by Friedman, Hastie, and Tibshirani [26]. We compare the two λ values in Table 6.2, along with the test and training accuracy of the regression performed using each value of λ.

Due to the negligible difference in test accuracy between the two λ values, we performed regression using $\lambda = \lambda_{1se}$. The regression models using λ_{1se} have fewer predictor variables, on average, than the models using λ_{min}; Table 6.3 presents the number of predictors in the regression model for each class. A key difference between the implementations of forward selection and LASSO is that in LASSO, each of the regression models that comprise the multinomial logistic regression can use different predictor variables. In forward selection, the same predictor variables are used in all logistic regression mod-

els. A full list of all included predictor variables for the regression models under λ_{1se} is given in Appendix C.2.3.

6.3.2 Support vector machines

Support vector machines (SVMs) aim to find the optimal separating hyperplane between classes, and do this by transforming the data into a higher-dimensional space. Important considerations when specifying a SVM model are the choice of kernel and the associated hyperparameters. The kernel specifies how the data is transformed, and the hyperparameters define parts of the SVM such as the kernel parameters and the misclassification penalty. This is in contrast to the parameters of the SVM, which define the equation of the hyperplane [17].

In the following analysis, we begin by considering a linear kernel which is defined by one hyperparameter, the cost $C \in \mathbb{R}^+$. We then decide whether a linear or non-linear (Gaussian) kernel should be used by comparing the results of the linear kernel to the results of the Gaussian kernel. The Gaussian kernel is parameterised by two hyperparameters, gamma $\gamma \in \mathbb{R}^+$ and the cost $C \in \mathbb{R}^+$, and is also known as the radial basis function (RBF) kernel. SVMs are fitted in R using the e1071 package [53].

For both types of kernels, the corresponding hyperparameters are optimised by using a grid search algorithm. As per the recommendations in Hsu *et al.* [35], we performed a grid search over exponentially growing values of C and γ. All combinations of the ranges $C = \{2^{-5}, 2^{-3}, 2^{-1}, 2, 2^3, 2^5\}$ and $\gamma = \{2^{-5}, 2^{-3}, 2^{-1}, 2, 2^3\}$ are considered. The grid search is conducted within a cross-validation framework using five folds. The final hyperparameters for each SVM are given in Table 6.4.

It was impractical to use 70% of the data for training due to computational limitations. To assess the performance of SVMs, we instead create a learning curve. Learning curves are a plot of the accuracy of the classifier against the percentage of the total data used to train the classifier, for different percentages of training data. Learning curves for SVMs using linear and Gaussian kernels are given in Figure 6.6.

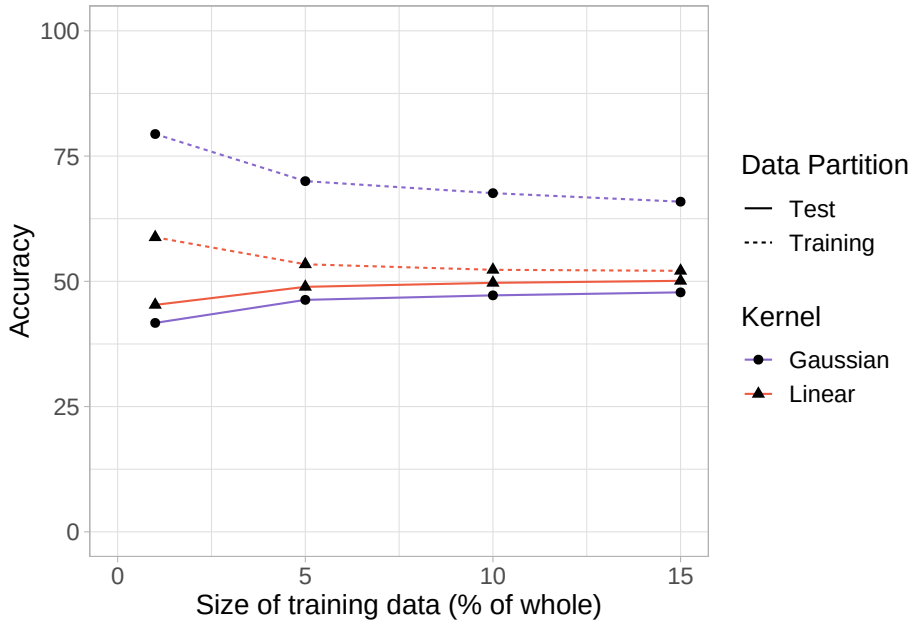


Figure 6.6: Training and test accuracy for SVMs with linear and Gaussian kernels against the percentage of the complete dataset used as training data. The test accuracy increases with the size of the training data for both kernels, while the training accuracy decreases with the size of the training data.

Kernel Type	Size of Training Data	C	γ	Test Accuracy	Cross-Validated Accuracy	Training Accuracy
Linear	1%	0.5	-	45.3%	44.8%	58.8%
Linear	5%	32	-	48.9%	47.8%	53.4%
Linear	10%	1	-	49.7%	50.1%	52.3%
Linear	15%	32	-	50.1%	49.7%	52.1%
Gaussian	1%	1	2^{-5}	41.7%	41.9%	79.4%
Gaussian	5%	1	2^{-5}	46.3%	45.9%	70.0%
Gaussian	10%	1	2^{-5}	47.2%	47.1%	67.6%
Gaussian	15%	1	2^{-5}	47.8%	47.6%	65.9%

Table 6.4: The test, cross-validated, and training accuracies of the SVMs for each combination of kernel and training data size.

In Table 6.4 we present the test, cross-validated, and training accuracies for each classifier, along with the best hyperparameters for each SVM kernel as found by a grid search. The test accuracy is the accuracy of the classifier evaluated on the test data, the training accuracy is the accuracy of the classifier evaluated on the whole training set, and the cross-validated accuracy is the accuracy found through 5-fold cross-validation when tuning the hyperparameters.

From the learning curves in Figure 6.6 and the SVM results in Table 6.4, it is clear that there is no benefit to using the more complex Gaussian kernel. It also seems likely that using a Gaussian kernel results in considerable overfitting, due to the difference between training and test accuracies. We note that small values of γ and C are selected for the Gaussian kernel. Overfitting is often coupled with large values of γ and C, because larger γ values allow more flexibility when determining the separating hyperplane and larger C values heavily penalize misclassification [66].

Since using the Gaussian kernel did not produce better results, we now consider the training and test accuracy for the linear kernel. From Figure 6.6 we see that test accuracy improves as the amount of training data increases, although the increase is less pronounced as more data is used for training. From Table 6.4, we also notice that as the amount of training data increases, the difference between the test accuracy and the training accuracy decreases. It is therefore likely that the effects of overfitting decrease as greater amounts of data are used as training data. Furthermore, the test accuracy only increased by 0.4% when the size of the training data was increased from 10% to 15%. It is likely that using greater amounts of training data will provide little benefit. Hence, we do not consider using more than 15% of the total data as training data, and will apply a linear kernel trained on 15% of the data to the observed summary statistics.

6.3.3 Neural networks

The theory of neural networks was previously described in Section 4.4.3. We used the keras library [13] in Python 3.7.2 [87] to train neural networks on the simulated summary statistics. The four different neural network architectures used are described in Figure 6.7. All architectures had 126 nodes in the input layer, because there were 126 summary statistics. The architectures also had the same number of nodes in the output layer. We used nine nodes because this is a multiclass classification problem with nine classes.

Consider Figure 6.7. Each architecture is represented as a labelled flow chart. Each box represents a layer, with the number of nodes given in brackets.

Architecture 1 is a neural network with two hidden layers.

Architecture 2 has a larger capacity, since it has an extra layer and more nodes in each layer (Figure 6.7.2). If the accuracy is limited by the network size of Architecture 1, we expect the accuracy of this neural network to be greater than the accuracy of Architecture 1.

Architecture 3 builds on Architecture 2, and introduces dropout into the network. Dropout is a regularization technique that makes some proportion of the nodes in a layer inactive, which reduces the chance of overfitting [79]. The proportion of nodes that are rendered inactive is given by the dropout rate. The effect of dropout is different to the effect of using a smaller neural network, since the nodes affected by dropout change with each forward pass through the network. If Architecture 2 results in low accuracy due to overfitting, we expect performance to improve in Architecture 3.

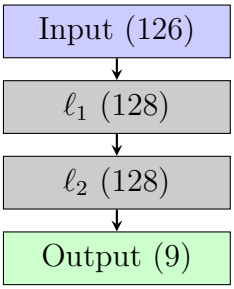
Architecture 4 is the same as Architecture 3, but the dropout rate is lower in both hidden layers where dropout is applied. This tests whether a high dropout rate unnecessarily hinders accuracy.

The data partitions for neural networks are similar in concept to the partitions used for support vector machines, in that we use training, validation, and test data. Initially, 30% of the data is set aside as test data. The remaining data are then partitioned into training and validation sets using a 70/30 split. We used the `train_test_split` function in scikit-learn [66], with the `random_state` seed 5467.

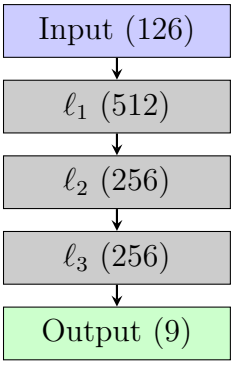
To find the optimal weights for each network, each network is trained for 1000 passes through the entire training dataset, *i.e.* for 1000 epochs. We use a batch size of 32, which means that the weights are updated after 32 observations have been processed. To minimise overfitting, we apply checkpointing through the model training process. Checkpointing calculates the validation accuracy every epoch, and only saves the weights if the validation accuracy has increased.

The accuracy for the optimal model found under each network architecture is given in Table 6.5. Architecture 2 has the best performance, with a test accuracy of 48.7%, validation accuracy of 49.4%, and a test accuracy of 49.1%. The similarity between the accuracies calculated on each type of data suggest that overfitting did not negatively affect the classifier performance.

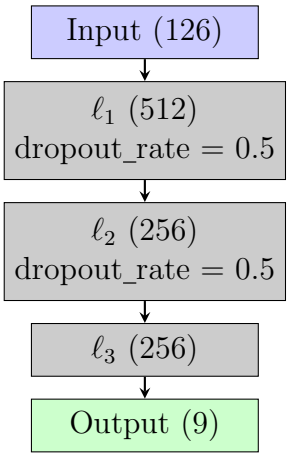
The minimal increase in accuracy from Architecture 1 to Architecture 2 sug-



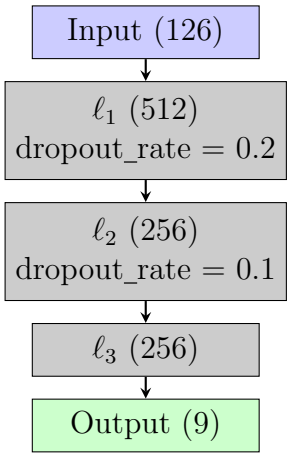
6.7.1: Architecture 1



6.7.2: Architecture 2



6.7.3: Architecture 3



6.7.4: Architecture 4

Figure 6.7: Four different neural network architectures that are trained on the simulated summary statistics. All neural networks have an input layer with 126 nodes (the number of variables), and an output layer with 9 nodes (the number of classes).

Architecture	Training Accuracy	Validation Accuracy	Test Accuracy
1	48.6%	49.0%	48.3%
2	49.4%	49.4%	49.3%
3	45.4%	45.5%	45.1%
4	47.1%	47.6%	47.0%

Table 6.5: Classification accuracies for each neural network architecture. Architectures are described in Figure 6.7.

gests that increasing the capacity of the network through adding nodes and another hidden layer had minimal effect, and so we should not expect significant benefits from using a network with greater capacity than Architecture 2. As already suggested from the accuracies of Architecture 2, we do not expect that overfitting is a problem. The lack of increase in accuracy when dropout is added in Architecture 3 further supports this assumption. We see that lower dropout rates in Architecture 4 produce higher accuracies than Architecture 3, but these are still below the Architectures with no dropout. Consequently, we will use Architecture 2 for prediction.

6.3.4 Evaluating classification methods using confusion matrices

Confusion matrices were used to investigate misclassification patterns for all classifiers. In Table 6.6, we present the confusion matrices for MLR with forwards selection; in Table 6.7 we present the confusion matrices for MLR with LASSO; in Table 6.8 we present the confusion matrices for the linear SVM trained on 15% of the data; and in Table 6.9 we present the confusion matrices for the neural network with architecture 2.

Unsurprisingly, given the PCA and UMAP results, we see that the aggregated model was always correctly classified for both the training and test data for all classifiers. For migration models other than the aggregated model, we notice two main patterns of misclassification. The confusion matrices for MLR with forward selection, MLR with LASSO, and SVMs all have a similar misclassification pattern, while the confusion matrices for the neural network have a different misclassification pattern. First, we will discuss the misclassification patterns seen for MLR with both types of variable selection

and for SVMs.

In the following discussion we refer to the confusion matrix for the test data, although we note that the confusion matrices for the training and test data are very similar. We give specific percentage values based on the confusion matrix of the test data for MLR with forward selection, but we note that the confusion matrices for the training and test data for MLR with forward selection, MLR with LASSO, and SVMs are all very similar (within 0.03 when comparing all confusion matrices elementwise).

Birdsell 1 and 2 were mostly classified correctly, with 72% and 76% of the observations of the respective migration models being classified correctly. 14 % of the true Birdsell 1 observations were misclassified as Birdsell 2, and 14% of the true Birdsell 2 observations were misclassified as Birdsell 1. The remaining 14% of Birdsell 1 observations and 10% of Birdsell 2 observations were misclassified as other migration models.

Bowdler 1 and 2 are classified correctly 34% and 39% of the time respectively, and are also commonly misclassified as each other (28% of Bowdler 1 observations are misclassified as Bowdler 2, while 26% of Bowdler 2 observations are misclassified as Bowdler 1) . Both of these classes are also occasionally misclassified (18% Bowdler 1 and 16% of Bowdler 2 observations) as Birdsell 1. Southern Sunda is classified correctly for 67% of observations; after the Birdsell migration models, this is the migration model with the third-highest percentage of correct classifications. Finally, we note that observations simulated under the Northern Sunda, Tindale 1, and Tindale 2 migration models are not as likely to be classified correctly, with only 17%, 30% and 20% of observations respectively being classified correctly. The Northern Sunda and Tindale models are also somewhat likely to be misclassified as each other, and are less likely to be misclassified as either of the Birdsell models.

We now discuss the misclassification patterns in the confusion matrices for the neural network (Table 6.9). Once again, we refer to the confusion matrix for the test data, and emphasise that the confusion matrices for the test and training data are near identical. Note that the Northern Sunda migration model no longer appears as a column with the other predicted models, which means that no observations were classified or misclassified as Northern Sunda. Summary statistics that were simulated under the Northern Sunda migration model are most likely to be classified as Tindale 2 (48% of observations) followed by Bowdler 2 (23% of observations).

As for the misclassification patterns of all other classes, Birdsell 1 and 2 are often correctly classified, with 80% and 72% of observations classified

correctly. Southern Sunda is also well identified, with 67% of observations classified correctly. The Bowdler 1 and Bowdler 2 migration models are most likely to be classified as Bowdler 2, with 62% and 64% of observations classified as Bowdler 2 respectively. We also notice that all observations from the Northern Sunda, Tindale 1, or Tindale 2 migration models are most likely to be classified as Tindale 2, with 48%, 45%, and 46% of observations being classified as Tindale 2. These apparent groupings of Bowdler 1 and 2, and then Northern Sunda, Tindale 1, and Tindale 2 are similar to those noticed in the confusion matrices for other classifiers, except here they are all classified as belonging to one class.

(A)

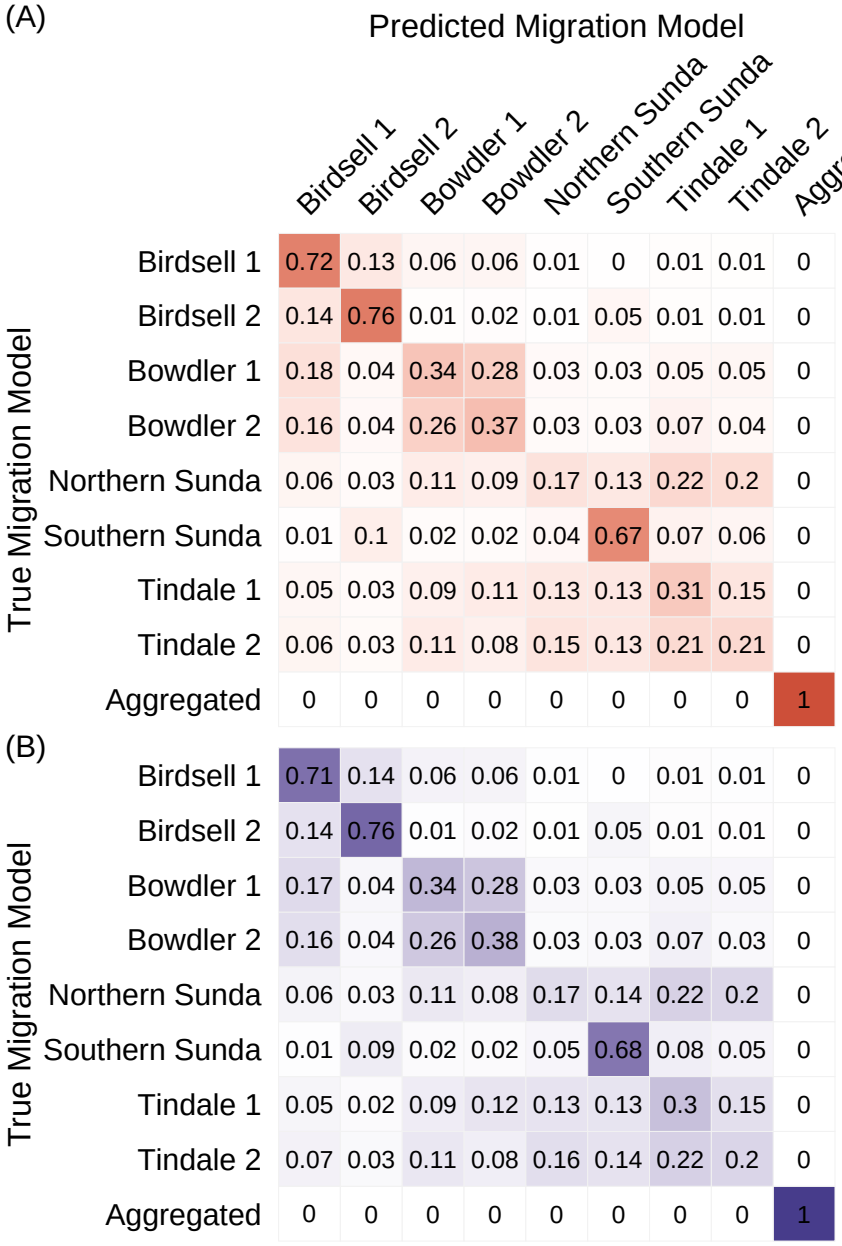
True Migration Model \ Predicted Migration Model	Birdsell 1	Birdsell 2	Bowdler 1	Bowdler 2	Northern Sunda	Southern Sunda	Tindale 1	Tindale 2	Aggregated
Birdsell 1	0.73	0.13	0.06	0.06	0.01	0	0.01	0.01	0
Birdsell 2	0.14	0.76	0.01	0.01	0.01	0.04	0.01	0	0
Bowdler 1	0.18	0.04	0.34	0.28	0.03	0.02	0.05	0.05	0
Bowdler 2	0.16	0.04	0.26	0.38	0.03	0.03	0.07	0.04	0
Northern Sunda	0.06	0.03	0.11	0.09	0.17	0.13	0.22	0.2	0
Southern Sunda	0.01	0.09	0.02	0.03	0.05	0.67	0.07	0.06	0
Tindale 1	0.05	0.03	0.09	0.12	0.13	0.13	0.31	0.15	0
Tindale 2	0.06	0.03	0.11	0.08	0.15	0.13	0.21	0.22	0
Aggregated	0	0	0	0	0	0	0	0	1

(B)

True Migration Model \ Predicted Migration Model	Birdsell 1	Birdsell 2	Bowdler 1	Bowdler 2	Northern Sunda	Southern Sunda	Tindale 1	Tindale 2	Aggregated
Birdsell 1	0.72	0.14	0.06	0.05	0.01	0	0.01	0.01	0
Birdsell 2	0.14	0.76	0.01	0.02	0.01	0.04	0.01	0.01	0
Bowdler 1	0.18	0.04	0.34	0.28	0.03	0.03	0.05	0.04	0
Bowdler 2	0.16	0.04	0.26	0.39	0.02	0.03	0.07	0.04	0
Northern Sunda	0.06	0.03	0.11	0.09	0.17	0.13	0.22	0.2	0
Southern Sunda	0.01	0.09	0.02	0.02	0.05	0.67	0.08	0.06	0
Tindale 1	0.05	0.02	0.09	0.12	0.13	0.12	0.3	0.16	0
Tindale 2	0.06	0.03	0.11	0.08	0.16	0.13	0.22	0.2	0
Aggregated	0	0	0	0	0	0	0	0	1

Table 6.6: A confusion matrix of the migration model predicted by MLR with forward selection and the true migration model. The proportions in matrix (A) calculated using the training data, while the proportions in matrix (B) were calculated using the test data. Rows correspond to the true migration model, while columns correspond to the predicted migration model. For example, the proportion of Birdsell 1 observations in the training data that were misclassified as Birdsell 2 is 0.13. Darker squares indicate that a higher proportion of observations fall into that category.

(A)



(B)

Table 6.7: A confusion matrix of the migration model predicted by MLR with LASSO and the true migration model. Matrix (A) was calculated from the training data, while matrix (B) was calculated from the test data. Rows correspond to the true migration model, while columns correspond to the predicted migration model. Darker squares indicate that a higher proportion of observations fall into that category.

(A)

True Migration Model \ Predicted Migration Model	Birdsell 1	Birdsell 2	Bowdler 1	Bowdler 2	Northern Sunda	Southern Sunda	Tindale 1	Tindale 2	Aggregated
Birdsell 1	0.74	0.12	0.05	0.05	0.01	0	0.01	0.01	0
Birdsell 2	0.15	0.78	0.01	0.01	0.01	0.04	0.01	0	0
Bowdler 1	0.19	0.03	0.38	0.27	0.04	0.03	0.03	0.03	0
Bowdler 2	0.17	0.04	0.24	0.42	0.03	0.02	0.06	0.03	0
Northern Sunda	0.06	0.03	0.11	0.1	0.21	0.1	0.21	0.18	0
Southern Sunda	0.02	0.09	0.03	0.03	0.06	0.66	0.06	0.06	0
Tindale 1	0.06	0.03	0.09	0.12	0.15	0.1	0.28	0.16	0
Tindale 2	0.06	0.03	0.11	0.09	0.17	0.1	0.21	0.23	0
Aggregated	0	0	0	0	0	0	0	0	1

(B)

True Migration Model \ Predicted Migration Model	Birdsell 1	Birdsell 2	Bowdler 1	Bowdler 2	Northern Sunda	Southern Sunda	Tindale 1	Tindale 2	Aggregated
Birdsell 1	0.73	0.12	0.05	0.07	0.01	0	0.01	0.01	0
Birdsell 2	0.16	0.74	0.01	0.02	0.01	0.05	0.01	0.01	0
Bowdler 1	0.19	0.04	0.34	0.3	0.04	0.02	0.04	0.04	0
Bowdler 2	0.17	0.03	0.28	0.38	0.03	0.02	0.05	0.03	0
Northern Sunda	0.07	0.03	0.11	0.1	0.18	0.11	0.2	0.2	0
Southern Sunda	0.01	0.1	0.03	0.03	0.06	0.64	0.07	0.06	0
Tindale 1	0.05	0.03	0.1	0.13	0.15	0.11	0.27	0.16	0
Tindale 2	0.07	0.03	0.12	0.1	0.18	0.11	0.2	0.2	0
Aggregated	0	0	0	0	0	0	0	0	1

Table 6.8: A confusion matrix for true and predicted migration models using a linear SVM trained on 15% of the data. All values are given in proportions. Matrix (A) is calculated from the training data, while matrix (B) is calculated from the test data.

(A)

True Migration Model \ Predicted Migration Model	Birdsell 1	Birdsell 2	Bowdler 1	Bowdler 2	Southern Sunda	Tindale 1	Tindale 2	Aggregated
Birdsell 1	0.82	0.07	0.03	0.05	0	0	0.02	0
Birdsell 2	0.21	0.71	0.01	0.02	0.04	0	0.02	0
Bowdler 1	0.21	0.02	0.04	0.63	0.02	0.03	0.04	0
Bowdler 2	0.2	0.02	0.04	0.65	0.03	0.03	0.03	0
Northern Sunda	0.07	0.02	0.02	0.24	0.12	0.06	0.47	0
Southern Sunda	0.02	0.09	0	0.07	0.67	0.02	0.14	0
Tindale 1	0.07	0.02	0.02	0.25	0.12	0.06	0.46	0
Tindale 2	0.07	0.02	0.02	0.24	0.12	0.06	0.47	0
Aggregated	0	0	0	0	0	0	0	1

(B)

True Migration Model	Birdsell 1	Birdsell 2	Bowdler 1	Bowdler 2	Southern Sunda	Tindale 1	Tindale 2	Aggregated
Birdsell 1	0.8	0.08	0.04	0.05	0	0	0.02	0
Birdsell 2	0.21	0.72	0.01	0.01	0.04	0	0.02	0
Bowdler 1	0.22	0.03	0.04	0.62	0.03	0.03	0.03	0
Bowdler 2	0.21	0.02	0.04	0.64	0.03	0.03	0.03	0
Northern Sunda	0.07	0.02	0.02	0.23	0.12	0.06	0.48	0
Southern Sunda	0.02	0.1	0	0.06	0.67	0.02	0.13	0
Tindale 1	0.07	0.02	0.02	0.25	0.13	0.06	0.45	0
Tindale 2	0.08	0.02	0.02	0.24	0.12	0.06	0.46	0
Aggregated	0	0	0	0	0	0	0	1

Table 6.9: A confusion matrix for true and predicted migration models using a neural network with Architecture 2. All values are given in proportions. Matrix (A) is calculated from the training data, while matrix (B) is calculated from the test data. Note that there is no column for Northern Sunda, as this migration model was never predicted by the classifier.

6.4 Using Classification Methods for Prediction

Now that we have trained all classifiers and compared their performance through accuracy and confusion matrices, we use the classifiers to predict the migration model where the simulated summary statistics most closely resemble the observed summary statistics. The outputs of MLR with forward selection, MLR with LASSO, a linear SVM trained on 15% of the data, and a neural network with Architecture 2 are given in Table 6.10.

According to MLR with forward selection, the most likely migration model was Southern Sunda with a probability of approximately one, although from the confusion matrices in Table 6.6 we note that summary statistics from models that were known to be Southern Sunda were only correctly identified 67% of the time.

To further investigate this seemingly confident prediction from MLR with forward selection, we determined the class probabilities for all observations in the test data. The resulting data was then filtered to include the observations that were classed as Southern Sunda, but were truly simulated under a different migration model. The mean and median probabilities of observations misclassified as Southern Sunda were 0.42 and 0.39 respectively, which is consistent with an uncertain result. We also note that the maximum probability of an observation coming from the Southern Sunda migration model, when it was simulated under another migration model, is 0.98. A class probability of 1 for Southern Sunda does not guarantee that the classification is correct.

MLR with LASSO produced less conclusive results, but the most likely migration model was again Southern Sunda, with a probability of 0.262. The second most likely migration model was Southern Sunda with a probability of 0.178. Based on the class probabilities in Table 6.10, it is very unlikely that the observed summary statistics are consistent with the scenarios described in either of the Birdsell migration models. The probability of these classes is low, and in the confusion matrices for training and test data (Table 6.7), we notice that if the predicted class is Southern Sunda, the actual class is almost never Birdsell, Bowdler, or the aggregated model. If we were to base our results solely on the results from this one classifier, we could consider the similarities between Northern Sunda, Southern Sunda, and the Tindale migration models to tentatively suggest that a single entry point to Australia via New Guinea, and then subsequent coastal migration, is the most likely

Migration Model	MLR with Forward Selection	MLR with LASSO	Linear SVM	Neural Network
Birdsell 1	1.07×10^{-22}	2.04×10^{-3}	**1**$^{\#}$	**0.489**
Birdsell 2	8.34×10^{-14}	5.31×10^{-3}	0	0.082
Bowdler 1	2.12×10^{-19}	0.106	0	0.151
Bowdler 2	1.17×10^{-35}	0.092	0	0.152
Northern Sunda	3.84×10^{-10}	0.178	0	0.040
Southern Sunda	$\approx$ **1.00**	**0.262**	0	0.008
Tindale 1	8.74×10^{-17}	0.162	0	0.039
Tindale 2	1.05×10^{-10}	0.195	0	0.039
Aggregated	4.45×10^{-26}	5.74×10^{-6}	0	$< 2.22 \times 10^{-16}$

Table 6.10: The output of each classifier, given as the predicted probability of the observed summary statistics coming from each migration model.
$^{\#}$ SVMs provide only a hard classification with no class probabilities, and so we represent the classification of the observed summary statistics as Birdsell 1 by a class probability of 1.

scenario based on the observed data.

We do note that both MLR with forward selection and MLR with LASSO both technically performed better than random guessing, which would give $1/9 \approx 11.1\%$ accuracy for nine classes. The accuracies of the classifiers may still not be high enough to confidently predict a migration model: the forward selection regression model had a test accuracy of 50.6%, while the LASSO regression model had a slightly lower test accuracy of 50.4%.

Using a SVM with a linear kernel trained on 15% of the data to predict the migration model that best describes the observed summary statistics results in a prediction of Birdsell 1. No probability or other measure of confidence is given, as SVMs only provide hard classifications.

Recall from the theoretical discussion of SVMs in Section 4.4.2 that SVMs do not inherently provide a multiclass classification, and instead consider all possible pairs of binary classification problems. In the case of a tie in the class most frequently selected by the binary classifiers, the LibSVM implementation that underpins SVMs in R selects the first class alphabetically. Since the alphabetically first migration model was selected, we should investigate the prediction of each binary classifier to ensure that a tie did not occur. These results are displayed in Table 6.11. It is clear from Table 6.11 that a

Migration Model	Number of times selected
Birdsell 1	8
Birdsell 2	6
Bowdler 1	3
Bowdler 2	3
Northern Sunda	2
Southern Sunda	4
Tindale 1	6
Tindale 2	4
Aggregated	0
(Total)	36

Table 6.11: The number of times that each migration model was selected by a binary SVM classifier. There were 36 binary classifiers in total, since there were nine different classes. Each class appeared in 8 binary classifiers.

tie did not occur, so Birdsell 1 was the migration model that was selected most frequently in all possible binary classifications.

The prediction of Birdsell 1 as the migration model that best explains the observed summary statistics is not consistent with the predictions obtained from MLR. From the confusion matrices in Figure 6.8, we notice that summary statistics simulated under Birdsell 1 are rarely classified as another model, and only 1% of the observations are misclassified as Southern Sunda. Both types of MLR predicted Southern Sunda as the migration model, with some support for other similar models such as Northern Sunda, Tindale 1, and Tindale 2. From the confusion matrices in Tables 6.6 and 6.7, we also notice that observations from any of these migration models are rarely misclassified as Birdsell 1. Hence both MLR and SVMs are unlikely to have provided a correct classification.

In Table 6.10 we also present the output of using a neural network for prediction, by giving the values of the output layer that correspond to each class. The model with the largest output value, and hence the classification of the observed summary statistics, is Birdsell 1. This is consistent with the prediction from the linear SVM, but inconsistent with the results from MLR with both types of variable selection.

6.5 Validation Analyses

Despite all classification methods producing a model with similar test accuracy we note that different migration models were selected by different classifiers, as summarised in Table 6.10. Across all classifiers that provide probabilities or output values, we notice that the probabilities are either low or unreliable. Recall that in Section 6.3.1 we found that when using MLR with forward selection, misclassified observations in the test data had class probabilities up to 0.98; this is what we mean by 'unreliable' in this context.

To further investigate the stability of these predictions, we conducted validation runs using a smaller percentage of the data due to computation time. The test accuracy and confusion matrices were similar across all classifiers, and not significantly worse than the accuracies and confusion matrices already presented. The results of all validation runs are given in Table 6.12. Further details of the validation process can be found in Appendix C.2.4.

From Table 6.12, we can see that MLR with forwards selection consistently predicts migration models with high probability, but these are not always the same model. The fourth validation set for MLR had more balanced probabilities, resembling the class probabilities from MLR with LASSO. MLR with LASSO consistently predicted Southern Sunda, although the class probabilities across all classes are relatively similar. Linear SVMs do not produce consistent results when using 10% of the data as training data. The second validation set resulted in a 'strong' prediction for Birdsell 1 in terms of the votes of binary classifiers; the fourth validation set returned Birdsell 1 due to a tie between Birdsell 1, Birdsell 2, Southern Sunda and Northern Sunda. Finally, neural networks predict both Southern Sunda and the aggregated migration model with high output values in different validations.

Method	Validation Run	Training Accuracy (%)	Val. Accuracy (%)	Test Accuracy (%)	Migration Model	Probability
MLR (forward selection)	1	51.4	-	50.0	Southern Sunda	1
	2	51.4	-	49.8	Birdsell 1	1
	3	51.1	-	49.9	Southern Sunda	1
	4	51.8	-	49.7	Southern Sunda	0.397
MLR (LASSO)	1	51.0	-	49.5	Southern Sunda	0.349
	2	50.6	-	49.3	Southern Sunda	0.323
	3	50.7	-	49.5	Southern Sunda	0.232
	4	51.5	-	49.5	Southern Sunda	0.384
SVM (linear)	1	52.3	49.4	49.7	Tindale 1	-
	2	52.7	49.6	49.6	Birdsell 1	-
	3	52.3	49.2	49.6	Tindale 1	-
	4	53.0	49.9	49.6	Birdsell 1^t	-
Neural Network (Arc. 2)	1	48.8	48.9	48.6	Aggregated	1
	2	48.8	49.9	49.7	Southern Sunda	0.999
	3	48.7	48.7	48.5	Southern Sunda	1
	4	49.7	49.6	49.3	Aggregated	1

Table 6.12: Results from validation runs across all classifiers. The t superscript indicates that the SVM classification was a tie between different migration models.

6.6 Discussion

Multinomial logistic regression, support vector machines, and neural networks did not consistently classify the observed summary statistics as a particular migration model. The best models found using each of these methods had test accuracies of 50.6%, 50.1%, and 49.3% respectively. Recall that random guessing with nine classes would result in an accuracy of 11.11%, and so it is clear that all classifiers perform better than random guessing. From inspecting confusion matrices for each classifier, it is also clear that the performance of classifiers is not uniform across classes. Some migration models are more distinguishable than others.

We hypothesise that common misclassifications are based on the locations that post-settlement migration is allowed to occur between. The different types of post-settlement migration are given in Figure 6.8, and the groups of different types of post-settlement migration are described in Table 6.13. With the exception of the aggregated model, these groups correspond to the patterns observed in the confusion matrices for all classifiers.

The aggregated model is likely to be perfectly classified as all populations in southern Australia are aggregated into one population, which results in the 'individual' populations in southern Australia having identical summary statistics. This is clearly not the case for all other migration models: even if the summary statistics for the two southern populations are similar, they would not be identical. Therefore, we do not expect Southern Sunda to be misclassified as the aggregated model or vice versa, even though they have identical post-settlement migration patterns. The link between post-settlement migration patterns and misclassification will be explored further in Chapter 7.

Migration Types	Migration Models
(None)	Birdsell 1
m_{23}	Birdsell 2
$m_{\{456\}}$	Bowdler 1, Bowdler 2
m_{34}, $m_{\{456\}}$	Northern Sunda, Tindale 1, Tindale 2
m_{23}, m_{34}, $m_{\{456\}}$	Southern Sunda, Aggregated*

Table 6.13: Locations where post-settlement migration occurs in each migration model. Recall that post-settlement migration was allowed to occur along three different migration corridors: between southern Wallacea and New Guinea (denoted m_{23}), New Guinea and northern Australia (denoted m_{34}), and within Australia (denoted $m_{\{456\}}$). These correspond to the migration matrix elements given in Figure 6.8.

$$M = \begin{array}{c|cccccc} & 1 & 2 & 3 & 4 & 5 & 6 \\ \hline 1 & m_{11} & m_{12} & m_{13} & m_{14} & m_{15} & m_{16} \\ 2 & m_{21} & m_{22} & m_{23} & m_{24} & m_{25} & m_{26} \\ 3 & m_{31} & m_{32} & m_{33} & m_{34} & m_{35} & m_{36} \\ 4 & m_{41} & m_{42} & m_{43} & m_{44} & m_{45} & m_{46} \\ 5 & m_{51} & m_{52} & m_{53} & m_{54} & m_{55} & m_{56} \\ 6 & m_{61} & m_{62} & m_{63} & m_{64} & m_{65} & m_{66} \end{array}$$

Figure 6.8: Top: A map indicating the population location corresponding to each row/column of the migration matrix.
Bottom: The migration matrix, with each level of migration having a different colour. Darker colours represent a higher probability of migration on average, and no colour indicates no migration between populations. Diagonal entries are shaded, because migration from a population to itself is meaningless.

The lack of accuracy exhibited by all classifiers, the clear misclassification patterns observed in all confusion matrices in Section 6.3.4, and the instability of the results highlighted in Section 6.5 mean that we could not reliably identify a single migration model that described the observed summary statistics. From the dimension reductions and the classification results we can conclude that the observed summary statistics are not consistent with the demographic history presented in the aggregated model, although we cannot conclusively state which of the other migration models is most likely.

This inconclusive result is consistent with the results of the previous phylogenetic analysis in Section 3.4. All trees had poor branch support approximately 50 ka when different haplogroup splits occurred, and the migration events that we base our migration models on occur in quick succession approximately 50 ka. The results are also consistent with the dimension reductions from Section 6.2. The representation of the summary statistics in two-dimensions were similar for all migration models except the aggregated model, and so it is not surprising that there was not a strongly-supported result.

Finally, we note that if we had used only one classification method, we may have erroneously been much more confident in our results. Considering each of the classifiers individually, there was either a clear migration model selected or a family of migration models with similar characteristics. There is no clear rationale for one classifier to have a superior result to all other classifiers, and so we cannot rule out predictions from this perspective. We should treat all neural network results with caution because of the occasional prediction of the aggregated model in the validation runs (see Table 6.12). Due to the black-box nature of this classification method, no reason could be found for these unlikely classifications. Even when disregarding the neural network results, we still have conflicting predictions from the linear SVM and MLR.

In conclusion, we could not select one migration model that best explains the observed summary statistics, although we did find that the aggregated model is extremely unlikely to adequately explain the observed data. Our inconsistent results are also reasonable in light of the phylogenetic analyses presented in Chapter 3. Different conclusions could have been made by only considering the results of one classification method, but when considering the results of all classification methods, we find no strong support for any single migration model.

In the next chapter we consider extensions to this simulation study to further investigate the effects of rapid migration events, as well as the consequences

of other modelling choices.

Chapter 7

Extended Analysis and Discussion

In this chapter we explore multiple extensions to our initial simulation study, where we will thoroughly examine the effect of the assumptions made in the model design process and assess whether they contributed to our inconclusive results in Chapter 6.

First, we apply a homogeneous post-settlement migration pattern, which is identical for all migration models. Recall that post-settlement migration is the small amount of migration that occurs after the initial migration events have occurred. This extension will further our understanding of how post-settlement migration rates affect the summary statistics.

Next, we investigate how the time between island hops through the islands of southeast Asia affect the summary statistics of different migration models. The initial migration models assumed a time of 250 years between island hops; we remind the reader that a human female generation time of 25 years [24] is used to convert years to generations. This short inter-event time of 250 years reflects potential rapid migration [63], which could result in a lack of signal in the data due to a lack of substitution events.

As mentioned in Section 5.1.1 when defining the times between migration events, reasonable estimates for the times between island hops are from 250 to 2,500 years. These values correspond to the migration between mainland southeast Asia (Sunda) and Sahul taking a total of 500 - 5,000 years [63, 94]. We present results that assume 2,500 years between island hops, as well as exploring the effect of using unreasonable inter-event times of 25,000

or 250,000 years between island hops. These longer times between island hops are clearly unrealistic, but they should illustrate how changing the time between island hops affects the summary statistics, and what length of time may be required to significantly increase the signal in the data.

Finally, we implement further subsetting of our data based on the haplogroups of the mtDNA samples. We will investigate whether mixed mtDNA haplogroups within the southern Australian population significantly affect our results, and if further subsetting results in more distinguishable simulated summary statistics.

7.1 Homogeneous Post-Settlement Migration

Recall that in our initial simulation study, we assumed that post-settlement migration only occurred between population locations connected by the initial migration route, and also that post-settlement migration occurred from the creation of a population to the present day.

To investigate the extent to which the distinct post-settlement migration patterns affected the signal in the summary statistics, we now apply homogeneous post-settlement migration patterns across all migration routes. We will also consider the effects of the Last Glacial Maximum on post-settlement migration, which was not accounted for in the initial simulations.

The Last Glacial Maximum began approximately 30 ka and ended approximately 8 ka [45, 16]. This resulted in sea levels rising, which would have impeded post-settlement migration between the southeast Asian islands, New Guinea, and mainland Australia. We therefore allow ongoing migration between southern Wallacea and New Guinea, as well as New Guinea and northeastern Australia from the settling of all population locations until 8 ka.

Migration routes with homogeneous post-settlement migration

Figure 7.1 illustrates the patterns of post-settlement migration applied to each of the migration routes. The solid arrows indicate large-scale migration events that result in the occupation of a new location, while the dashed arrows indicate ongoing post-settlement migration.

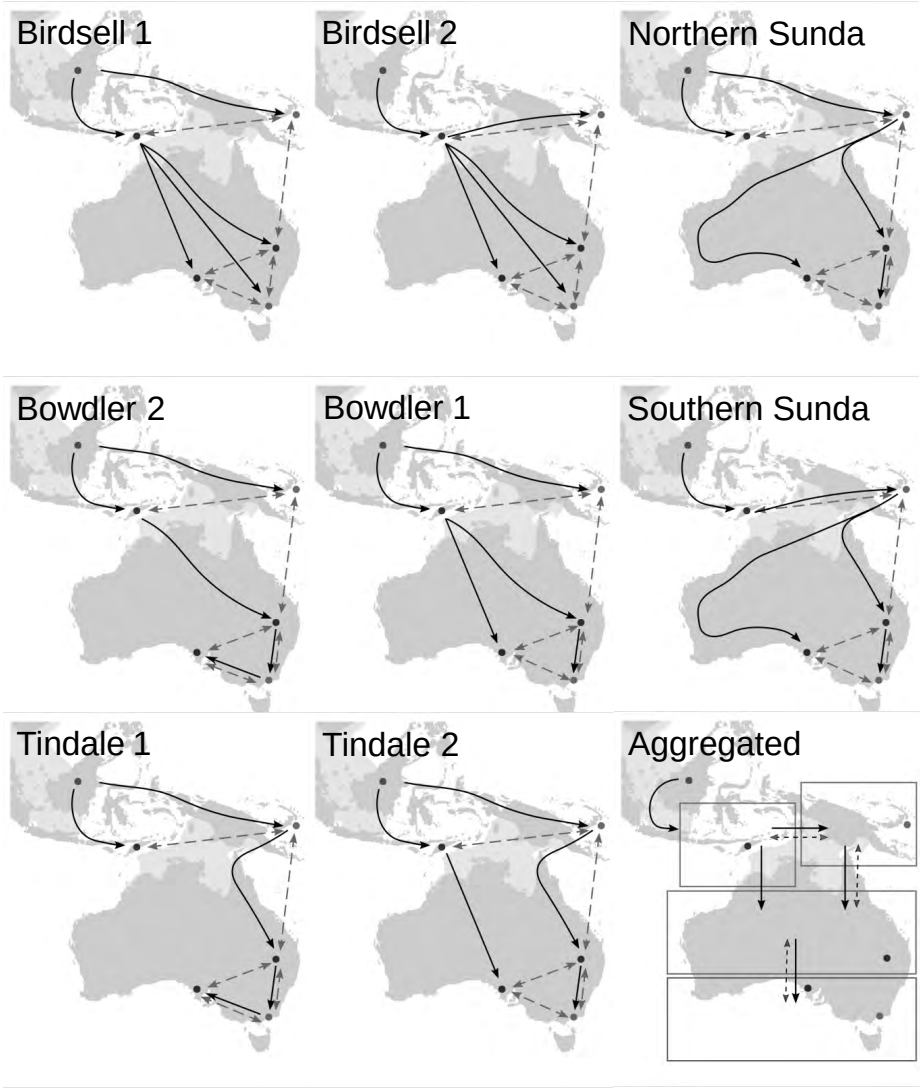


Figure 7.1: All migration routes (solid arrows) and ongoing patterns of post-settlement migration (dashed arrows). The name of the corresponding migration model is given in the top left corner of each diagram.

Duration of homogeneous post-settlement migration

Since post-settlement migration can now occur between populations that are not connected by an initial large-scale migration, we require that post-settlement migration begins after all population locations have been inhabited. The time that post-settlement migration commences is variable, be-

cause we allow the timing of the peopling of Sahul to uniformly occur any time between 50 ka and 65 ka. The duration of post-settlement migration through the islands of southeast Asia is restricted by the Last Glacial Maximum which caused sea levels to rise, making the maritime travel required along this migration corridor more difficult. Post-settlement migration within Australia occurs from the time that all population locations were inhabited to the present day.

Comparing homogeneous post-settlement migration to the original migration models

Before comparing the initial migration models to those assuming homogeneous post-settlement migration, we perform PCA on the summary statistics for our migration models with homogeneous post-settlement migration (see Figure 7.2). We see that the principal components for all models except the aggregated model overlap significantly. We also note that the observed summary statistics do not lie within the range of the principal components for any migration models. UMAP was also used for dimension reduction, and we observe a similar overlap (see Figure C.26, Appendix C.3.1).

Figure 7.3 shows the first and second principal components of the original simulated summary statistics, and those simulated assuming homogeneous post-settlement migration. Convex hulls are added to visualise the boundary and spread of each type of summary statistics. Note that all points lie either on the boundary of or inside a convex hull, and the polygon created by a convex hull is therefore convex.

The convex hulls are considerably different, which is due to the significantly different summary statistics for the aggregated model under homogeneous and non-homogeneous assumptions of post-settlement migration. We repeated the PCA after excluding the aggregated model in both cases, and obtained Figure 7.4. Here, we can see that the summary statistics simulated under the original post-settlement migration models and the homogeneous post-settlement migration models overlap considerably.

Next, consider Table 7.1. The mean of each summary statistic was calculated for each migration model, for both the original and homogeneous post-settlement migration patterns. The resulting point for each migration model is referred to as the centroid of the summary statistics for that migration model. The absolute Manhattan distances between all pairwise combinations of means were then calculated for each type of post-settlement migration pat-

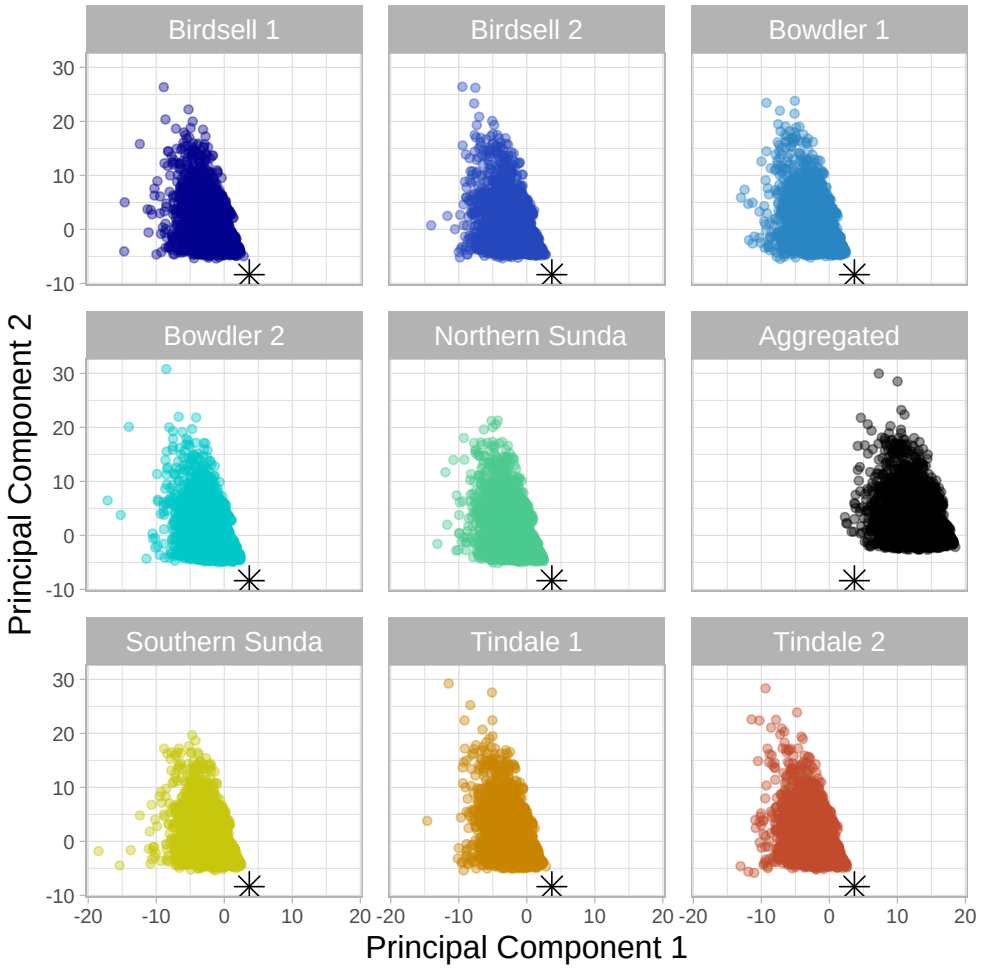


Figure 7.2: The first two principal components of the summary statistics from migration models assuming homogeneous post-settlement migration patterns. The plot was facetted by migration model due to the significant overlap of the principal components. The observed summary statistics are given as the black star in all panels.

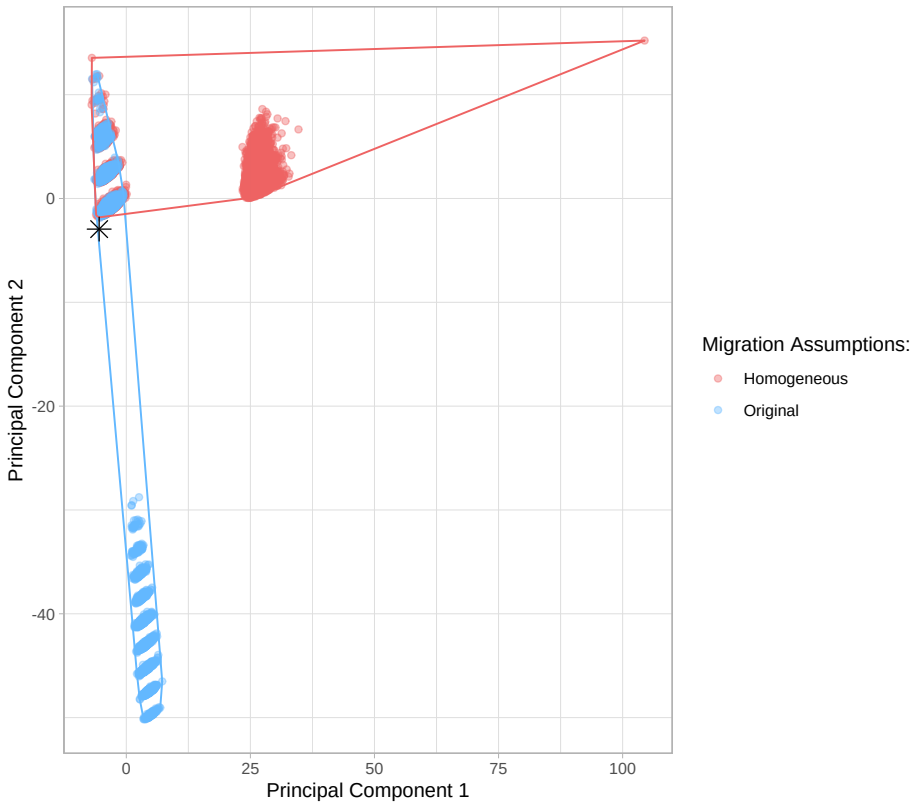


Figure 7.3: Simulated summary statistics from the original migration models (blue) and the migration models with homogeneous post-settlement migration (red). Models are not given distinct colours, as it has already been shown that the summary statistics for different models overlap significantly (see Figure 7.2). Convex hulls are shown as a solid line in the corresponding colour for each type of summary statistics.

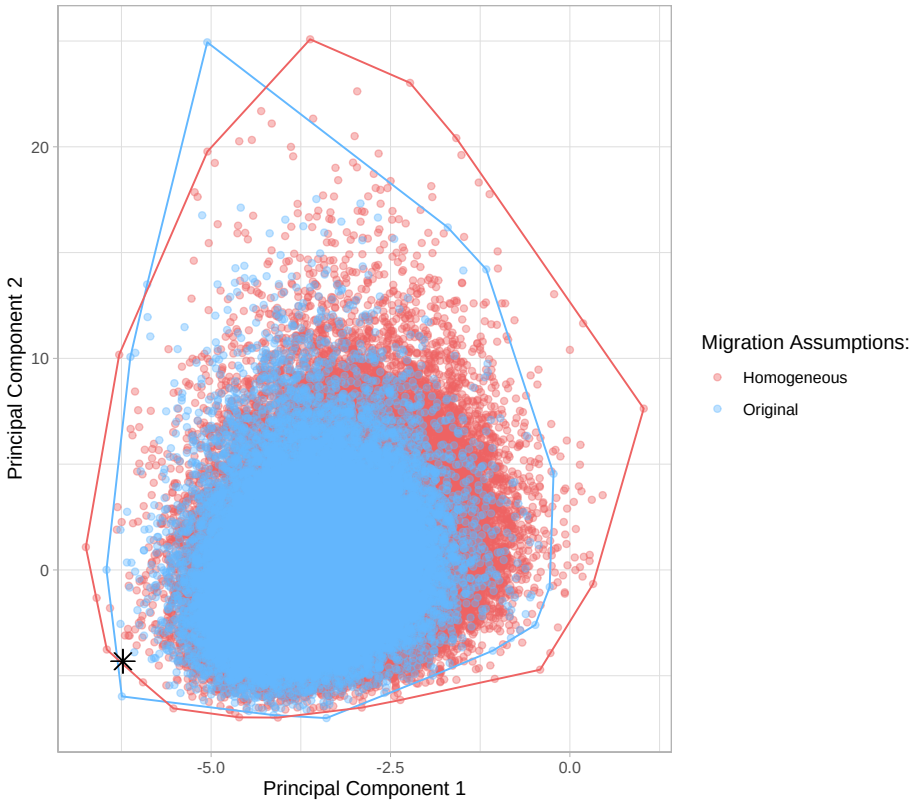


Figure 7.4: Simulated summary statistics from the original migration models (blue) and the migration models with homogeneous post-settlement migration (red), excluding the aggregated model. Migration models are not given distinct colours, as it has already been shown that the summary statistics for different models overlap significantly (see Figure 7.2). Convex hulls are shown as a solid line in the corresponding colour for each type of summary statistics.

(Between)	Birdsell 2	Bowdler 1	Bowdler 2	Aggregated
Birdsell 1	10356.7 **303.8**	51.1 **5.0**	176.8 **10.5**	130.7 **13.9**
Birdsell 2	- -	10386.1 **303.0**	10460.3 **304.4**	10425.9 **301.6**
Bowdler 1	- -	- -	131.4 **9.9**	122.6 **13.5**
Bowdler 2	- -	- -	- -	55.0 **9.6**
Aggregated	- -	- -	- -	- -
Northern Sunda	- -	- -	- -	59.4 **11.2**
Southern Sunda	- -	- -	- -	- -
Tindale 1	- -	- -	- -	- -

(Between)	Northern Sunda	Southern Sunda	Tindale 1	Tindale 2
Birdsell 1	79.9 **5.7**	77.6 **14.1**	132.2 **7.6**	132.0 **8.1**
Birdsell 2	10393.7 **303.5**	10387.7 **300.3**	10431.6 **301.1**	10429.8 **300.1**
Bowdler 1	127.8 **6.2**	126.0 **14.5**	126.1 **7.7**	124.8 **9.0**
Bowdler 2	99.6 **7.0**	107.8 **11.1**	48.7 **7.4**	49.7 **8.5**
Aggregated	- -	56.9 **5.2**	10.7 **9.7**	**8.8** 10.6
Northern Sunda	- -	**9.4** 12.1	55.6 **6.2**	55.4 **5.9**
Southern Sunda	- -	- -	64.3 **9.0**	63.2 **9.8**
Tindale 1	- -	- -	- -	4.1 **3.9**

Table 7.1: L1 (Manhattan) distance between the centroids of the summary statistics for each pair of migration models. The distances for the original migration models are the topmost element of each cell, while the distances for the migration models assuming homogeneous post-settlement migration are the second element given in each cell.

tern (original or homogeneous). Manhattan distances can be thought of as the total 'element-wise' distance.

Since Euclidean distances are commonly used even for high-dimensional distances, we include the Euclidean distances between each centroid in Appendix C.3.2. To determine if the conclusions are affected by significant skew in the summary statistics, we also take the median of each summary statistic for each migration model. The resulting point for each migration model is referred to as the geometric median. The distances between geometric medians are also included in Appendix C.3.2.

For nearly all comparisons, the means of the summary statistics resulting from homogeneous post-settlement migration are much closer together than the means of the original summary statistics. Similar patterns were noticed for comparisons involving Euclidean distances and geometric medians.

To further illustrate the effects of homogeneous post-settlement migration, we use a classification method to predict the migration model given a set of summary statistics. We will only use MLR with LASSO to obtain a confusion matrix; in Chapter 6 we noticed that even if classifiers produced different results, the confusion matrices were nearly identical across different methods, and accuracies were very similar. As in Chapter 6, we use 70% of the simulations as training data and the remaining 30% as test data. MLR with LASSO is again implemented through the glmnet package in R, and 5-fold cross-validation is used to determine the penalization parameter λ.

The test accuracy of the classifier was 23.3% and the training accuracy was 24.7%. The confusion matrix is given in Table 7.2. We see that only the aggregated model is clearly distinguished from the other models. This is consistent with the PCA results, and is also a property of the confusion matrices calculated in Section 6.3.4.

There are no discernible groupings for the remaining migration models. We can see that summary statistics are most likely to be classified as the Bowdler 2 or Tindale 1 migration models regardless of the true migration model. Equivalently, if an observation is classified as any model other than the aggregated model, it is likely to have come from any other migration model with near-equal probability (again, excluding the aggregated model).

The closer centers of summary statistics under homogeneous post-settlement migration, the smaller convex hull seen in Figure 7.3, and the lack of structure in the confusion matrices in Table 7.2 all suggest that assuming a homogeneous pattern of post-settlement migration across all migration models results in summary statistics that are less distinct.

(A)

True Migration Model \ Predicted Migration Model	Birdsell 1	Birdsell 2	Bowdler 1	Bowdler 2	Northern Sunda	Southern Sunda	Tindale 1	Tindale 2	Aggregated
Birdsell 1	0.18	0.1	0.04	0.21	0.11	0.07	0.23	0.06	0
Birdsell 2	0.15	0.12	0.04	0.22	0.1	0.06	0.24	0.06	0
Bowdler 1	0.16	0.1	0.05	0.23	0.1	0.07	0.23	0.06	0
Bowdler 2	0.13	0.08	0.03	0.28	0.08	0.05	0.3	0.04	0
Northern Sunda	0.15	0.11	0.04	0.23	0.11	0.07	0.24	0.06	0
Southern Sunda	0.15	0.1	0.04	0.22	0.09	0.08	0.25	0.07	0
Tindale 1	0.12	0.08	0.04	0.24	0.08	0.05	0.34	0.05	0
Tindale 2	0.15	0.1	0.04	0.21	0.11	0.07	0.24	0.08	0
Aggregated	0	0	0	0	0	0	0	0	1

(B)

True Migration Model \ Predicted Migration Model	Birdsell 1	Birdsell 2	Bowdler 1	Bowdler 2	Northern Sunda	Southern Sunda	Tindale 1	Tindale 2	Aggregated
Birdsell 1	0.16	0.11	0.05	0.25	0.1	0.07	0.21	0.05	0
Birdsell 2	0.17	0.11	0.04	0.21	0.11	0.06	0.23	0.07	0
Bowdler 1	0.16	0.09	0.03	0.24	0.1	0.07	0.24	0.07	0
Bowdler 2	0.12	0.08	0.03	0.25	0.1	0.05	0.31	0.05	0
Northern Sunda	0.14	0.11	0.03	0.22	0.1	0.07	0.27	0.06	0
Southern Sunda	0.15	0.1	0.04	0.21	0.11	0.07	0.25	0.06	0
Tindale 1	0.12	0.09	0.03	0.26	0.09	0.06	0.32	0.04	0
Tindale 2	0.16	0.09	0.04	0.21	0.12	0.07	0.25	0.06	0
Aggregated	0	0	0	0	0	0	0	0	1

Table 7.2: Confusion matrices for the true and predicted migration models. Matrix (A) is based on the training data, while matrix (B) is based on the test data. The proportion of observations that fall into each category is given, with rows summing to one. Darker squares indicate greater proportions.

7.2 Mutations before Australia

In Sections 6.2 and 7.1 we have highlighted that the summary statistics for different migration models are quite similar, and the differences that are seen might be at least partially explained by post-settlement migration patterns. A natural topic to explore from here is the differences between sequences that should be expected due to the geographical migration routes. The expansion from southeast Asia through to Australia was quite rapid, and so the number of mutations that occurred in the migration events through the southeast Asian islands will be small compared to the number of mutations that occurred in the subsequent 50,000 years within Australia.

The signal present in the mtDNA sequence data that we are attempting to identify through the classification process comes from mutations that occur at different stages along the migration path. Certain mutations will be present in some populations but not others, depending on the migration path taken. Since no signal has been clearly identified, there are possibly no or very few substitutions currently occurring in the island hops between southeast Asia and Australia. We investigate this statement by looking at the expected number of point mutations and the probability of observing at least one point mutation.

7.2.1 Expected number of point mutations

When considering a single lineage and a substitution rate of $\mu = 6.0629 \times 10^{-3}$ substitutions per mtDNA coding region per generation [28], there is an expected time between substitution events of approximately 4,000 years (160 generations). Since there are multiple simulated DNA sequences, and the substitution process is stochastic, the expected time could be less than this, but it is still unlikely that many mutations will occur in ten generations.

Let $N \in Z^+ \cup \{0\}$ be a discrete random variable that is the number of substitutions occurring in one lineage. Since substitutions are modelled according to a Poisson process with rate μ, the number of substitutions occurring in the interval $(0, t]$ has a Poisson distribution with rate μt [7]. Therefore

$$\begin{aligned} P(N \geq 1) &= 1 - P(N = 0) \\ &= 1 - \frac{(\mu t)^0 e^{-\mu t}}{0!} \\ &= 1 - e^{\mu t}, \end{aligned}$$

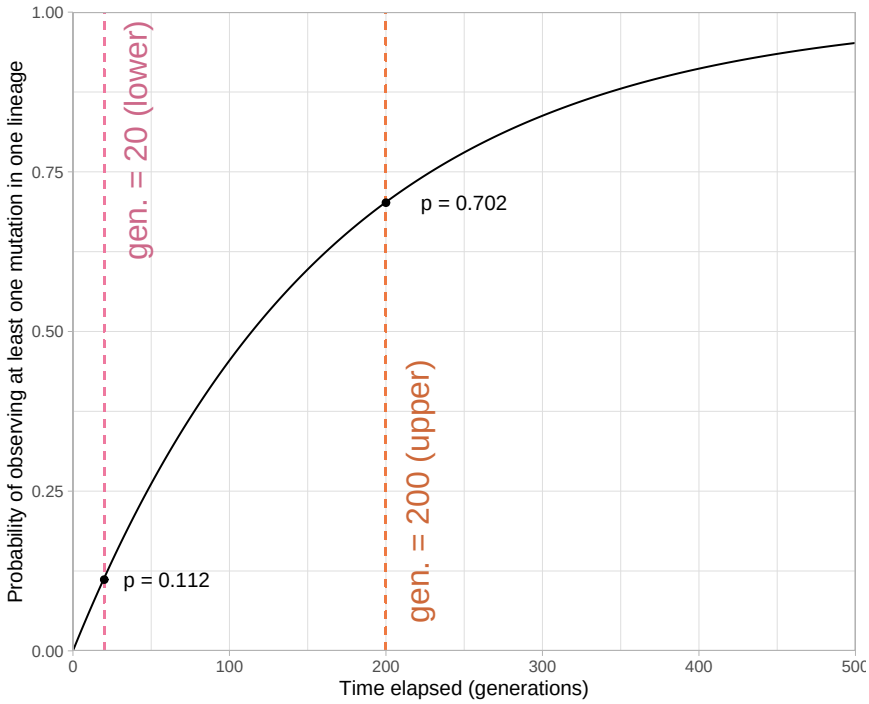


Figure 7.5: Probability of observing at least one substitution in one lineage.

where t is the amount of time elapsed in generations and μ is the substitution rate per sequence per generation.

Figure 7.5 displays the probability of observing at least one point mutation in one lineage, after allowing zero to 500 generations of point mutations to occur.

We define $\mathbb{E}[N|t = g]$ as the probability of observing N substitutions in a single lineage after g generations. Figure 7.6 displays the expected number of substitutions in one lineage for g between zero and 500 generations. Recall that substitutions occur according to a Poisson process with the rate defined by the substitution rate μ, and therefore the expected number of substitutions in t generations will be μt.

We can see that if the shortest reasonable time between island hops is assumed, with an inter-event time of 10 generations, less than one point mutation per lineage is expected to occur between each island hop. In the same timeframe, the probability of observing one or more substitutions is low (probability ≈ 0.1). Assuming the maximum reasonable time between island hops, *i.e.* an inter-event time of 100 generations, results in the expected number of substitutions per lineage being slightly above one. On

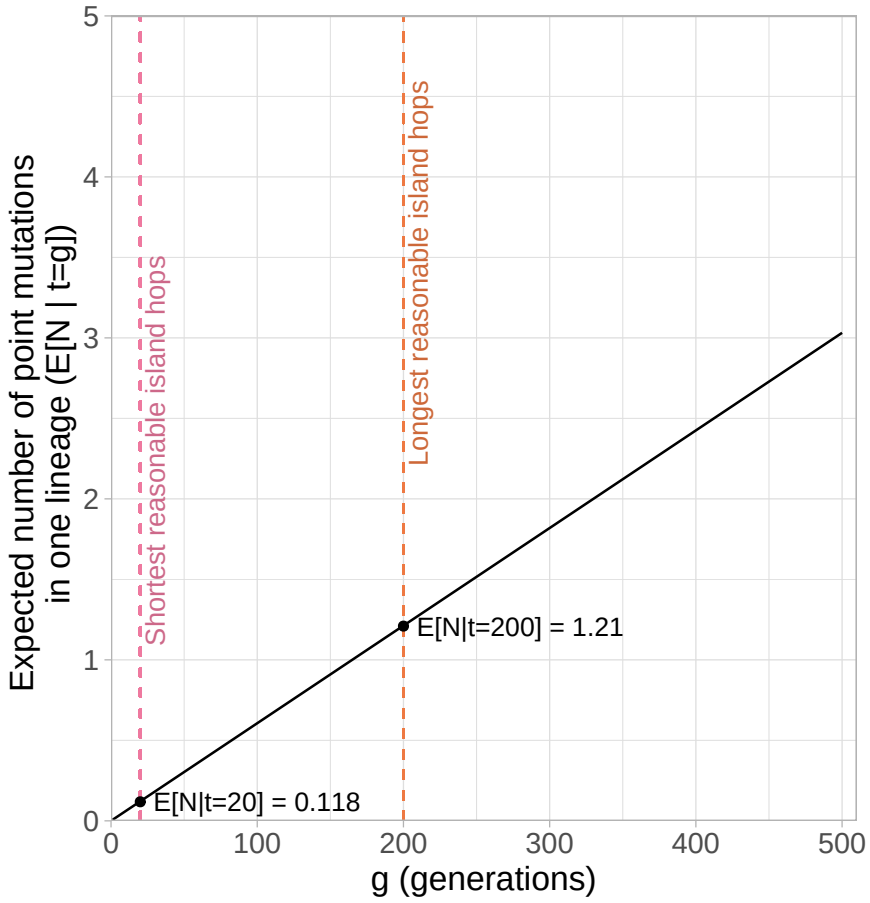


Figure 7.6: The expected number of substitutions in one lineage after g generations, $\mathbb{E}[N|t = g]$. The expected number of substitutions only passes 1 after 165 generations.

average, a greater number of substitutions would yield greater difference in the summary statistics of different migration models, making them more distinguishable from each other.

We also emphasise that this is an *expected value*: it is impossible for exactly 0.118 mutations to occur after 20 generations because the number of substitutions is a discrete random variable that cannot take non-integer values. The important conclusion from Figure 7.6 is that in general, we expect less than one substitution to occur in one lineage for the entire duration of the migration through island southeast Asia, which is unlikely to create any traceable signal of these migration events.

In conclusion, Figures 7.5 and 7.6 demonstrate that there are likely to be a very small number of substitutions accumulated in the migration through the southeast Asian islands. While one may hope the presence of any number of substitutions could result in clear selection of a migration model, the 12.1 substitutions that are expected to occur in the subsequent 50,000 years (for one lineage) after reaching Sahul makes it difficult to clearly identify any trace of the early migration events in the mtDNA. Post-settlement migration occurring after the peopling of Sahul would further obscure any signal present.

7.3 Increasing the Time Between Migration Events

If the low power to distinguish between migration models corresponds to the short times between migration events early in the migration, then one would expect the summary statistics to become more distinct as the time between migration events increases. In the previous section, we noted that the probability of observing at least one mutation in 2,500 years (100 generations) was higher than the probability of observing at least one mutation in 25 years (10 generations). Here, we explore how allowing 2,500 years, 25,000 years, and 250,000 years between migration events through the islands of southeast Asia affects the summary statistics of different migration models. Increasing the time between migration events increases the number of substitutions that are expected to occur in this region, which should make summary statistics from different migration models more distinguishable.

This lengthening of the wait times between migration events is only implemented for the original models, and not the models with homogeneous

Time Between SE. Asian Migration Events (years)	Training Accuracy	Test Accuracy
250	50.5%	50.4%
2,500	54.5%	54.5%
25,000	71.8%	71.8%
250,000	75.0%	74.7%

Table 7.3: Training and test accuracy for MLR with LASSO predicting the migration model based on summary statistics. The results in Chapter 6 assume 250 years between migration events in southeast Asia; these accuracies were taken from Section 6.3.1.

post-settlement migration.

7.3.1 Comparing summary statistics from different inter-event times

We produced simulations with inter-event times of 2,500, 25,000, and 250,000 years for migration events through the southeast Asian islands. Specifically, this involved lengthening the times taken before lineages from population regions in Borneo, Southern Wallacea, and New Guinea could coalesce.

In Section 6.6 all classifiers had similar test accuracies that were between 45.1% and 50.6%. Since performance was comparable for all classifiers, we will use the classification method with the lowest computation time in the interest of computational efficiency.

Table 7.3 gives the test and training accuracies of MLR with LASSO calculated on the summary statistics simulated under each of the times between migration events. We can see that classifier performance, as measured by training and test accuracy, increases as the time between migration events through southeast Asia increases.

We also consider the confusion matrices for the data for each time between migration events. The confusion matrices for classification of data assuming 2,500, 25,000, and 250,000 years between migration events in southeast Asia are all given in Table 7.4. Based on these confusion matrices, it is clear that migration models become statistically more distinct as the time between migration events increases. This separation only increases to a certain extent though, because even when there are 250,000 years between migration events,

Bowdler 1 and Bowdler 2 are still commonly misclassified as each other, as are Northern Sunda and Tindale 1. The two characteristics shared by these pairs is that they have the same path through the southeast Asian islands (the migration models of both pairs contain Birdsell's Northern Route through the southeast Asian islands), and they also share the same entry point to Australia. This demonstrates that, given enough signal in the data in the form of informative substitutions, migration models can be statistically distinguished from each other.

We also expect to see signs of this increase in predictive accuracy visually. We performed linear and nonlinear dimension reduction techniques on the summary statistics, in the form of PCA and UMAP respectively.

The PCA results are given in Figure 7.7. All PCA plots appear very similar, and there is no indication that the summary statistics from different migration models become more distinct as the time between migration events through southeast Asia increases.We have not included the projected observed summary statistics in Figure 7.7 because these longer durations of island hops are clearly unreasonable, and

Recall that UMAP can identify non-linear trends in the summary statistics[51]. It considers both the overall structure of the summary statistics and the local structure around each point. The trade-off in local and global structure is controlled through the parameters `min_dist` and `n_neighbors`. Here, we use `min_dist = 0.5` and `n_neighbors = 100`, which emphasises the global structure of the data. 200 training epochs are used to optimise the low-dimensional representation. Each dimension reduction used a different random state. Referring to Figure 7.8, random states of 12094, 72815, and 88212 were used for subfigures (A), (B), and (C) respectively.

The UMAP dimension reductions for the summary statistics simulated under island hops of 2,500, 25,000, and 250,000 years are displayed in Figure 7.8. We notice that class separation appears to increase as the time between migration events increases. The summary statistics of some migration models are still completely overlapping after the times between migration events have increased to 250,000 years; these are also the summary statistics of the migration models that are commonly confused in the confusion matrix from MLR with LASSO (Table 7.4).

By extending the time between migration events through southeast Asia in the original migration models, we have found that the training and test accuracies of a classifier (MLR with LASSO) improve as the time between migration events increases (see Table 7.3). This increasing accuracy is consis-

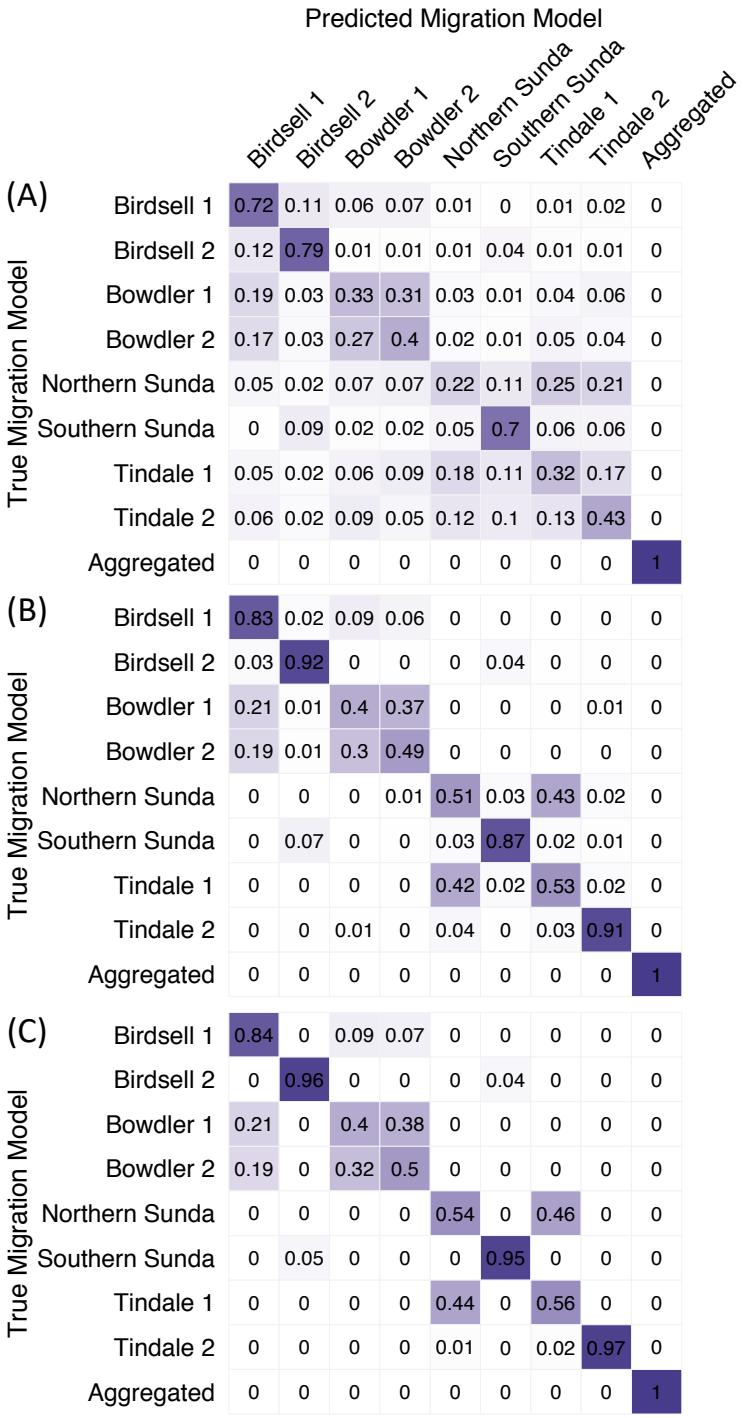


Table 7.4: Confusion matrices for the true and predicted migration models of summary statistics simulated assuming (A) 2,500 years, (B) 25,000 years, and (C) 250,000 years between migration events through southeast Asia. All confusion matrices are calculated from test data. The confusion matrices calculated from the training data are nearly identical, and are presented in Appendix C.3.3. The proportion of observations that fall into each category is given, with rows summing to one. Darker squares indicate greater proportions.

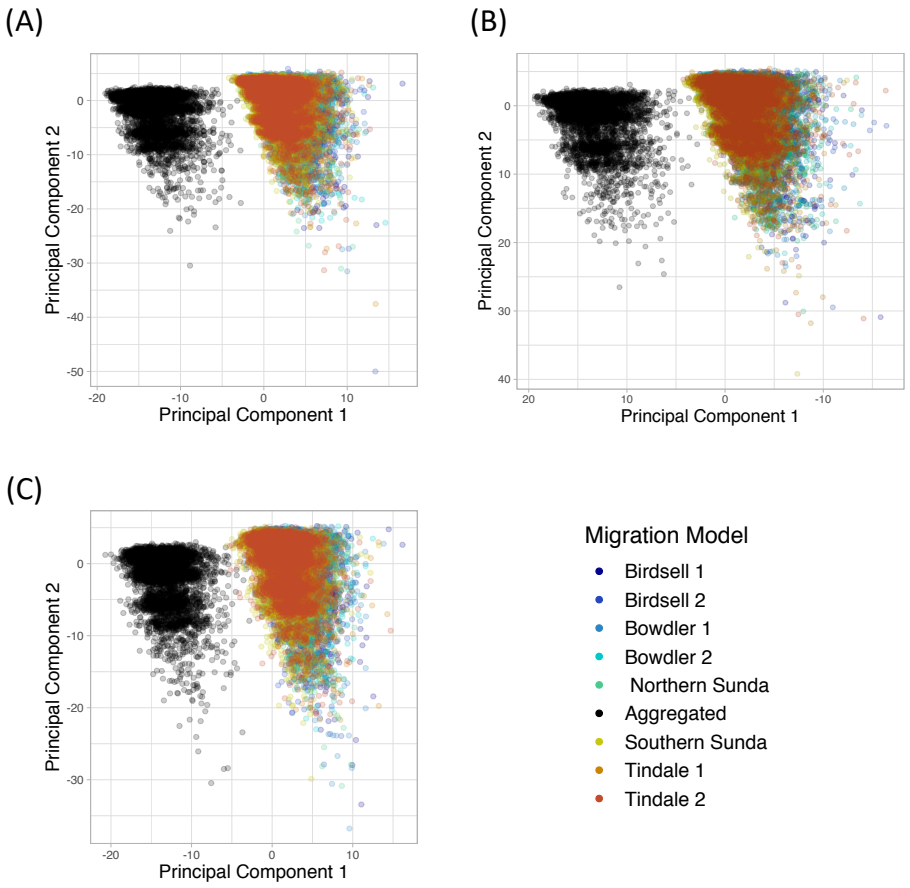


Figure 7.7: PCA plots of the summary statistics assuming (A) 2,500 years, (B) 25,000 years, and (C) 250,000 years between migration events through southeast Asia. There is no perceivable difference between the different PCA plots. Both axes of plot (B) were reversed so that the points were more comparable between all plots.

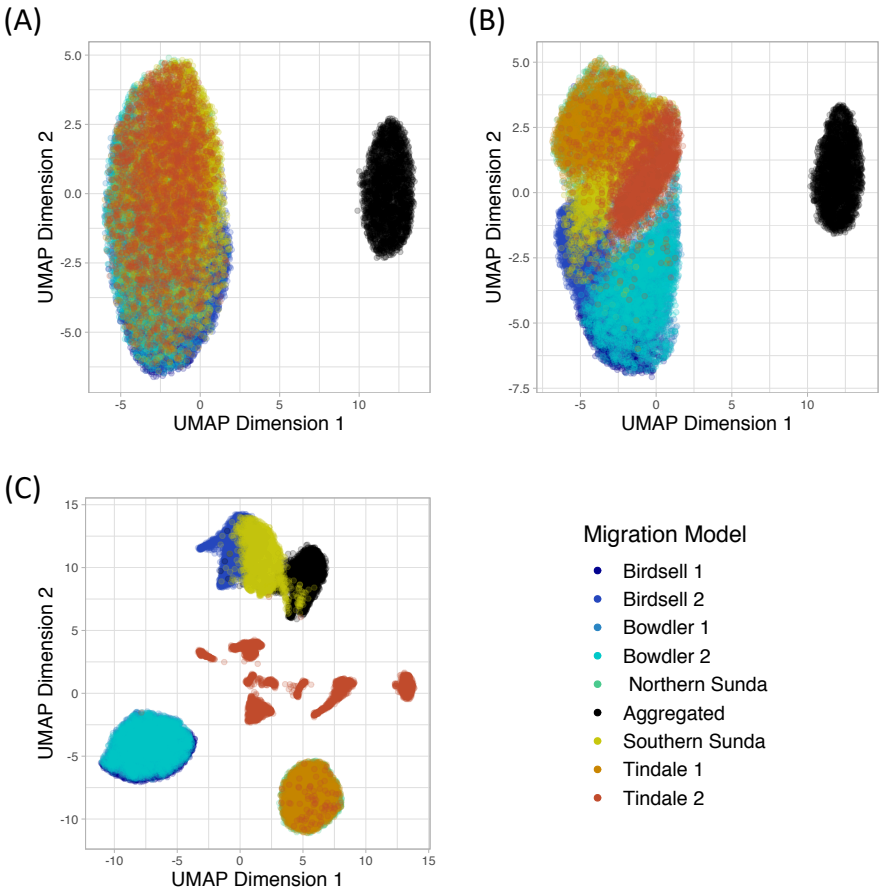


Figure 7.8: UMAP dimension reductions of the summary statistics assuming (A) 2,500 years, (B) 25,000 years, and (C) 250,000 years between migration events through southeast Asia. Summary statistics for different migration models become more distinct as the time between migration events increases.

tent with the greater separation of summary statistics from different models that is observed in UMAP dimension reductions (see Figure 7.8).

We emphasise that even though longer times between migration events resulted in greater separation of summary statistics and greater predictive accuracy of the underlying migration model, we cannot make any conclusions about the observed summary statistics based on summary statistics simulated assuming longer migration events. Waiting 25,000 or 250,000 years between migration events is incredibly unrealistic, and so this data cannot be used to infer what may have happened with shorter, more realistic, times between migration events through southeast Asia.

These analysis conducted in this section suggests that if there is a strong signal of past migration events, as is the case with long times between migration events, it can be detected in the simulated summary statistics. It is very likely that the short times between migration events through the southeast Asian islands result in a weak signal present in the mtDNA sequence data.

7.4 Mitochondrial DNA Haplogroup Analysis

Initially, only geographical location and relatedness were used to filter the mtDNA samples comprising the observed data. All mtDNA haplogroups in a particular population location were included in the population. Here, we investigate whether grouping all mtDNA haplogroups for the population located in southern Australia significantly masked any signal present in the observed data. In the rest of this chapter, we use the general term 'haplogroup' to refer to mtDNA haplogroups.

Tobler *et al.* [85] found that haplogroups R and O were likely to have migrated around the west coast of Australia while haplogroups M, N, P and S were likely to have migrated around the east coast of Australia. These two sets of haplogroups then met in southern Australia. This is reflected in the haplogroups present in the southern Australian population (Population ID #5). A breakdown of all haplogroups for all populations is given in Table 7.5.

To avoid combining mtDNA sequences that arose from potentially different migration histories in one panmictic population, we exclude haplogroups R

Population ID (Location)	Haplogroups
0 (Borneo)	M20 (1), M21a (1), M22a (1), M74b2 (1), N21a (1), N22a (1), N9a6a (2), N9a6b (1), R21(3)
1 (Southern Wallacea)	M21b (1), M73a (1), P1 (1), P1d (3), P1d2 (1) Q3 (1)
2 (New Guinea)	M25 (1), M27a (6), M27b (2), M27c (8), M28a (8), M29a (1), M29b (1), P1 (2), Q1a (1), Q1b (1), Q1c (7), Q1e (2), Q2a (1)
3 (NE Australia)	M (1), M42 (3), M42a (9), N (2), P (3), P4b1 (9), P5 (7), S1 (2), S2 (5)
4 (SE Australia)	M42 (3), M42a (2), P4b1 (1) S1 (1)
5 (S. Australia)	M42 (2), **O (5)**, P (1), P4b1 (6), **R12 (1)**, S (2), S1 (1), S1a (2), S2 (2)

Table 7.5: A breakdown of the mtDNA haplogroups present in each of the population group. Haplogroups corresponding to the suspected migration around the western coast of Australia are written in bold.

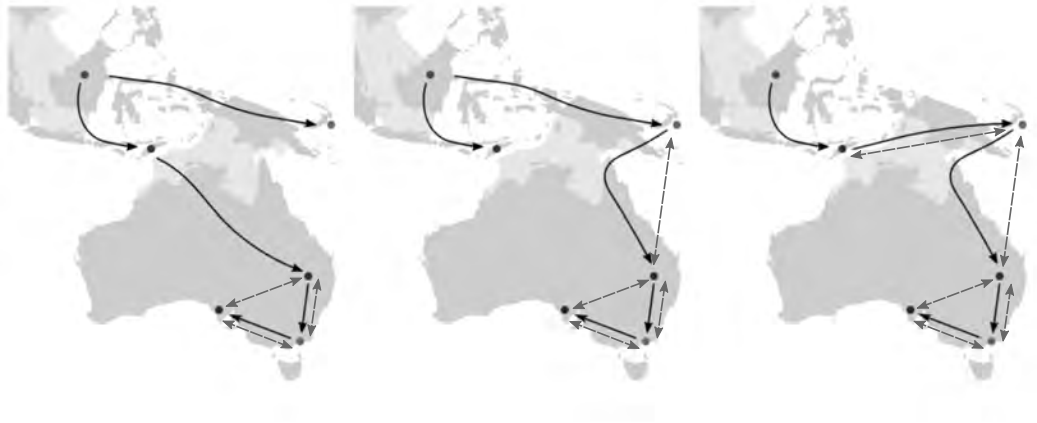


Figure 7.9: Migration routes where migration events (solid arrows) in Australia only occur around the east coast. Distinct patterns of post-settlement migration (dashed arrows) still occur along the migration routes that were taken.
L-R: Bowdler, Tindale/Northern Sunda, and Southern Sunda migration routes.

and O from the samples comprising the southern Australian population. We choose to exclude R and O instead of M, P and S due to sample sizes: combined, there are 6 mtDNA sequences with either haplogroup R or O, but 14 mtDNA sequences with haplogroups M, P, or S. Note that we do not repeat the analysis for haplogroups R and O only due to the small sample size.

Excluding the samples with haplogroups involved in migration along the west coast of Australia means that it is not sensible to include migration along the west coast in candidate migration routes. This reduces the number of migration models: the two Bowdler models are now identical to each other, as are the two Tindale models. The Northern and Southern Sunda routes must also be adapted to include migration along the eastern coast of Australia. After adapting Northern Sunda to include migration along the east coast, it is identical to the Tindale model.

Summary statistics were simulated under the new migration models given in Figure 7.9, *i.e.* without haplogroups R and O present in southern Australia. We explore how excluding haplogroups R and O from the southern Australian population affects the simulated summary statistics, and then investigate whether these new summary statistics result in distinguishable migration models.

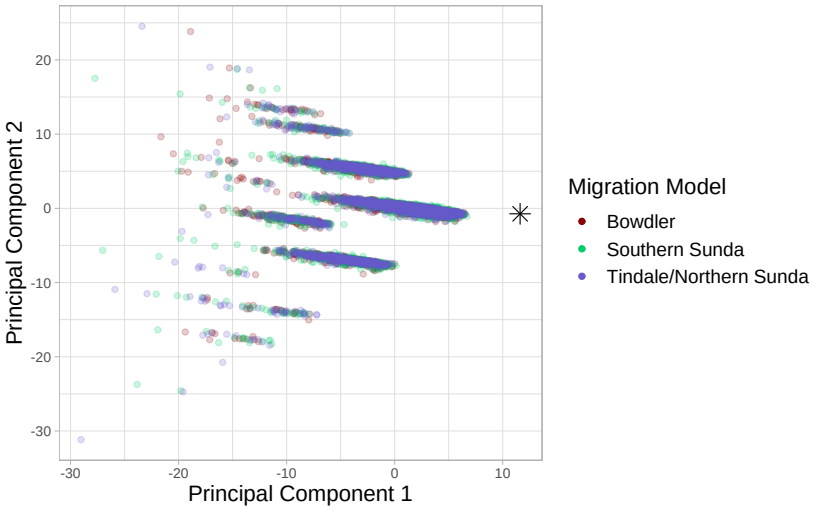


Figure 7.10: A PCA plot of the summary statistics of migration models without migration around the western coast of Australia. The observed summary statistics (recalculated excluding haplogroups R and O) are represented by the black star.

We first visualize the summary statistics using PCA and UMAP dimension reduction techniques. The observed summary statistics were recalculated to exclude haplogroups R and O in the southern Australian population, and are shown as a black star in both dimension reductions.

The PCA plot is given in Figure 7.10, and the UMAP dimension reduction is given in Figure 7.11. For the UMAP dimension reduction, we again used 100 nearest neighbours, a minimum distance of 0.5 and a random state of 70859.

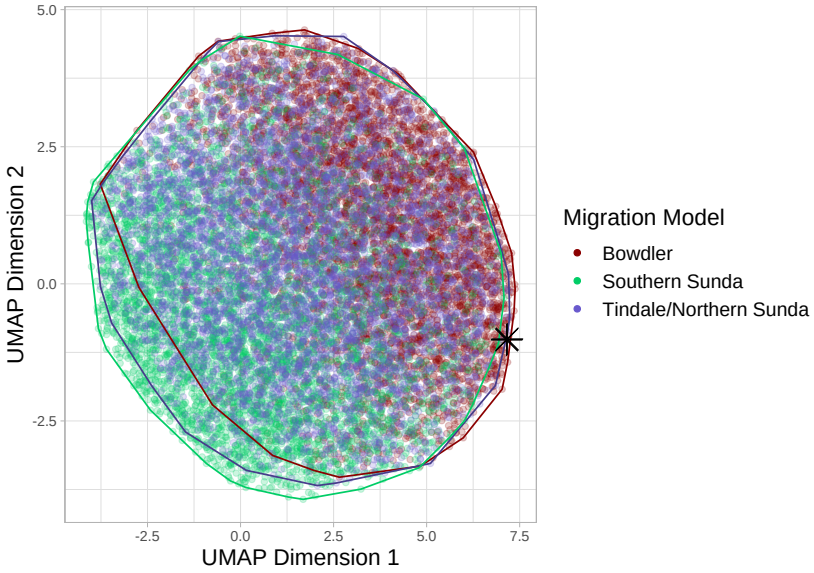


Figure 7.11: A UMAP dimension reduction of the summary statistics of migration models without migration around the western coast of Australia. Convex hulls have been added to better visualise the range of semi-transparent points. The observed summary statistics (recalculated excluding haplogroups R and O) are represented by the black star.

We performed MLR to assess whether a migration model can be reliably selected under these new assumptions, again using LASSO for variable selection. 70% of the data was used as training data, while the remaining 30% was set aside as test data. As in Section 7.1, we used 5-fold cross-validation to determine the λ parameter that defines the penalization term in LASSO.

The MLR classifier had a test accuracy of 72.0% and a training accuracy of 73.1%. For three classes, guessing a migration at random has an expected accuracy of 33.33%, so this classifier is more accurate than random guessing. The confusion matrices for the training and testing data are given in Table 7.6. The probabilities that summary statistics are classified correctly (the diagonal elements) are similar to those for MLR with LASSO in the main results section (see Table 6.7, Section 6.3.1), with the exception of Northern Sunda/Tindale. In this section, approximately 58% of observations that were truly simulated under the Northern Sunda/Tindale migration model were classified correctly; there were fewer correctly classified observations in Table 6.7. This may be because this analysis used a smaller number of migration models that were more distinct than those investigated in Chapter 6.

The estimated probabilities of the observed summary statistics coming from each migration model are given in Table 7.7. The highest class probability is for Southern Sunda, and it is noticeably larger than the probabilities for other migration models.

This is a stronger prediction than that in Chapter 6 based on the probabilities, so we seek to verify our results using support vector machines (SVMs) trained on 15% of the data. A summary of the training and test accuracies are given in Table 7.8. We see that using a Gaussian kernel results in significant overfitting despite the kernel parameters and the cost being determined through a cross-validation process. Hence, we apply an SVM with a linear kernel to the observed data.

Confusion matrices showing misclassification patterns produced by an SVM with a linear kernel are given in Table 7.9. We notice similar patterns to the confusion matrices for MLR with LASSO (see Table 7.6).

Using a linear SVM to predict the migration model that best explains the observed summary statistics results in a prediction of Tindale/Northern Sunda. Looking at the breakdown of binary SVM classifications given in Table 7.10, we find that the Bowdler migration model was never selected in the voting process.

Both classifiers in this section have high enough test accuracy to rule that

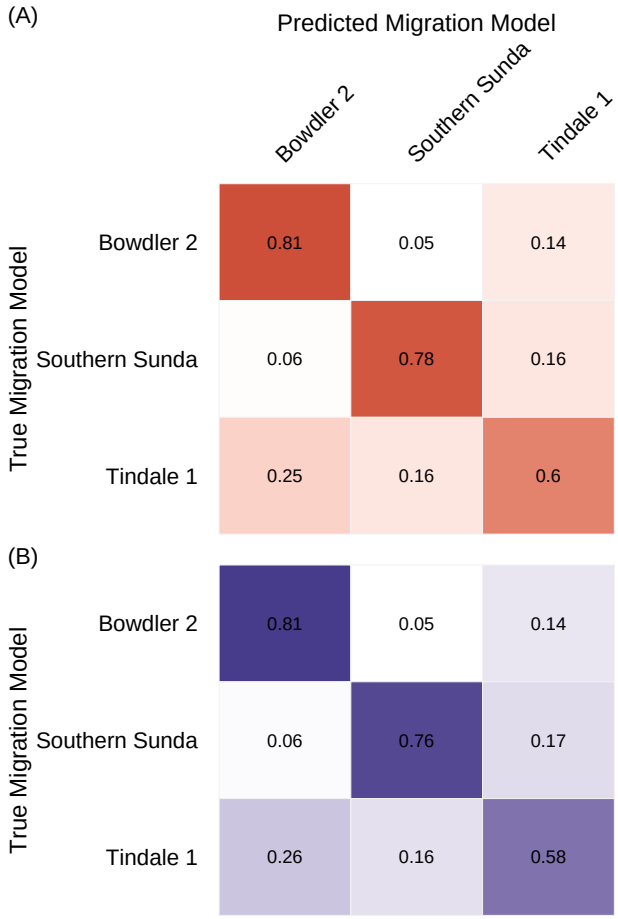


Table 7.6: Confusion matrix for MLR, using LASSO for variable selection. Matrix (A) is calculated based on the training data, while matrix (B) is based on the test data.

Migration Model	Probability
Bowdler	0.127
Tindale/Northern Sunda	0.291
Southern Sunda	0.582

Table 7.7: The output of MLR with LASSO. These are the probabilities that the observed summary statistics are best explained by each migration model.

Kernel	Training Accuracy	Test Accuracy
Linear	74.1%	72.6%
Gaussian	95.1%	63.9%

Table 7.8: Training and test accuracies for SVMs with linear and Gaussian kernels trained on 15% of the data.

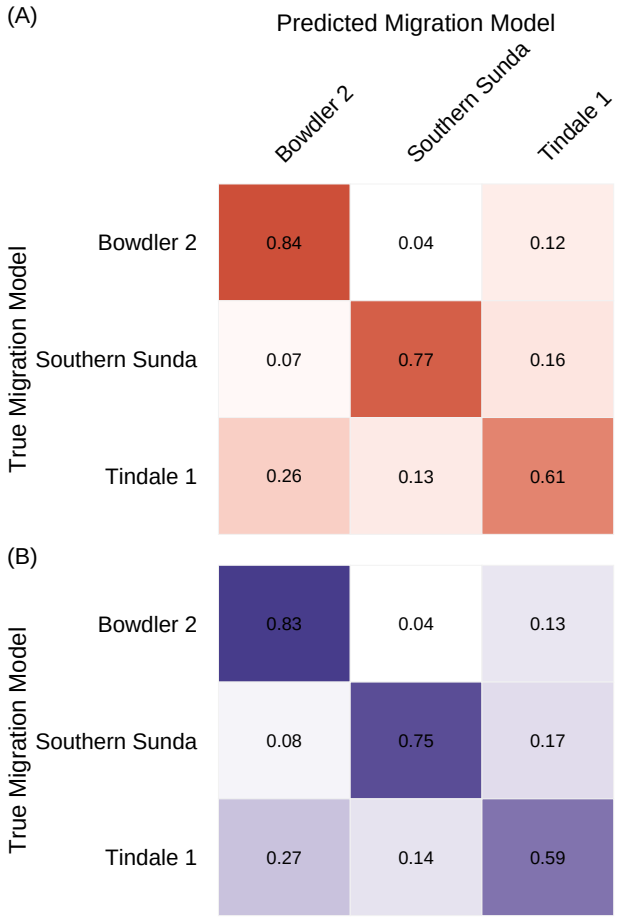


Table 7.9: Confusion matrices for an SVM with a linear kernel trained on 15% of the data. Matrix (A) is calculated based on the training data, while matrix (B) is based on the test data.

Migration Model 1	Migration Model 2	Selected Model
Bowdler	Southern Sunda	Southern Sunda
Bowdler	Tindale/Northern Sunda	Tindale/Northern Sunda
Southern Sunda	Tindale/Northern Sunda	Tindale/Northern Sunda

Table 7.10: Results of all binary classifiers comprising the multiclass SVM classifier.

the observed summary statistics more closely resemble those simulated under the Tindale/Northern Sunda and Southern Sunda migration models than the Bowdler migration model. Referring to Figure 7.9, we see that the common characteristic between the Southern Sunda migration model and the Tindale/Northern Sunda migration model is the entry point to Australia via New Guinea. Considering the patterns of post-settlement migration that occur, Tindale/Northern Sunda and Southern Sunda also both allow migration between New Guinea and northeastern Australia. Hence, we conclude that the observed summary statistics are consistent with a migration model that suggests some level of migration to Australia from New Guinea, and possibly between the two locations.

It does not seem unreasonable to obtain a more concrete result in this haplogroup analysis than for the original simulation study. We note that the original simulation study considered a wider range of models with different migration patterns within Australia. Since we based this extended analysis on the conclusions of Tobler *et al.*, we enforced coastal migration within Australia and discarded all models that suggested migration around the western coast of Australia.

Considering only haplogroups suspected to be involved in migration around the east coast of Australia leads to some limitations on our finding. Since the observed summary statistics are no longer based on southern Australian sequences with mtDNA haplogroups R and O, we cannot comment on the migration paths taken by individuals with mtDNA haplogroups R and O. Therefore the predictions of Southern Sunda and Tindale/Northern Sunda do not rule out a potential different entry point to Australia for haplogroups suspected to be involved in migration around the eastern coast of Australia.

Earlier in Section 7.1, we also found that homogeneous patterns of post-settlement migration significantly reduced the signal present in the summary statistics. As noted earlier in this discussion, it is possible that the migra-

tion models selected merely indicate that post-settlement migration occurred between Australia and New Guinea. To test this, we perform the same analysis but with the assumption of homogeneous post-settlement migration patterns.

After enforcing homogeneous post-settlement migration patterns, MLR with LASSO had a test accuracy of 34.9% and a training accuracy of 35.5%. We note that this is already a considerably lower accuracy than for when we were considering distinct post-settlement migration patterns. The resulting confusion matrices further demonstrate poor predictive accuracy for all migration models. Further details of this analysis can be found in Appendix C.3.4.

This suggests that the predictions of Southern Sunda and Tindale/Northern Sunda by the two different classifiers indicate that the observed summary statistics are consistent with some level of ongoing migration between north-eastern Australia and New Guinea.

Our findings are also not inconsistent with our investigation of how timings between migration events affect the summary statistics (see Section 7.3). We previously concluded that short times between migration events through the southeast Asian islands contributed to the lack of signal in the summary statistics. This is again supported by our findings, since both Southern Sunda and Tindale/Northern Sunda were selected by different classifiers. Southern Sunda assumes a southern migration route through the southeast Asian islands, while Tindale/Northern Sunda assumes a northern migration route through these islands. Furthermore, assuming identical post-settlement migration patterns once again significantly decreased the test accuracy of the classifier.

7.5 Summary of Extended Analysis

First, we sought to explore the effect that using homogeneous post-settlement migration patterns would have on the summary statistics. By using dimension reduction methods and comparing the locations of the summary statistics in the original high-dimensional space, we found that the summary statistics for migration models assuming homogeneous post-settlement migration are more similar than those allowing different patterns of post-settlement migration. Through comparing the results of classification algorithms, we found that a considerable amount of the signal present in the summary statistics

from the initial migration models was indeed a consequence of the patterns of post-settlement migration used.

The findings of Section 7.3.1 suggest that rapid migration through the southeast Asian islands results in a small to negligible amount of signal of the initial migration of Aboriginal Australians to Australia in the first place. Section 7.4 demonstrates that a careful haplogroup analysis combined with a more limited set of migration models can yield some rudimentary findings. We found that the observed summary statistics were consistent with some level of post-settlement migration between New Guinea and northeastern Australia. We reiterate that no conclusion can be drawn about the migration path taken by ancestors of individuals with haplogroups R and O in southern Australia, since they were excluded from analysis.

Chapter 8

Conclusion

In the early chapters of this thesis, we presented all theoretical concepts required to obtain and understand our results. We introduced mitochondrial DNA (mtDNA), coalescent theory, and the DNA simulation process in Chapter 2, and then dimension reduction and classification methods in Chapter 4. The migration models needed to perform simulations for the simulation study were defined in Chapter 5.

We presented the results of our phylogenetic analysis in Chapter 3, which involved reconstructing a maximum likelihood tree using IQ-TREE, reconstructing a phylogenetic tree using BEAST, and then also performing an Extended Bayesian Skyline analysis as part of the BEAST analysis to recover the effective population size over time.

The results of our simulation study were presented in Chapter 6, where we compared the performance of four different multiclass classification methods: multinomial logistic regression (MLR) using forward selection for variable selection, MLR using LASSO for variable selection, support vector machines (SVMs) using a linear kernel, and neural networks. We then used these classifiers to predict the most likely migration model for the observed summary statistics.

Both MLR classifiers selected the Southern Sunda migration model, which includes a southern route through the southeast Asian islands, entry to Australia via New Guinea and then subsequent coastal migration. The SVM and neural network selected the Birdsell 1 migration model, which includes a northern route through the southeast Asian islands, a northwestern entry point to Australia and then no migration between Australian populations.

These migration models are clearly very different. A further validation analysis also found that these results were unstable, and different results were reached if slightly different training data was used (always balanced across classes). In the case of neural networks, different results were obtained by re-running the same algorithm, due to the stochasticity in the training process.

We then conducted further analyses in Chapter 7, which were based on the initial simulation study. The misclassification patterns seen in Chapter 6 led us to consider using the same pattern of post-settlement migration across all migration models. We also explored the effect that extending the times between migration events through the southeast Asian islands had on the simulated summary statistics. We determined how many substitution events we would theoretically expect to see in one lineage for different stages of the migration, and concluded by exploring more restrictive subsetting based on mtDNA haplogroups, along with a reduced set of migration models.

We summarise our key findings in the concluding statements.

8.1 Concluding Statements

The maximum likelihood tree reconstructed with IQ-TREE and the phylogenetic tree reconstructed with BEAST (BEAST tree) were mostly consistent with respect to topology, and produced reasonable results in terms of the ancestry of different mtDNA haplogroups. For the BEAST tree, there was little branch support for the splits that occurred from approximately 45 ka to 55 ka, which is the time that the main migration events were occurring. Branches at a similar point in the maximum likelihood tree had similarly poor branch support. Furthermore, the branches with poor support were estimated to be quite short, which foreshadows the very small number of distinguishing DNA substitutions that are likely to have occurred in this time.

Our simulation study did not give conclusive results by selecting one migration model, or even a family of migration models. We were able to determine that the observed summary statistics were not consistent with the aggregated model, but could have come from any other migration model. We highlight that each classification method, other than MLR with LASSO, clearly chose one migration model, but these selected migration models were not consistent across classification methods. In the case where there is weak phylogenetic

signal in the data, we therefore suggest thoroughly exploring the classification results to make sure that the migration model selected is reliable. We also noticed that misclassification patterns of all classifiers seemed to occur according to the different types of post-settlement migration that were included in the migration models.

In Chapter 7, we first applied the same pattern of post-settlement migration to all models, and found that there was no distinction between the summary statistics of different migration models.

By increasing the times between migration events through the southeast Asian islands, we found that longer times resulted in more distinct migration models. We suggest that the short times between these migration events contributed to the difficulty in selecting a migration model in the initial simulation study. For models to become distinct enough to reliably choose a model, we would have required at least 25,000 years between migration events through the southeast Asian islands. This is clearly an unreasonable length of time, based on the estimated times that modern humans were first present in southeast Asia [94, 63].

Finally, we explored the effect of using further haplogroup subsetting based on the different migration patterns suggested by Tobler *et al.*, with mtDNA haplogroups R and O migrating around the west coast of Australia, and mtDNA haplogroups M, P, and S migrating around the east coast of Australia [85]. This further subsetting also resulted in a smaller number of distinct migration models. The two different migration models selected shared the common characteristic of post-settlement migration between northeastern Australia and New Guinea.

In conclusion, we were unable to reliably select one migration model, and hence could not select a single geographical path from southeast Asia to Australia, based on the information contained in the mtDNA samples that we analysed. Through our extended analysis, we did find that post-settlement migration between northeastern Australia and New Guinea, assuming that coastal migration around the eastern coast of Australia occurred, was consistent with the observed mtDNA data.

8.2 Further Work

Based on limitations that we encountered when specifying the migration models, further research into:

1. the effective population sizes of Aboriginal Australian populations over time, and
2. levels of post-settlement migration between different populations in southeast Asia and Australia

would assist in filling gaps in the current body of literature.

While we did find limited success in identifying post-settlement migration patterns that have occurred over approximately the past 50,000 years, we did not succeed in identifying a particular migration path that was taken from southeast Asia through to Australia. Since summary statistics from different migration models became more distinguishable when the time between migration events was increased, it is likely that the rapid migration events through the southeast Asian islands and then around Australia did not create a strong signal in the mtDNA sequence data.

We recommend that further research into the migration events that resulted in the peopling of Australia is conducted using nuclear DNA, especially if the migration paths of interest are of a finer geographical scale. Furthermore, nuclear DNA is inherited from both parents, and so results determined from nuclear DNA give more detail than the conclusions about the matrilineal history that mtDNA analysis provides.

Appendix A

Calculations with DNA

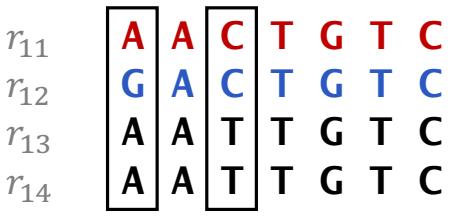
This appendix contains detailed information on summary statistics for DNA, as well as a full definition of all substitution models encountered in this thesis.

A.1 Summary Statistics

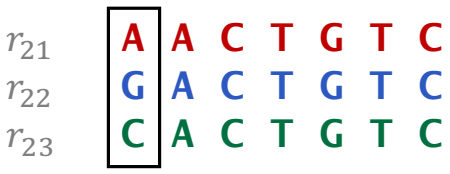
We use two types of summary statistics calculated from DNA sequence alignments: 'within-population summary statistics' and 'between-population summary statistics'. Within-population summary statistics are used to describe the differences between sequences within a population, while between-population summary statistics are used to compare the genetic composition of two or more populations.

In contrast to standard statistical terminology, we use the term 'population' to refer to a subgroup of a DNA alignment in which all sequences come from a similar geographical region. In practice, the DNA sequences in each population are samples from a much larger population that cannot be completely sampled. Suppose that Population 1 is composed of the DNA sequences from four hunter-gatherer individuals from South-East Asia. Since the summary statistics are calculated from only the four DNA sequences sampled from the wider population, they are referred to as *statistics*. If we somehow collected DNA sequences from the entire wider population (all individuals with hunter-gatherer ancestry in South-East Asia), the quantities corresponding to the summary statistics but calculated for all DNA sequences would be called population parameters.

The within-population statistics and between-population statistics calculated by BayeSSC are detailed in the following paragraphs. All examples of within-population statistics are calculated for the artificial example alignment for Population 1 (see Figure A.1).



A.1.1: Population 1



A.1.2: Population 2

Figure A.1: Examples of DNA alignments for two populations. Each row is a DNA sequence for a different individual, while each column is a different site on the genome. Colours represent distinct haplotypes (defined in Section A.1.1), and segregating sites (defined in Section A.1.1) within each population are enclosed in boxes. The j^{th} sequence from the i^{th} population is denoted r_{ij}.

A.1.1 Within-Population Statistics

Number of Unique Haplotypes

BayeSSC considers a haplotype to be any unique form of DNA sequence in the set of simulated DNA sequences, although other definitions are common in the literature. An alternative definition is that a haplotype is a set of Single Nucleotide Polymorphisms (SNPs) inherited together. There are three haplotypes in Population 1, which correspond to the different colours in Figure A.1.1.

Segregating Sites

A segregating site is defined as a position in the DNA alignment that does not have the same nucleotide in all sequences [93].

For example, the first site in Population 1 (Figure A.1.1) is a G for the second sequence, but an A for all other sequences in the alignment. There are two

segregating sites in Population 1, *i.e.* $\hat{S}_1 = 2$. These are indicated by the boxes in Figure A.1.1.

The number of segregating sites is an important quantity that is used to define other statistics, so we also define other forms of segregating sites. The number of segregating sites between sequence i and sequence j is denoted $\hat{S}_{ij}$, *e.g.* in Population 1 (see Figure A.1.1), $\hat{S}_{12} = 1$ and $\hat{S}_{34} = 0$. Finally, Population i will have some true number of segregating sites S_i, but we do not know the true value of this parameter as it would require sequencing the DNA of the entire population.

Average Number of Pairwise Differences

The average number of pairwise differences in Population p is given by

$$\hat{k}_p = \frac{1}{\binom{n_p}{2}} \sum_{i<j} \hat{S}_{ij}, \tag{A.1}$$

where $\hat{S}_{ij}$ is the observed number of segregating sites between sequences i and j, and n_p is the number of sequences in Population p [58]. For Population 1, $n_1 = 4$, and the average number of pairwise differences is

$$\begin{aligned} \hat{k}_1 &= \frac{1}{\binom{4}{2}}(\hat{S}_{12} + \hat{S}_{13} + \hat{S}_{14} + \hat{S}_{23} + \hat{S}_{24} + \hat{S}_{34}) \\ &= \frac{1}{6}(1 + 1 + 1 + 2 + 2 + 0) \\ &= \frac{7}{6}. \end{aligned} \tag{A.2}$$

Haplotype Diversity

The haplotype diversity for Population p is defined as

$$\hat{h}_p = \frac{n_p}{n_p - 1}\left(1 - \sum_{i=1}^{g_p} x_i^2\right), \tag{A.3}$$

where x_i is the proportion of samples with haplotype i, n_p is the number of samples in Population p, and g_p is the total number of unique haplotypes in Population p [59]. The term $1 - \sum x_i^2$ is the observed probability of randomly selecting two different haplotypes from all haplotypes in the sample, when

sampling with replacement. The probability is multiplied by $n_p/(n_p - 1)$ to give the intuitive property that a population with only unique haplotypes has a haplotype diversity of one. This also serves to make the statistic comparable between different-sized populations.

For Population 1, $n_1 = 4$ and $g_1 = 3$. The observed haplotype diversity is then

$$\begin{aligned}\hat{h}_1 &= \frac{4}{3}\left(1 - \left(\frac{1}{4}\right)^2 - \left(\frac{1}{4}\right)^2 - \left(\frac{1}{2}\right)^2\right) \\ &= \frac{4}{3}\left(\frac{5}{8}\right) \\ &= \frac{5}{6}.\end{aligned} \tag{A.4}$$

Nucleotide Diversity

Nucleotide diversity is the observed average proportion of segregating sites. It is defined as

$$\hat{\pi}_p = \sum_{i<j} f_i f_j \pi_{ij},$$

where f_i and f_j are the frequencies of sequence i and sequence j in population p, and π_{ij} is the number of nucleotide differences per site [58]. In this formulation, π_{ij} are only calculated for distinct i and j, *i.e.* each distinct pair should contribute only once to the summation. It can also be rewritten as a scalar multiple of the average number of pairwise differences:

$$\hat{\pi}_p = \frac{1}{\ell}\hat{k}_p,$$

for sequences of length ℓ.

In Figure A.1.1, sequences are of length $\ell = 7$, and so we can calculate the nucleotide diversity from the value of the pairwise differences calculated in Equation A.2:

$$\begin{aligned}\hat{\pi}_1 &= \frac{1}{7}\left(\frac{7}{6}\right) \\ &= \frac{1}{6}.\end{aligned}$$

Tajima's D

Tajima's D is a statistic that was formulated for testing the hypothesis that a population is evolving under neutral conditions. This is calculated by comparing two estimators for θ, which is the expected number of mutations between any two sequences. A value of Tajima's D significantly different from zero implies either a change in population size or evidence for selection [81]. Note that the true value of θ is often unknown since it is a population parameter, and so to define Tajima's D, we first need to define Watterson's estimator for θ, $\hat{\theta}_W$. We begin by considering the expected number of segregating sites in population j, $E[S_j]$.

$$E[S_j] = \theta \sum_{i=1}^{n_p-1} \frac{1}{i}, \tag{A.5}$$

where n_p is the number of sequences. Rearranging equation A.5,

$$\begin{aligned} \theta &= E[S_j] \left(\sum_{i=1}^{n_p-1} \frac{1}{i} \right)^{-1} \\ \Rightarrow \hat{\theta}_W &= \hat{S}_j \left(\sum_{i=1}^{n_p-1} \frac{1}{i} \right)^{-1}. \end{aligned} \tag{A.6}$$

Note that the observed number of segregating sites in population j, $\hat{S}_j$ is not necessarily the same as expected number of segregated sites $E[S_j]$, and is only used to estimate this value.

Tajima's D is then defined as

$$D = \frac{\hat{\theta}_W - \hat{k}}{\sqrt{\mathrm{Var}(\hat{\theta}_W - \hat{k})}},$$

where $\hat{k}$ is the average number of pairwise differences (Equation A.1). Both $\hat{\theta}_W$ and $\hat{k}$ are unbiased estimators for θ, but selection or changes in population size will affect these values differently, potentially resulting in significant non-zero values of D. Details for estimating the variance of D can be found in Tajima's original paper [81]. For Population 1, $D \approx 0.5916$. Detailed calculations are omitted for this statistic.

A.1.2 Between-Population Statistics

Private Alleles and Private Haplotypes

Alleles and haplotypes are said to be *private* if they appear in one population only [76], and are then said to be 'private to' that specific population. For example, Populations 1 and 2 each have one private haplotype (the black and green haplotypes respectively), as the red and blue haplotypes are found in both populations. The number of private alleles and private haplotypes are both summary statistics, and are calculated for each pair of populations.

Average Number of Pairwise Differences

Consider two populations p_1 and p_2, which are made up of n_{p_1} and n_{p_2} aligned sequences respectively. The average number of pairwise differences between populations p_1 and p_2 is similar to the number of pairwise differences within a population, except sequences i and j must be from different populations. We define the average number of pairwise differences between populations p_1 and p_2 as $\hat{k}_{p_1p_2}$:

$$\hat{k}_{p_1p_2} = \frac{1}{n_{p_1} n_{p_2}} \sum_{i,j} \hat{S}_{ij}, \tag{A.7}$$

where $\hat{S}_{ij}$ is defined in subsection A.1.1. For the populations in Figure A.1, $n_1 = 4$, $n_2 = 3$, and the segregating sites between sequences are given in Table A.1.

	r_{21}	r_{22}	r_{23}
r_{11}	0	1	1
r_{12}	1	0	1
r_{13}	1	2	2
r_{14}	1	2	2

Table A.1: Number of segregating sites between sequences from Population 1 and sequences from Population 2 in Figure A.1. Sequence r_{pi} is the ith sequence in population p.

It follows that the average number of pairwise differences between the two populations is

$$\begin{aligned}\hat{k}_{12} &= \frac{1}{12}(1+1+1+1+1+2+2+1+2+2) \\ &= \frac{7}{6}. \end{aligned} \tag{A.8}$$

Mean Diversity $\bar{H}_{ij}$

Mean (haplotype) diversity is calculated by finding the haplotype diversity of the two separate populations, and then taking the arithmetic mean. For any two Populations i and j with haplotype diversities $\hat{h}_i$ and $\hat{h}_j$ respectively, the mean haplotype diversity is $\bar{H}_{ij} = (\hat{h}_i + \hat{h}_j)/2$.

The haplotype diversity of Population 1 is 5/6, which was calculated in Section A.1.1. The haplotype diversity for Population 2 is 1, and so the mean diversity of Population 1 and Population 2 is

$$\begin{aligned}\bar{H}_s &= \frac{1}{2}\left(\frac{5}{6}+1\right) \\ &= \frac{11}{12}. \end{aligned}$$

Pooled Diversity H_T

Pooled (haplotype) diversity is calculated by merging two or more populations, and then finding the haplotype diversity for the single population using Equation A.3. BayeSSC calculates this statistic by pooling only two populations at a time. In the pooled population made up of Populations 1 and 2 (Figure A.1), there are two 'red' sequences, two 'blue' sequences, two 'black' sequences, and one 'green' sequence (each colour represents a different haplotype). There are seven sequences in the pooled population, and so the pooled haplotype diversity is

$$\begin{aligned}H_T &= \frac{7}{6}\left(1-\left(\frac{2}{7}\right)^2-\left(\frac{2}{7}\right)^2-\left(\frac{2}{7}\right)^2-\left(\frac{1}{7}\right)^2\right) \\ &= \frac{7}{6}\left(\frac{36}{49}\right) \\ &= \frac{6}{7}. \end{aligned}$$

Fixation Index F_{ST}

F_{ST} was first introduced by Sewall Wright in 1931, and many formulae have been developed since its first definition. We present the formula derived by Hudson [36],

$$F_{ST} = 1 - \frac{H_W}{H_B}, \tag{A.9}$$

where H_W is the mean of the average number of pairwise differences within populations, and H_B is the average number of pairwise differences between populations. When calculating F_{ST} for populations i and j, note that

$$H_W = \frac{\hat{k}_1 + \hat{k}_2}{2}, \text{ and}$$
$$H_B = \hat{k}_{12}.$$

While other equations for F_{ST} have been developed, Hudson's estimator of F_{ST} was used for the analysis in Chapters 6 and 7 due to its automatic inclusion in the output of BayeSSC.

For example, F_{ST} is calculated below for Population 1 and Population 2 (Figure A.1) using equation A.9. Since the average number of pairwise differences for Populations 1 and 2 separately are $\hat{k}_1 = 7/6$ and $\hat{k}_2 = 1$,

$$\begin{aligned} H_W &= \frac{1 + 7/6}{2} \\ &= \frac{13}{12}. \end{aligned} \tag{A.10}$$

From equation A.8, $H_B = 7/6$. Then

$$\begin{aligned} F_{ST} &= 1 - \frac{13/12}{7/6} \\ &= 1 - \frac{13}{14} \\ &= \frac{1}{14}. \end{aligned}$$

Conceptually, F_{ST} quantifies genetic differences between populations due to population structure [61]. If two populations have a low value of F_{ST}, they are more similar, while a higher value indicates greater difference between the populations. The range of values considered 'high' or 'low' depends on context - a large value of F_{ST} for two populations of the same species may be considered small for two populations of different species.

A.2 Substitution Models

Substitution models define the rates at which different types of substitutions occur. These models are first introduced in Section 2.4.5, which mentions the Jukes-Cantor, Kimura two-parameter, and GTR substitution models. Here, we mathematically define all substitution models that are mentioned in this thesis.

For brevity, we define the set of all nucleotides as $D = \{A, C, G, T\}$. We present the substitution model as a matrix of the form

$$Q = \begin{pmatrix} q_A & q_{AG} & q_{AC} & q_{AT} \\ q_{GA} & a_G & q_{GC} & q_{GT} \\ q_{CA} & q_{CG} & q_C & q_{CT} \\ q_{TA} & q_{TG} & q_{TC} & q_T \end{pmatrix}, \tag{A.11}$$

where q_{ij} is the rate at which nucleotide j becomes nucleotide i, and

$$q_i = -\sum_{k \in D \setminus \{i\}} q_{ik}.$$

All substitution models presented here model the substitution process as a continuous-time Markov chain. This means that the time between substitutions is exponentially distributed with rate q_i, where i is the current state, and that the next state (the substitution that occurs next) only depends on the current state.

Since q_{ij} is non-negative for all $i, j \in D, i \neq j$, all q_i are non-negative and finite, and all rows sum to zero, Matrix A.11 satisfies all conditions for a 'transition rate matrix', which defines the transitions of a continuous-time Markov chain. This terminology may be ambiguous, as transitions and transversions are also types of substitutions that can occur. We will instead use the term 'substitution rate matrix'.

We define the proportion of A's, G's, C's, and T's in a DNA sequence as π_A, π_G, π_C, and π_T respectively. If a substitution model assumes equal base frequencies, then $\pi_A = \pi_G = \pi_C = \pi_T = 1/4$. Models with unequal base frequencies allow these values to be different with the requirement that

$$\sum_{i \in D} \pi_i = 1.$$

All parameters for substitution models, including unequal base frequencies, can be estimated from a sequence alignment using a program such as ModelGenerator [42]. We note that the simulation program BayeSSC does not support substitution models with unequal base frequencies.

A.2.1 Jukes-Cantor model

In the Jukes-Cantor model, all substitutions are equally likely [39]. This corresponds to the substitution rate matrix

$$Q_{\text{JC69}} = \begin{pmatrix} -3\mu/4 & \mu/4 & \mu/4 & \mu/4 \\ \mu/4 & -3\mu/4 & \mu/4 & \mu/4 \\ \mu/4 & \mu/4 & -3\mu/4 & \mu/4 \\ v & \mu/4 & \mu/4 & -3\mu/4 \end{pmatrix}.$$

A.2.2 Kimura two-parameter model

The Kimura two parameter model, commonly abbreviated to the K2P or K80 model, allows transitions to occur at a different rate than transversions [43]. Let transitions occur with rate r and transversions occur with rate ν. Then the substitution rate matrix is

$$Q_{K2P} = \begin{pmatrix} -(r+2\nu) & r & \nu & \nu \\ r & -(r+2\nu) & \nu & \nu \\ \nu & \nu & -(r+2\nu) & r \\ \nu & \nu & r & -(r+2\nu) \end{pmatrix}.$$

This substitution model can also be parameterised in terms of the transition-transversion ratio κ, where transitions are expected to occur κ times more frequently than transversions.

A.2.3 Kimura three-parameter model

The Kimura three-parameter model, commonly abbreviated as the K81 or K3P model, extends the Kimura two-parameter model by allowing different rates for two different types of transversions. The rate of substitutions between A and C, and T and G is ν_1; the rate of substitutions between A and

T, and C and G is ν_2. Transitions occur at rate r. The substitution rate matrix is

$$Q_{K3P} = \begin{pmatrix} -(r+\nu_1+\nu_2) & r & \nu_1 & \nu_2 \\ r & -(r+\nu_1+\nu_2) & \nu_2 & \nu_1 \\ \nu_1 & \nu_2 & -(r+\nu_1+\nu_2) & r \\ \nu_2 & \nu_1 & r & -(r+\nu_1+\nu_2) \end{pmatrix}.$$

A.2.4 HKY model

The HKY model [32] is named after its creators Hasegawa, Kishino, and Yano. This model extends the Kimura two-parameter model by allowing unequal base frequencies.

To keep notation compact, we will denote diagonal elements as q_i, where i is the state and

$$q_i = -\sum_{k \in D \setminus \{i\}} q_{ik}$$

as defined earlier in this section.

The substitution rate matrix is

$$Q_{HKY} = \begin{pmatrix} q_A & \kappa\pi_G & \pi_C & \pi_T \\ \kappa\pi_A & q_G & \pi_C & \pi_T \\ \pi_A & \pi_G & q_C & \kappa\pi_T \\ \pi_A & \pi_G & \kappa\pi_C & q_T \end{pmatrix}.$$

A.2.5 Tamura-Nei model

The Tamura-Nei substitution model extends the HKY model by allowing different rates for each type of transition [82]. Suppose that transitions between A and G occur at rate κ_{AG}, and transitions between C and T occur at rate κ_{CT}. This model also allows unequal base frequencies. The substitution rate matrix then has the form

$$Q_{TN} = \begin{pmatrix} q_A & \kappa_{AG}\pi_G & \pi_C & \pi_T \\ \kappa_{AG}\pi_A & q_G & \pi_C & \pi_T \\ \pi_A & \pi_G & q_C & \kappa_{CT}\pi_T \\ \pi_A & \pi_G & \kappa_{CT}\pi_C & q_T \end{pmatrix}.$$

A.2.6 Transversion model

In the transversion model, or TVM, each type of transversion occurs at a different rate [99]. The model assumes equal base frequencies by default, but this assumption can be relaxed. Using similar notation to other substitution models that have a single rate for all transitions and different rates for different transversions, we will denote the rate at which transitions occur as r and the rates of different transversions as ν_1, ν_2, ν_3 and ν_4.

The substitution rate matrix has the form

$$Q_{TVM} = \begin{pmatrix} q_A & r & \nu_1 & \nu_2 \\ r & q_G & \nu_3 & \nu_4 \\ \nu_1 & \nu_3 & q_C & r \\ \nu_2 & \nu_4 & r & q_T \end{pmatrix}.$$

A.2.7 Generalised Time Reversible model

As mentioned in Chapter 2, the GTR model is one of the most general substitution models, allowing a different rate for each type of substitution and accommodating unequal base frequencies [83]. It can be represented by the substitution rate matrix Q_{GTR}, where

$$Q_{GTR} = \begin{pmatrix} q_A & r_1 & \nu_1 & \nu_2 \\ r_1 & q_G & \nu_3 & \nu_4 \\ \nu_1 & \nu_3 & q_C & r_2 \\ \nu_2 & \nu_4 & r_2 & q_T \end{pmatrix}.$$

Appendix B

DNA sample metadata

B.1 Sunda and Sahul mtDNA

In Table B.1 we present a more detailed description of the sample locations previously described in Table 3.1, Chapter **??**. Accession numbers and references for the aligned mtDNA sequences in the filtered dataset are given in Table B.2.

General Location	Population ID Number	Population/ Sampling Location	n. Samples (Unfiltered)	n. Samples (Filtered)
Borneo	0	Bidayuh	10	3
	0	Jehai	19	3
	0	Seletar	15	1
	0	Temuan	14	5
Philippines	NA	Mindanao	19	0
	NA	Bataan	9	0
	NA	Babuyan Island	2	0
	NA	Luzon	2	0
	NA	Capul Island	1	0
Timor	1	East Timor	15	8

Table B.1: Detailed information about the sample locations of DNA sequences or the populations of the individuals from which mtDNA samples were obtained, as well as the number of samples before and after filtering.

General Location	Population ID Number	Population/ Sampling Location	n. Samples (Unfiltered)	n. Samples (Filtered)
Remote Oceania	NA	Cook Island	1	0
	NA	Fiji	8	0
	NA	Futuna	2	0
	NA	Niue	1	0
	NA	Samoa	3	0
	NA	Tonga	4	0
	NA	Tuvalu	1	0
Santa Cruz	NA	Santa Cruz	40	0
New Britain	2	New Britain	8	4
	2	Nakanai	25	6
	2	Ata	15	4
Bougainville (New Guinea)	2	Buin	21	8
	2	Buka	4	3
	NA	Nagovisi	1	0
	2	Nasioi	20	9
	2	Siwai	8	1
	2	Torau	15	6
Solomon Islands	NA	Choiseul	7	0
	NA	Gela	11	0
	NA	Guadalcanal	13	0
	NA	Isabel	7	0
	NA	Kolombangara	2	0
	NA	Makira	2	0
	NA	Malaita	29	0
	NA	Ranonga	18	0
	NA	Russel	3	0
	NA	Savo	5	0
	NA	Shortland	5	0
	NA	Vella Lavella	8	0

Table B.1: Detailed information about the sample locations of DNA sequences or the populations of the individuals from which mtDNA samples were obtained, as well as the number of samples before and after filtering.

General Location	Population ID Number	Population/ Sampling Location	n. Samples (Unfiltered)	n. Samples (Filtered)
Australia	NA	Australia *(no location)*	124	0
	4	Lake Tyers	14	7
	3	Cherbourg	23	21
	5	Koonibba	47	10
	3	Brewarrina	31	20
	5	Point Pearce	43	12

Table B.1: Detailed information about the sample locations of DNA sequences or the populations of the individuals from which mtDNA samples were obtained, as well as the number of samples before and after filtering.

Reference	Sample Location(s)	Accession Numbers
Duggan *et al.* [21]	Bougainville, New Britain	KJ154486, KJ154492, KJ154496, KJ154500, KJ154508, KJ154518, KJ154544, KJ154546, KJ154554, KJ154579, KJ154581, KJ154584, KJ154590, KJ154600, KJ154601, KJ154603, KJ154604, KJ154618, KJ154621, KJ154624, KJ154625, KJ154626, KJ154630, KJ154631, KJ154634, KJ154635, KJ154638, KJ154851, KJ154852, KJ154859, KJ154861, KJ154866, KJ154887, KJ154888, KJ154895, KJ154904, KJ154928, KJ154932, KJ154933, KJ154934, KJ154937
Gomes *et al.* [29]	East Timor	KJ676777, KJ676780, KJ676784, KJ676786, KJ676787, KJ676788, KJ676789, KJ676790
Jinam *et al.* [37]	Borneo	AP012349, AP012354, AP012368, AP012383, AP012391, AP012392, AP012413, AP012421, AP012422, AP012425, AP012430, AP012431
Tobler *et al.* [85]*	Cherbourg, Point Pearce, Koonibba	PRJEB15344
(Unpublished)	Brewarrina, Lake Tyers	-

Table B.2: GenBank accession numbers, sample locations, and references for all sequences used in this study.

* All sequences from Tobler *et al.* were submitted to the European Nucleotide Archive (ENA) under the study accession number provided.

B.2 Neanderthal mtDNA

The accession numbers of the Neanderthal sequences used to root the maximum likelihood tree in Section 3.2.3 are FM865407, FM865408, FM865409, FM865410, and FM865411 [11].

Appendix C

Supplementary Figures and Tables

C.1 Extended Results from Chapter 3

C.1.1 Densitree summary

Figure C.1 shows the DensiTree output, which superimposes all trees in the posterior sample after burn-in was removed [9]. Each tree is partly transparent, so that darker regions indicate greater support. We notice that there is a clear split between the samples AP012430.1_N22a and KJ154851.1_M28a7b, which is indicative of the deep split between haplogroups M and N. We also notice that some haplogroups form distinct monophyletic clades, such as M28, M42a, and P4b1.

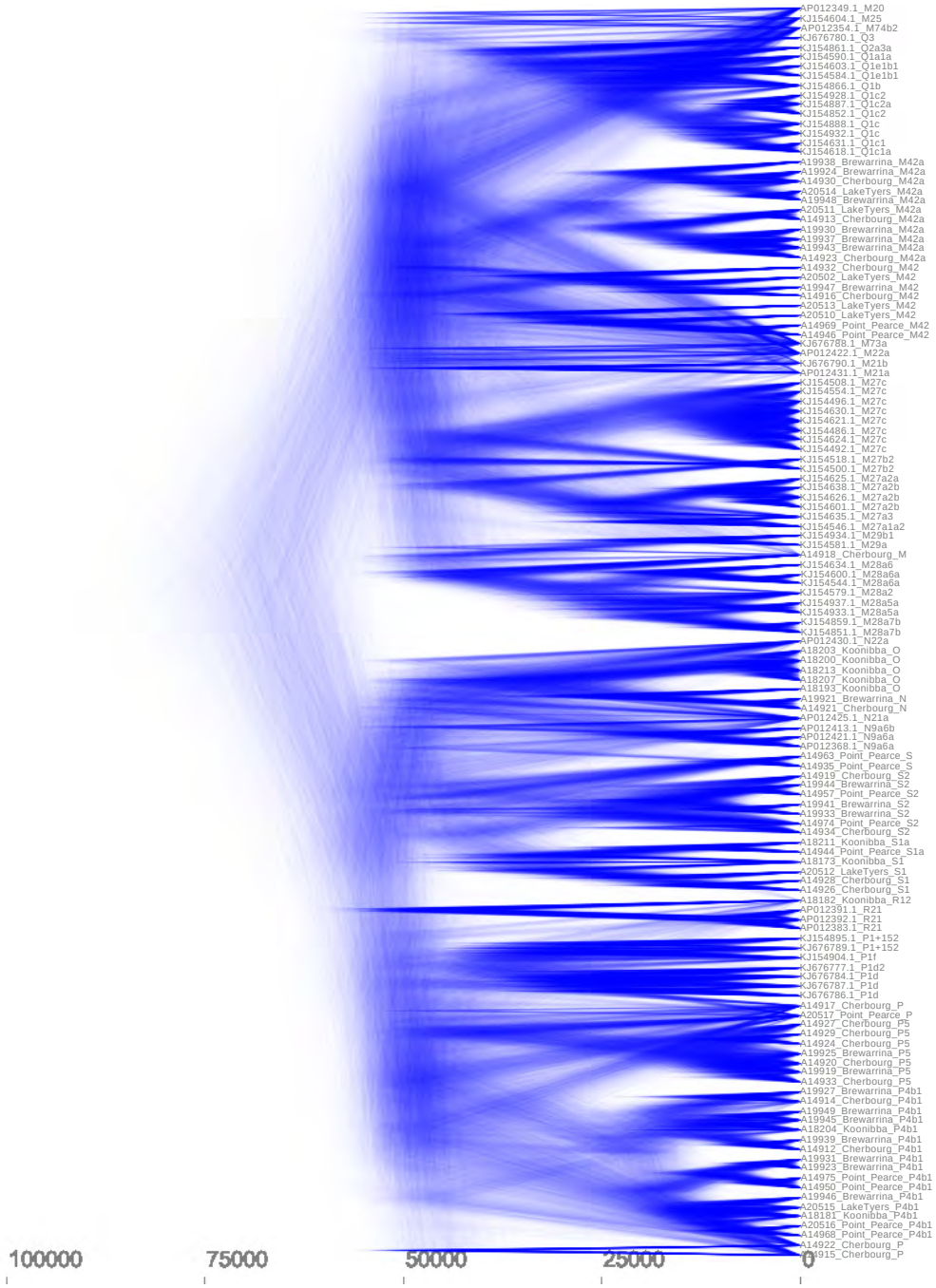


Figure C.1: DensiTree output showing all trees in the posterior sample. The timescale is given in the number of years before present.

C.1.2 Effective Sample Sizes (ESSs) for all variables from the combined posterior samples

In Section 3.5.2, we ran three MCMC chains via BEAST. We then combined the posterior samples from these chains after removing burn-in samples, which were 10%, 75%, and 10% of all samples in each chain, respectively. The ESSs for all parameters were calculated with Tracer v1.7.1 [71], and are presented here in Tables C.1, C.2, C.3, and C.4.

Parameter	ESS	Parameter	ESS
indicators.alltrees1	2178.7	indicators.alltrees31	9102.1
indicators.alltrees2	2637.9	indicators.alltrees32	4142.9
indicators.alltrees3	3836.3	indicators.alltrees33	7606
indicators.alltrees4	3515.9	indicators.alltrees34	5530.3
indicators.alltrees5	3319.2	indicators.alltrees35	5990.3
indicators.alltrees6	2513.3	indicators.alltrees36	8104.2
indicators.alltrees7	4474.3	indicators.alltrees37	8795.6
indicators.alltrees8	5075.4	indicators.alltrees38	8752
indicators.alltrees9	6414.2	indicators.alltrees39	6701.6
indicators.alltrees10	5422.6	indicators.alltrees40	5449.2
indicators.alltrees11	6063.1	indicators.alltrees41	8848.3
indicators.alltrees12	5033.1	indicators.alltrees42	10805.2
indicators.alltrees13	6960.2	indicators.alltrees43	4425.6
indicators.alltrees14	5767.3	indicators.alltrees44	9452
indicators.alltrees15	5598.3	indicators.alltrees45	5969.5
indicators.alltrees16	5551.2	indicators.alltrees46	7510.9
indicators.alltrees17	5253.1	indicators.alltrees47	8715.5
indicators.alltrees18	7091.6	indicators.alltrees48	8573.2
indicators.alltrees19	4801.9	indicators.alltrees49	7395.6
indicators.alltrees20	5434	indicators.alltrees50	9046.9
indicators.alltrees21	6729.2	indicators.alltrees51	6090.8
indicators.alltrees22	7385.6	indicators.alltrees52	6513
indicators.alltrees23	8162.5	indicators.alltrees53	5456.8
indicators.alltrees24	8206.4	indicators.alltrees54	9465
indicators.alltrees25	7866.2	indicators.alltrees55	7228.2
indicators.alltrees26	7415.9	indicators.alltrees56	10083.8
indicators.alltrees27	6267.1	indicators.alltrees57	9328.6
indicators.alltrees28	6050.9	indicators.alltrees58	9430
indicators.alltrees29	9724.5	indicators.alltrees59	7155.2
indicators.alltrees30	6129.9	indicators.alltrees60	8672.5

Table C.1: ESSs for the indicators.alltrees parameters.

Parameter	ESS
indicators.alltrees61	9817.4
indicators.alltrees62	10694.3
indicators.alltrees63	12646
indicators.alltrees64	9053.8
indicators.alltrees65	10280.5
indicators.alltrees66	10406.6
indicators.alltrees67	9668.9
indicators.alltrees68	9964.4
indicators.alltrees69	10781.8
indicators.alltrees70	10014.5
indicators.alltrees71	12347.8
indicators.alltrees72	10662
indicators.alltrees73	6212.9
indicators.alltrees74	10863.5
indicators.alltrees75	10492.4
indicators.alltrees76	7161.6
indicators.alltrees77	10610.5
indicators.alltrees78	8338.3
indicators.alltrees79	10973
indicators.alltrees80	8579.1
indicators.alltrees81	10104.7
indicators.alltrees82	11382.1
indicators.alltrees83	11736.1
indicators.alltrees84	8430.2
indicators.alltrees85	6437.4
indicators.alltrees86	10173
indicators.alltrees87	8983.4
indicators.alltrees88	7026.9
indicators.alltrees89	9447
indicators.alltrees90	7906.5
indicators.alltrees91	8273.6
indicators.alltrees92	8382
indicators.alltrees93	7791
indicators.alltrees94	4700.1
indicators.alltrees95	8017.4
indicators.alltrees96	4700.6
indicators.alltrees97	4832.7
indicators.alltrees98	3303
indicators.alltrees99	445.1
indicators.alltrees100	1455.3
indicators.alltrees101	1963.1
indicators.alltrees102	2187.5
indicators.alltrees103	1550.5
indicators.alltrees104	695.3
indicators.alltrees105	1395.1
indicators.alltrees106	1323.2
indicators.alltrees107	704.7
indicators.alltrees108	807.3
indicators.alltrees109	145.5
indicators.alltrees110	98.6

Parameter	ESS
indicators.alltrees111	64.2
indicators.alltrees112	434.7
indicators.alltrees113	206.7
indicators.alltrees114	97
indicators.alltrees115	180.4
indicators.alltrees116	-
indicators.alltrees117	-
indicators.alltrees118	17
indicators.alltrees119	8.9
indicators.alltrees120	-
indicators.alltrees121	-
indicators.alltrees122	7.6
indicators.alltrees123	-
indicators.alltrees124	-
indicators.alltrees125	-
indicators.alltrees126	-
indicators.alltrees127	-
indicators.alltrees128	-
indicators.alltrees129	13294
indicators.alltrees130	10355.4

Table C.2: ESSs for the remaining indicators.alltrees parameters.

Parameter	ESS	Parameter	ESS
popSizes.alltrees1	2282.5	popSizes.alltrees51	758.3
popSizes.alltrees2	1192.9	popSizes.alltrees52	793.5
popSizes.alltrees3	777.3	popSizes.alltrees53	518.7
popSizes.alltrees4	648.9	popSizes.alltrees54	831.9
popSizes.alltrees5	2017.9	popSizes.alltrees55	796.4
popSizes.alltrees6	1177.9	popSizes.alltrees56	449.5
popSizes.alltrees7	800.7	popSizes.alltrees57	210
popSizes.alltrees8	2040.6	popSizes.alltrees58	978.3
popSizes.alltrees9	2428.1	popSizes.alltrees59	91.1
popSizes.alltrees10	747.1	popSizes.alltrees60	441.8
popSizes.alltrees11	797.2	popSizes.alltrees61	395.6
popSizes.alltrees12	934.2	popSizes.alltrees62	552.1
popSizes.alltrees13	995.2	popSizes.alltrees63	850.1
popSizes.alltrees14	400.3	popSizes.alltrees64	333.7
popSizes.alltrees15	715.3	popSizes.alltrees65	474
popSizes.alltrees16	602.7	popSizes.alltrees66	710
popSizes.alltrees17	199.8	popSizes.alltrees67	716.1
popSizes.alltrees18	935.1	popSizes.alltrees68	58.3
popSizes.alltrees19	519.8	popSizes.alltrees69	812.1
popSizes.alltrees20	741.7	popSizes.alltrees70	259.6
popSizes.alltrees21	2006.2	popSizes.alltrees71	591.8
popSizes.alltrees22	1049.4	popSizes.alltrees72	675.3
popSizes.alltrees23	852	popSizes.alltrees73	468.2
popSizes.alltrees24	789.5	popSizes.alltrees74	58.7
popSizes.alltrees25	963.6	popSizes.alltrees75	788.4
popSizes.alltrees26	685.9	popSizes.alltrees76	677.4
popSizes.alltrees27	932.4	popSizes.alltrees77	393.7
popSizes.alltrees28	808.9	popSizes.alltrees78	192.8
popSizes.alltrees29	688.6	popSizes.alltrees79	84.5
popSizes.alltrees30	746.1	popSizes.alltrees80	542.6
popSizes.alltrees31	535.4	popSizes.alltrees81	112.8
popSizes.alltrees32	768.7	popSizes.alltrees82	524.3
popSizes.alltrees33	1695.2	popSizes.alltrees83	299.1
popSizes.alltrees34	353	popSizes.alltrees84	408.9
popSizes.alltrees35	557.5	popSizes.alltrees85	358.6
popSizes.alltrees36	637.4	popSizes.alltrees86	263.4
popSizes.alltrees37	252.8	popSizes.alltrees87	233.8
popSizes.alltrees38	769.3	popSizes.alltrees88	328.7
popSizes.alltrees39	672.7	popSizes.alltrees89	120.2
popSizes.alltrees40	522.9	popSizes.alltrees90	254.3
popSizes.alltrees41	319.4	popSizes.alltrees91	706.6
popSizes.alltrees42	382.8	popSizes.alltrees92	323.1
popSizes.alltrees43	369.5	popSizes.alltrees93	282
popSizes.alltrees44	481.9	popSizes.alltrees94	415.8
popSizes.alltrees45	105.3	popSizes.alltrees95	415
popSizes.alltrees46	740.2	popSizes.alltrees96	140.4
popSizes.alltrees47	54.3	popSizes.alltrees97	279
popSizes.alltrees48	185.1	popSizes.alltrees98	559.2
popSizes.alltrees49	54	popSizes.alltrees99	824.9
popSizes.alltrees50	543.3	popSizes.alltrees100	171.5

Table C.3: ESSs for the first 100 popSizes.alltrees parameters.

Parameter	ESS
popSizes.alltrees101	467.5
popSizes.alltrees102	400
popSizes.alltrees103	319.9
popSizes.alltrees104	81.3
popSizes.alltrees105	341.6
popSizes.alltrees106	157.7
popSizes.alltrees107	608.8
popSizes.alltrees108	438.2
popSizes.alltrees109	69.7
popSizes.alltrees110	68.5
popSizes.alltrees111	323.2
popSizes.alltrees112	211.2
popSizes.alltrees113	40.9
popSizes.alltrees114	131.4
popSizes.alltrees115	90.5
popSizes.alltrees116	86.2
popSizes.alltrees117	2832.3
popSizes.alltrees118	2832.3
popSizes.alltrees119	67.7
popSizes.alltrees120	59.8
popSizes.alltrees121	1544
popSizes.alltrees122	2314.2
popSizes.alltrees123	56.7
popSizes.alltrees124	2156.7
popSizes.alltrees125	319.6
popSizes.alltrees126	1943.5
popSizes.alltrees127	115.9
popSizes.alltrees128	2832.3
popSizes.alltrees129	1482
popSizes.alltrees130	70.9
popSizes.alltrees131	114.2

Parameter	ESS
posterior	1939.8
likelihood	10187.7
prior	1925.1
TreeHeight	10775.9
kappa	12244.8
gammaShape	12581.2
ExtendedBayesianSkyline	343.3
populationMean.alltrees	2832.3
sum(indicators.alltrees)	442.3
ucldStdev	9589.3
rate.mean	11364.7
rate.variance	9345.4
rate.coefficientOfVariation	9568.4

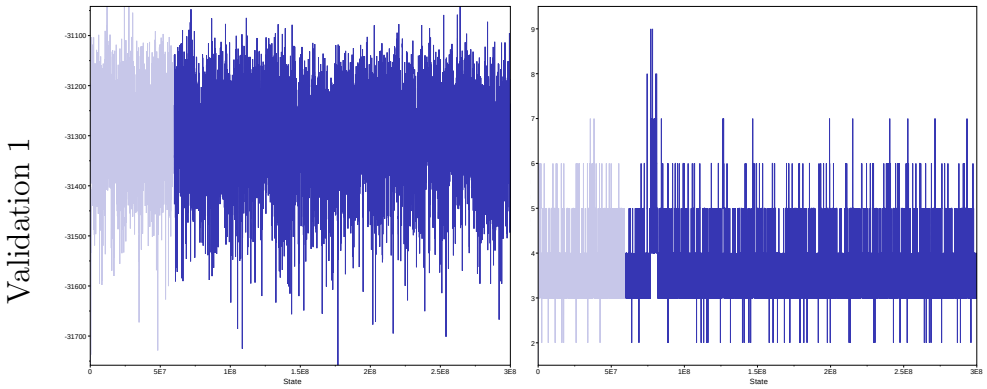
Table C.4: ESSs of remaining popSizes.alltrees parameters, as well as the other, more general, parameters.

C.1.3 EBSPs from BEAST validation runs

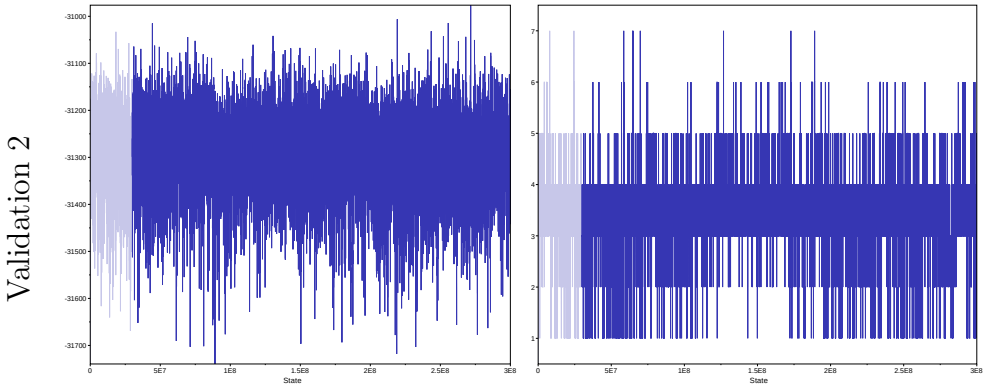
In this section, we present the trace plots for the validation runs of the Extended Bayesian Skyline analysis. The trace plots are for the posterior and the parameter'sum(indicators.alltrees)'. Recall that 'posterior' is the log posterior probability, and sum(indicators.alltrees) is the number of changepoints for the trajectory of the effective population size. Figure C.2 displays all trace plots as well as the corresponding effective sample sizes (ESSs). All trace plots seem reasonable, although there is a small deviation in the trace plot for sum(indicators.alltrees) in Validation 1. All ESSs are above 200.

The kernel density estimate for the posterior across all validation runs is given in Figure C.3, and the bar graph comparing the marginal distributions of the posterior samples of sum(indicators.alltrees) is given in Figure C.4.

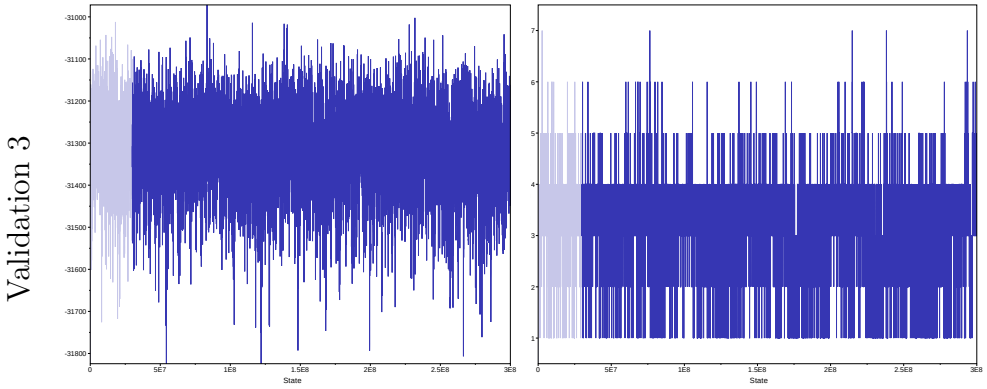
From Figure C.4 and the RHS of Figure C.2, we can see that each validation run has a slightly different distribution of posterior samples of the parameter sum(indicators.alltrees). All validation runs accept 3 major changes in the trajectory of effective population size more often than any other number. We notice that the posterior samples of sum(indicator.alltrees) are lower in the third validation run, because there is a greater posterior probability for 1 or 2 changes in effective population size compared to other runs. Conversely, the first validation run has greater posterior probabilities for sum(indicators.alltrees) ≥ 4 than both other validation runs. From Figure C.3, the marginal densities of the log posterior probability are similar for all validation runs. Despite the slightly different densities across the validation runs, all three produced similar Extended Bayesian Skyline Plots (EBSPs).



C.2.1: ESS = 2883 C.2.2: ESS = 371



C.2.3: ESS = 523 C.2.4: ESS = 2807



C.2.5: ESS = 1585 C.2.6: ESS = 139

Figure C.2: Trace plots from the three validation runs for the posterior (left) and sum(indicators.alltrees) (right) of the raw posterior samples. Each row is a different MCMC chain and the ESS for each parameter is given as a subcaption. The burn-in for each chain is indicated by a transparent region in each trace plot.

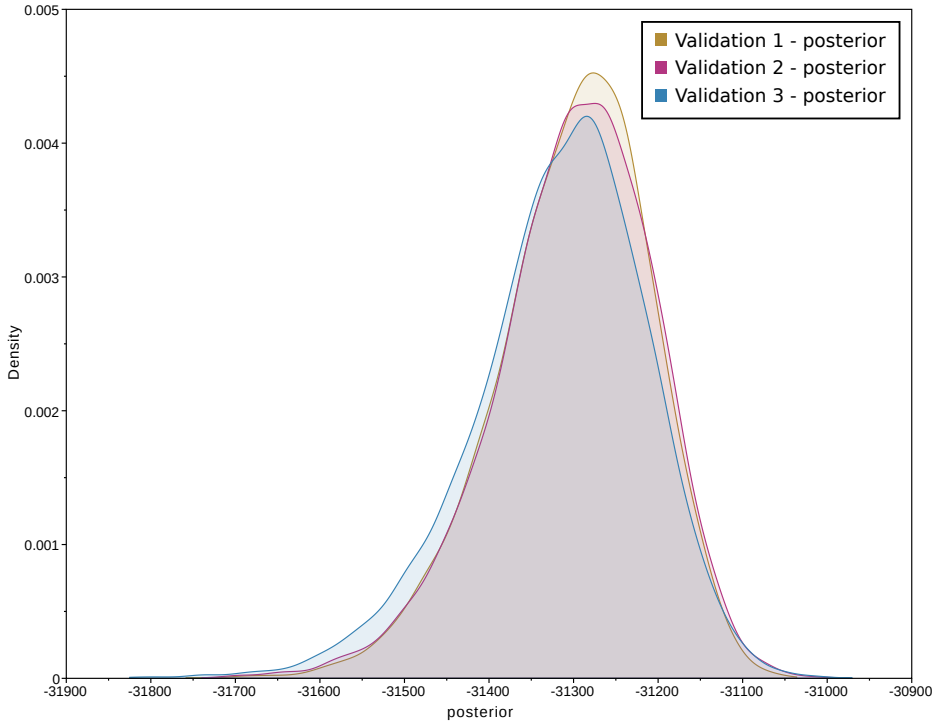


Figure C.3: Kernel density estimate of the posterior for all three validation chains.

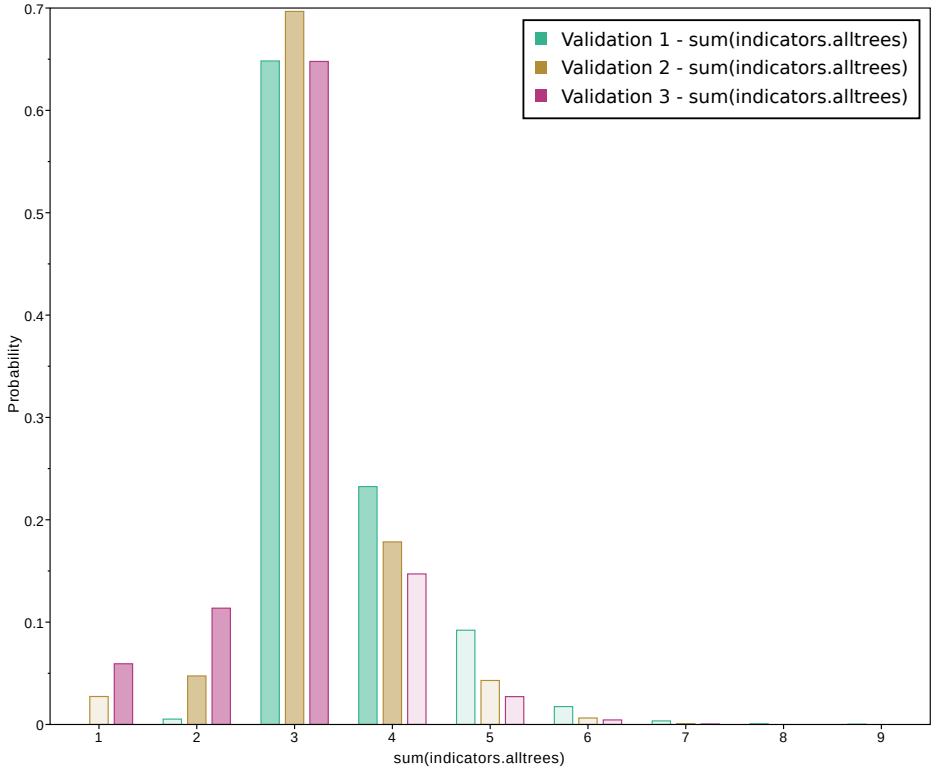


Figure C.4: Bar graph of the marginal distribution of sum(indicators.alltrees) for all three validation chains.

C.2 Extended Results from Chapter 6

C.2.1 UMAP dimension reduction of summary statistics

We present the UMAP dimension reductions of the observed and simulated summary statistics for `n_components` $= 2$, `min_dist` $= 0.1, 0.3, 0.5, 0.7, 0.9$ and `n_neighbours`$= 5, 10, 20, 50, 100$. The random seeds (`random_state` parameters) for each combination of minimum distance and number of nearest neighbours are given in Table C.5.

Minimum distance (`min_dist`)	Number of nearest neighbours (`nn`)				
	5	10	20	50	100
0.1	44193	65095	303	6259	56520
0.3	65587	71788	98174	30671	63216
0.5	82390	33110	34477	62206	71817
0.7	14215	78506	31	96043	29632
0.9	68184	92076	84761	47948	20461

Table C.5: Random states used to initialize the UMAP algorithm for all combinations of parameters.

Each value of minimum distance is given as a set of plots on a new page, with all possible values for nearest neighbours. The plots are consistent across all values for minimum distance and nearest neighbours, which suggests that we are not falsely identifying any clusters. We notice that there is more spread in the dimension reduction for smaller numbers of nearest neighbours, which is true for all minimum distances.

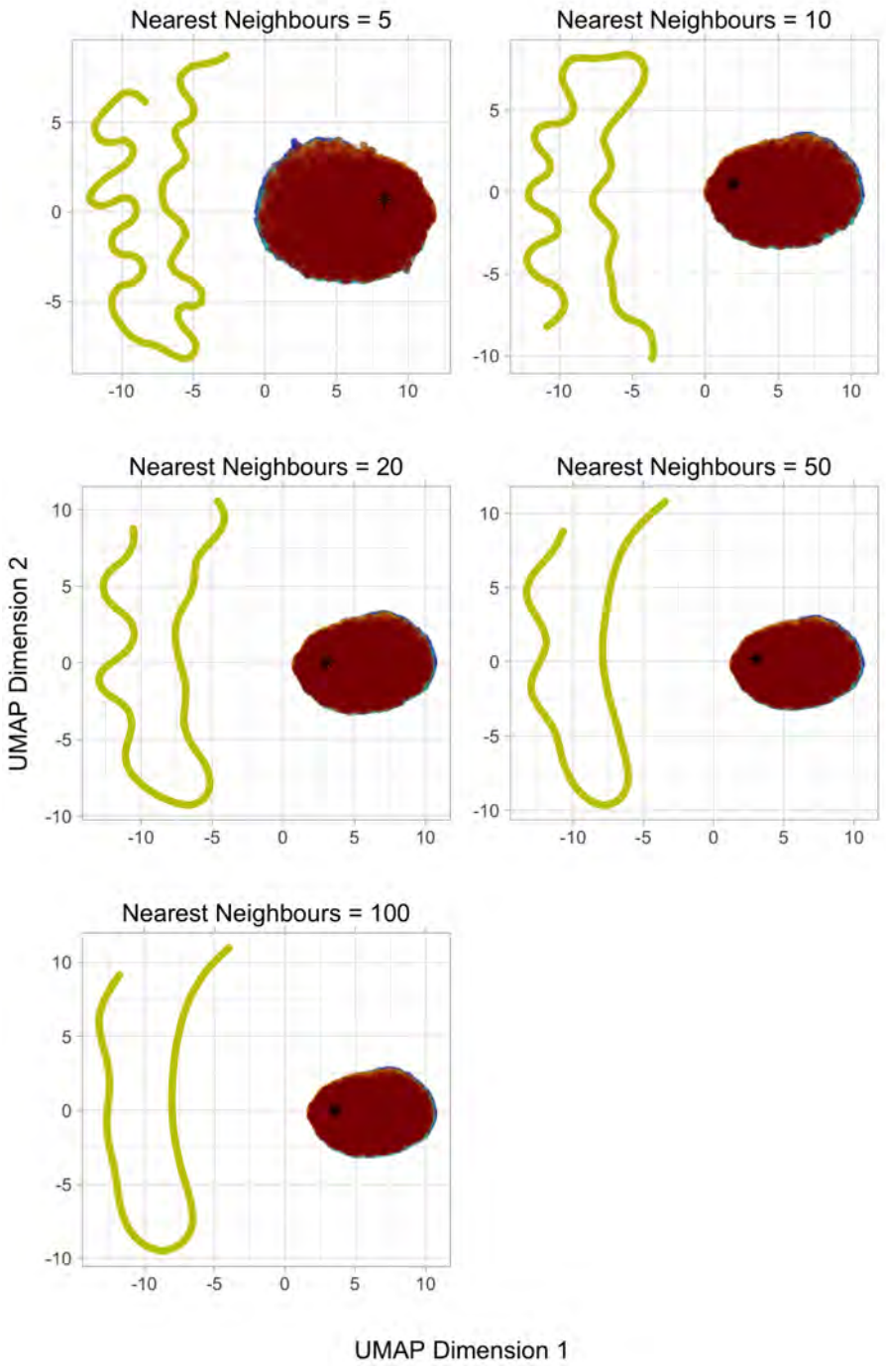


Figure C.5: UMAP dimension reductions with a minimum distance of 0.1. This means that only a small distance is required between points in the two-dimensional space.

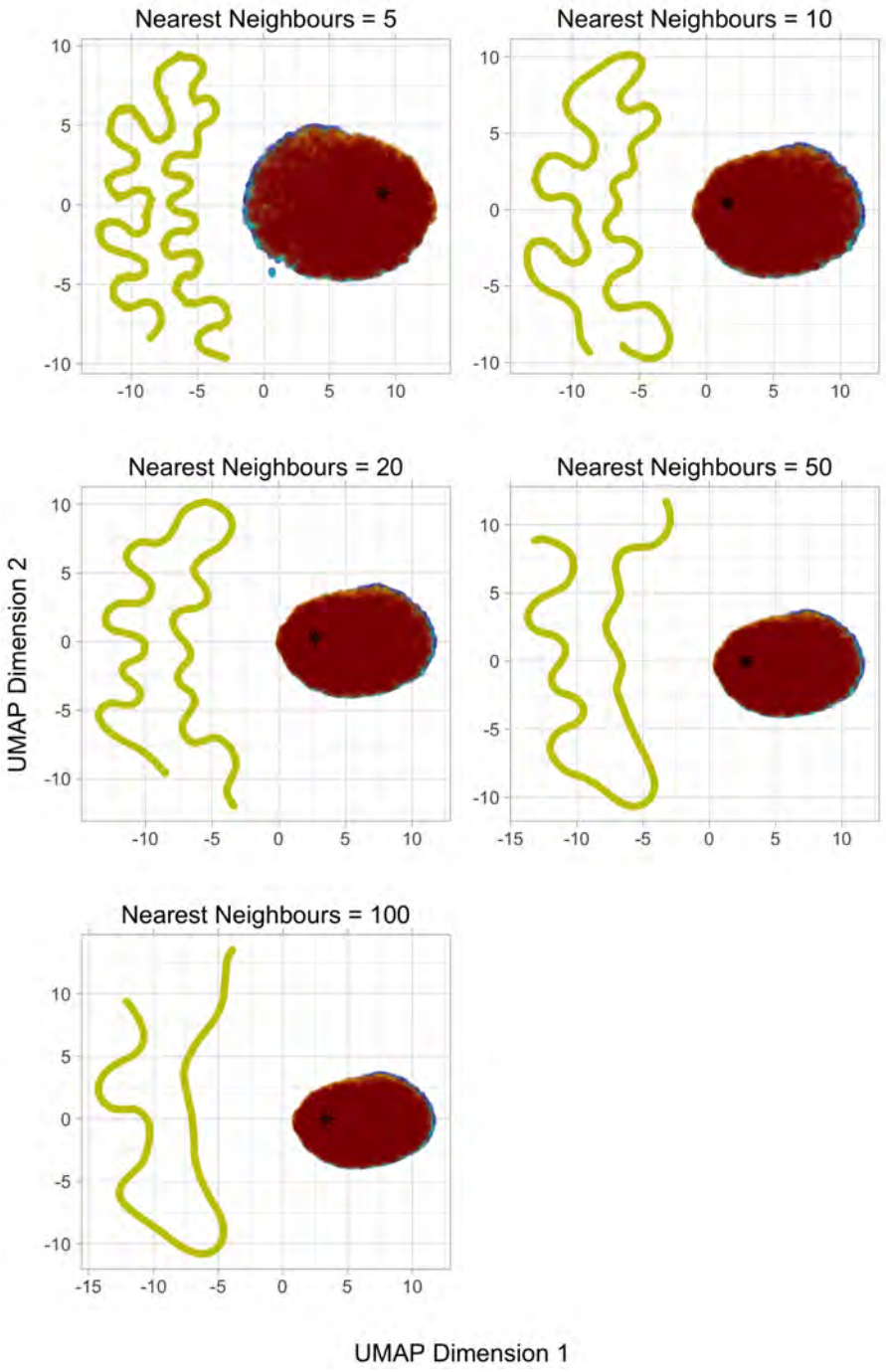


Figure C.6: UMAP dimension reductions with a minimum distance of 0.3.

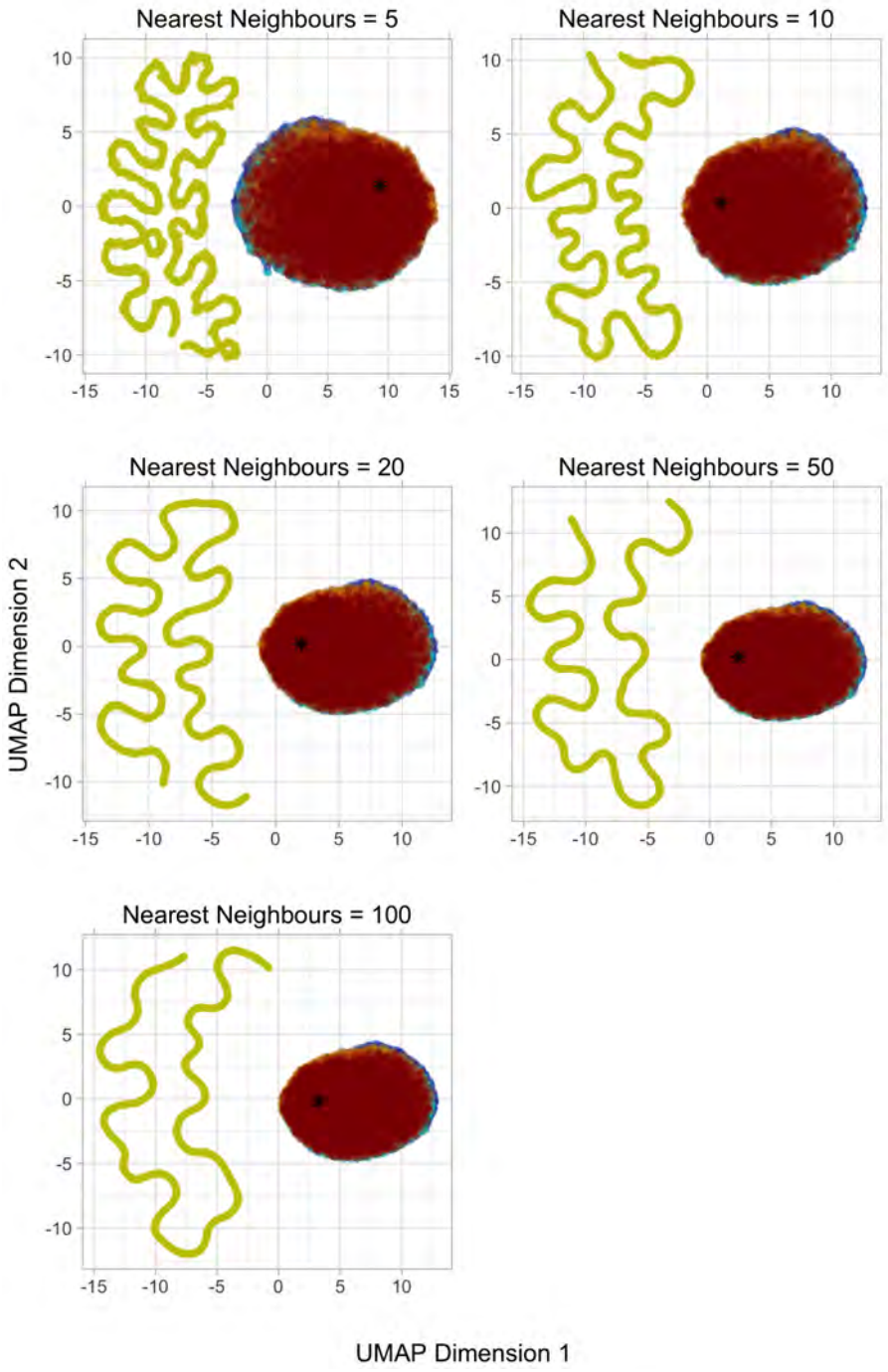


Figure C.7: UMAP dimension reductions with a minimum distance of 0.5.

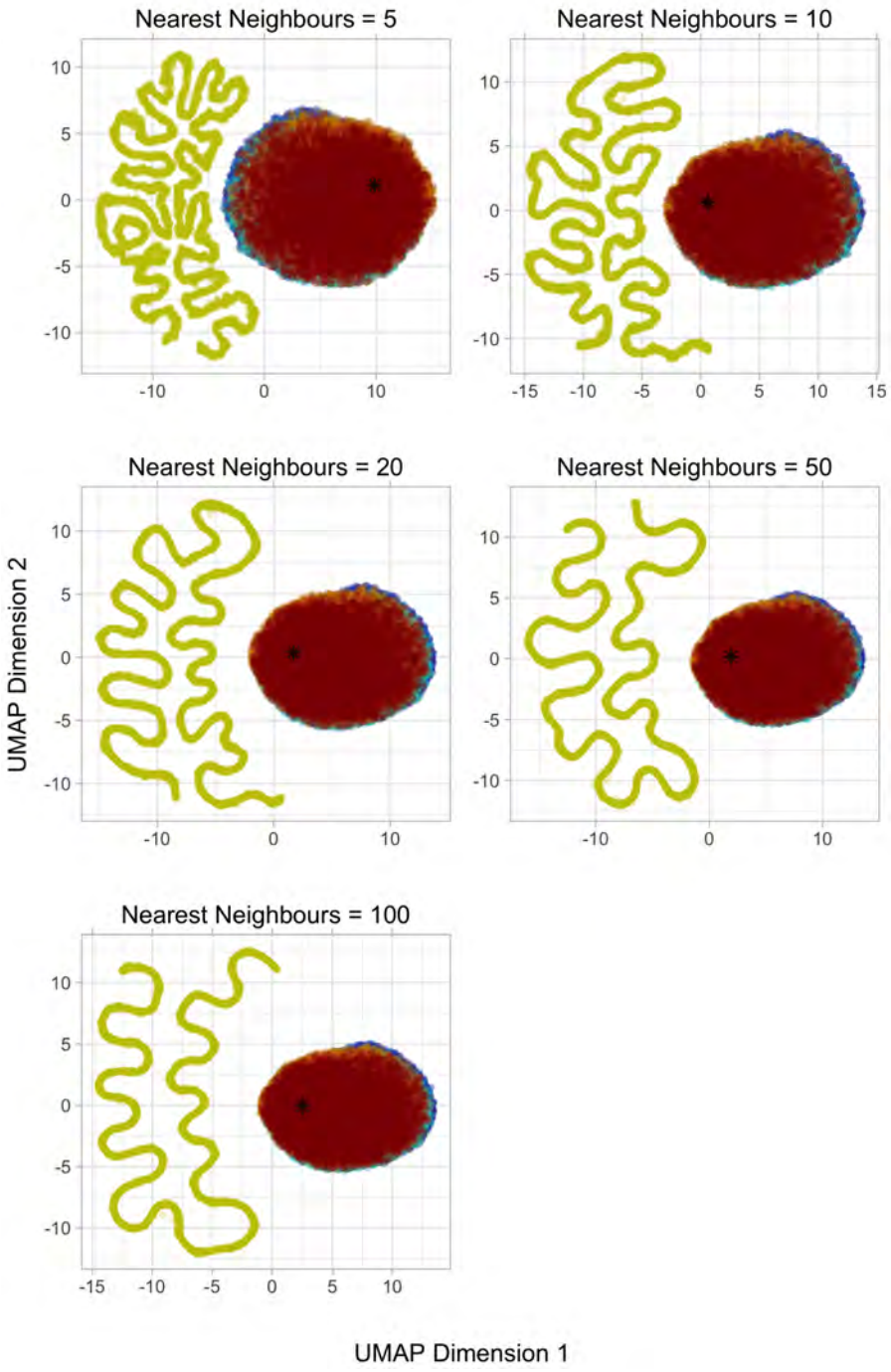


Figure C.8: UMAP dimension reductions with a minimum distance of 0.7.

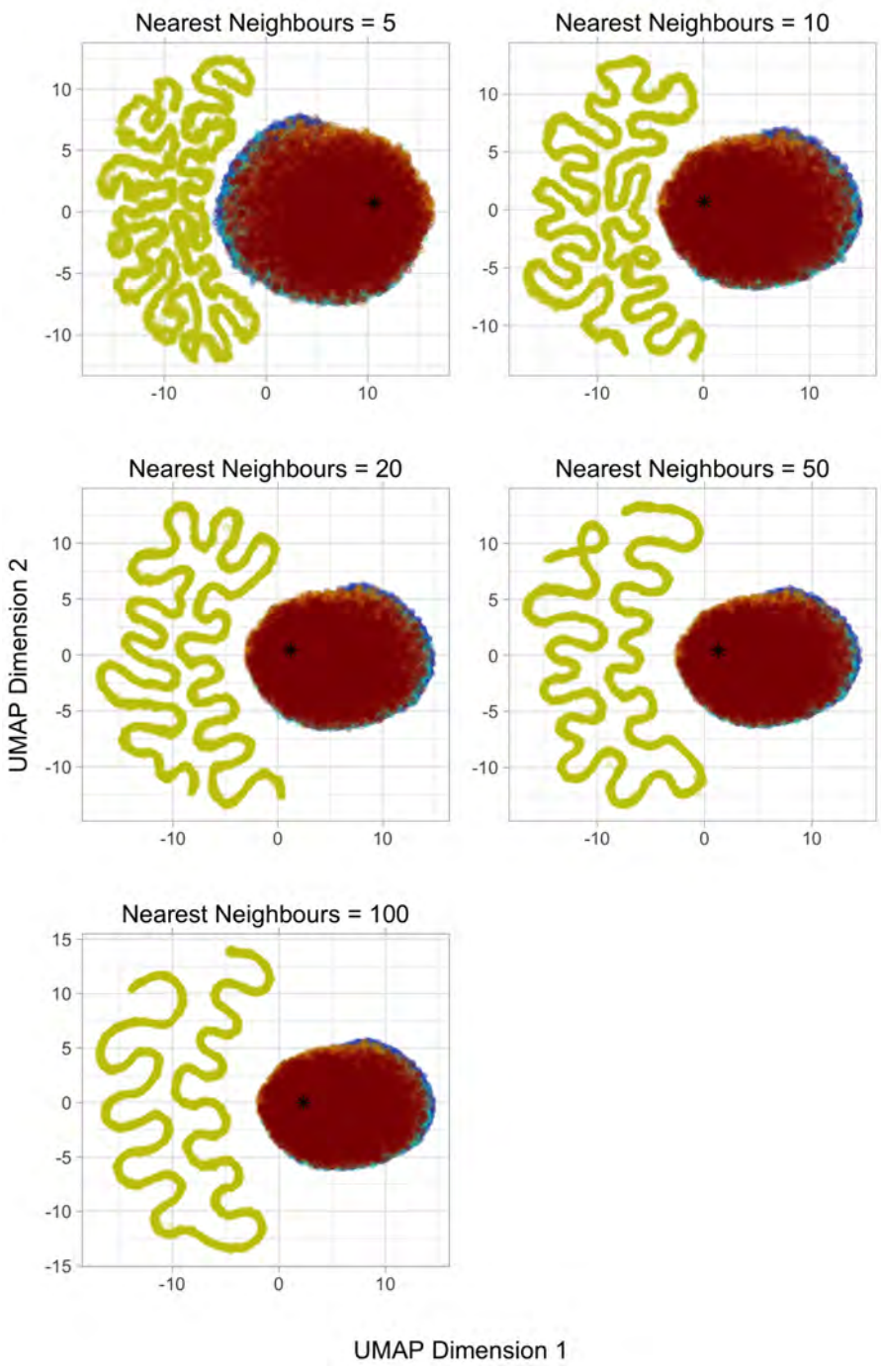


Figure C.9: UMAP dimension reductions with a minimum distance of 0.9. This means that a comparatively large distance is required between points in the two-dimensional space.

C.2.2 Predictor variables for MLR with forward selection

The predictor variables and estimated coefficients from MLR with forward selection are given in Tables C.2 and C.3. Recall that MLR compares two classes in each regression model, and so there are no coefficients corresponding to the Birdsell 1 migration model because it is the reference class.

The names of the predictor variables correspond to the summary statistics in the BayeSSC output. 'MeanDiv.Hs.bar' is the name given to mean haplotype diversity, while 'PoolDiv.Ht' is the name given to pooled haplotype diversity. Since there are multiple populations, there are more than one of each of the summary statistics. Every time that a summary statistic is repeated, a number is incremented at the end of the statistic name. For example, the first time the number of segregated sites appears, it is given as 'SegSites'. The second time this statistic appears, it is given as 'SegSites.1'.

For within-population summary statistics, this naming convention means that the population ID is appended to within-population summary statistics (excluding pairwise differences). Within-population summary statistics that do not have a number appended are for Population 0, *i.e.* Borneo.

The numbers appended to the names of the summary statistics are not as easily interpretable for between-population summary statistics or for the average number of pairwise differences. All numbers appended to the names of between-population summary statistics are explained in Table C.6.1. Suppose that we wish to find which populations were considered in the calculation of 'Fst.10'. Using Table C.6.1, we can see that 'Fst.10' is F_{ST} calculated from the DNA simulated for Populations 2 and 4.

There are a larger number of pairwise differences statistics because it is both a within-population and between-population statistic. The numbers appended to the average number of pairwise differences are explained in Table C.6.2. Giving another two examples, we can see from Table C.6.2 that 'PairDiffs.18' is the average number of pairwise differences calculated from Population 4, while 'PairDiffs.13' would be the average number of pairwise differences between populations 2 and 4.

Number appended	Population IDs used for statistic
(blank)	0, 1
1	0, 2
2	0, 3
3	0, 4
4	0, 5
5	1, 2
6	1, 3
7	1, 4
8	1, 5
9	2, 3
10	2, 4
11	2, 5
12	3, 4
13	3, 5
14	4, 5

C.6.1: Numbering convention for between-population summary statistics, excluding pairwise differences.

Number appended	Population IDs used for statistic
(blank)	0
1	0, 1
2	0, 2
3	0, 3
4	0, 4
5	0, 5
6	1
7	1, 2
8	1, 3
9	1, 4
10	1, 5
11	2
12	2, 3
13	2, 4
14	2, 5
15	3
16	3, 4
17	3, 5
18	4
19	4, 5
20	5

C.6.2: Numbering convention for pairwise differences.

Predictor Variable	Birdsell 2	Bowdler 1	Bowdler 2	Northern Sunda
(Intercept)	-29.40	0.03	-0.09	-2.69
Fst	0.70	-0.41	-0.57	-0.59
Fst.1	0.89	-0.15	-0.25	0.10
Fst.10	0.39	0.52	0.76	1.59
Fst.11	0.50	0.42	0.55	0.91
Fst.12	-0.59	-3.79	-3.41	-3.32
Fst.13	-0.06	0.39	-0.20	1.08
Fst.14	-0.02	-1.95	-2.39	-1.94
Fst.2	0.02	0.09	0.12	1.14
Fst.3	-0.50	1.03	1.00	0.67
Fst.4	-0.58	0.37	0.62	-0.01
Fst.5	-6.79	-0.13	-0.49	0.47
Fst.6	0.76	0.43	0.27	0.91
Fst.7	0.85	1.16	1.25	1.01
Fst.8	1.18	0.03	0.49	-0.18
Fst.9	0.82	0.29	0.25	-5.12
MeanDiv.Hs.bar..5	-146.00	27.00	18.90	35.70
PairDiffs	0.76	1.73	1.58	2.29
PairDiffs.1	-0.58	0.20	0.28	0.17
PairDiffs.10	-0.45	0.03	-0.14	0.12
PairDiffs.11	-9.99	2.80	2.47	-5.17
PairDiffs.12	-0.34	-0.07	-0.06	1.36
PairDiffs.13	-0.17	-0.13	-0.21	-0.52
PairDiffs.14	-0.24	-0.20	-0.16	-0.40
PairDiffs.15	-2.27	-6.50	-7.76	-11.40
PairDiffs.16	0.18	0.82	0.82	0.62
PairDiffs.17	-0.04	-0.14	-0.02	-0.41
PairDiffs.18	1.63	-7.49	-4.37	-5.18
PairDiffs.19	0.00	0.75	0.78	0.79
PairDiffs.2	-0.28	0.03	0.05	-0.06
PairDiffs.20	1.79	-3.23	-3.86	-2.52

Table C.7: Estimated coefficients for each pairwise logistic regression that comprises MLR. Birdsell 1 is the reference class, and forward selection was used for variable selection.

Predictor Variable	Birdsell 2	Bowdler 1	Bowdler 2	Northern Sunda
PairDiffs.3	0.02	-0.03	-0.01	-0.37
PairDiffs.4	0.21	-0.33	-0.31	-0.22
PairDiffs.5	0.27	-0.16	-0.22	0.07
PairDiffs.6	-4.00	0.44	0.72	0.70
PairDiffs.7	1.99	0.01	0.14	-0.16
PairDiffs.8	-0.19	-0.10	-0.03	-0.27
PairDiffs.9	-0.27	-0.43	-0.47	-0.38
PoolDiv.Ht..13	0.14	0.38	0.33	0.49
PrivHaps.2	30.70	-16.20	-15.10	7.41
PrivHaps.5	-20.30	40.00	38.80	38.70
PrivTo2.1	-126.00	65.40	61.30	-31.00
PrivTo3	-111.00	17.50	17.20	123.00
PrivTo3.2	2.20	10.60	10.60	-20.10
PrivTo3.3	157.00	-84.60	-84.20	-82.20
PrivTo4.3	0.20	-2.10	-2.03	-2.20
PrivTo5.3	5.13	-23.90	-21.40	-21.10
PrivTo5.4	18.00	-29.70	-29.60	-29.70
SegSites.1	1.65	0.17	0.00	0.25
SegSites.2	3.12	-0.02	-0.03	2.18
SegSites.3	0.38	2.14	2.03	3.22
SegSites.4	-0.26	1.04	0.46	0.86
SegSites.5	0.07	1.40	1.30	1.23
TajimasD.1	0.17	0.07	0.03	0.12
TajimasD.3	0.49	-0.03	0.06	-0.10
TajimasD.4	-0.04	0.08	-0.09	0.02

Table C.7: Estimated coefficients for each pairwise logistic regression that comprises MLR. Birdsell 1 is the reference class, and forward selection was used for variable selection.

Predictor Variable	Aggregated	Southern Sunda	Tindale 1	Tindale 2
(Intercept)	-5.55	-43.90	-12.70	-1.94
Fst	-1.37	0.09	-0.85	-0.48
Fst.1	-0.02	1.34	0.05	-0.04
Fst.10	0.84	2.14	1.58	1.73
Fst.11	-0.52	1.30	1.32	0.99
Fst.12	-1.47	-3.71	-3.05	-3.55
Fst.13	0.94	0.72	0.19	1.22
Fst.14	-2.32	-2.07	-2.26	-1.87
Fst.2	1.27	0.70	0.88	1.06
Fst.3	1.59	0.30	0.93	0.82
Fst.4	-0.31	-0.19	0.34	-0.19
Fst.5	-2.73	-7.15	0.18	0.47
Fst.6	2.11	2.29	1.17	0.92
Fst.7	2.30	1.51	1.00	0.76
Fst.8	0.85	0.78	0.13	-0.27
Fst.9	-4.01	-3.44	-5.06	-5.10
MeanDiv.Hs.bar..5	45.60	-159.00	74.00	46.60
PairDiffs	2.09	3.93	2.30	2.03
PairDiffs.1	0.32	-0.16	0.30	0.10
PairDiffs.10	-0.51	-0.33	0.03	0.12
PairDiffs.11	-13.60	-11.90	-4.61	-5.00
PairDiffs.12	0.92	0.67	1.34	1.34
PairDiffs.13	-0.02	-0.80	-0.51	-0.57
PairDiffs.14	0.55	-0.52	-0.49	-0.48
PairDiffs.15	5.66	-14.20	-10.20	-12.00
PairDiffs.16	-0.54	0.79	0.69	0.72
PairDiffs.17	-0.99	-0.34	-0.17	-0.40
PairDiffs.18	-3.53	-7.78	-6.87	-7.43
PairDiffs.19	0.54	0.86	0.70	0.73
PairDiffs.2	-1.08	-0.57	-0.06	-0.00
PairDiffs.20	-0.82	-1.44	-3.00	-2.56

Table C.8: Estimated coefficients for each pairwise logistic regression that comprises MLR. Birdsell 1 is the reference class, and forward selection was used for variable selection.

Predictor Variable	Aggregated	Southern Sunda	Tindale 1	Tindale 2
PairDiffs.3	-0.77	-0.25	-0.25	-0.34
PairDiffs.4	-0.39	-0.09	-0.33	-0.29
PairDiffs.5	-0.40	0.18	-0.00	0.15
PairDiffs.6	-9.34	-3.69	1.01	1.20
PairDiffs.7	0.34	2.12	-0.08	-0.16
PairDiffs.8	0.02	-0.62	-0.38	-0.26
PairDiffs.9	-1.03	-0.52	-0.32	-0.22
PoolDiv.Ht..13	0.23	0.36	0.54	0.33
PrivHaps.2	1.25	30.40	-9.25	5.90
PrivHaps.5	2.70	38.80	38.30	38.90
PrivTo2.1	-6.05	-125.00	36.30	-25.00
PrivTo3	1.01	121.00	125.00	124.00
PrivTo3.2	-0.27	-19.70	-20.30	-20.20
PrivTo3.3	-4.46	-81.30	-83.40	-82.90
PrivTo4.3	-0.34	-2.17	-1.66	-1.93
PrivTo5.3	-4.33	-21.40	-21.30	-21.80
PrivTo5.4	-0.09	-29.60	-29.20	-29.60
SegSites.1	4.93	1.80	0.14	-0.05
SegSites.2	1.98	3.96	2.22	2.15
SegSites.3	3.50	3.99	2.70	3.27
SegSites.4	2.03	1.40	1.14	1.23
SegSites.5	-0.42	1.28	1.22	1.31
TajimasD.1	1.65	0.24	0.07	0.00
TajimasD.3	-0.19	0.74	-0.41	0.00
TajimasD.4	0.32	0.23	0.16	0.17

Table C.8: Estimated coefficients for each pairwise logistic regression that comprises MLR. Birdsell 1 is the reference class, and forward selection was used for variable selection.

C.2.3 Predictor variables for MLR with LASSO

LASSO allows different predictor variables in each of the logistic regression models that comprise the multinomial logistic regression. The estimated coefficients for each of the predictor variables are displayed in Tables C.9, C.10, and C.11. Each predictor variable appears in at least one regression model. Table C.9 contains the estimated coefficients for the Birdsell 1, Birdsell 2, and Northern Sunda migration models; Table C.10 contains the estimated coefficients for the Bowdler 1, Bowdler 2, and Southern Sunda migration models; and Table C.11 contains the estimated coefficients for the Tindale 1, Tindale 2, and Aggregated migration models. All estimated coefficients are given to three significant figures. Predictor variables that do not appear in a particular regression model are denoted by a dash.

Predictor Variable	Birdsell 1	Birdsell 2	Northern Sunda
(Intercept)	0.649	0.0785	0.277
Haptypes	-	-	-
PrivHaps	-0.0139	-	-
SegSites	-0.234	-0.114	0.0904
PairDiffs	-	-	-
HapDiver	-	-	-
TajimasD	-0.178	-0.0484	0.0368
PrivTo0	-	-	-
PrivTo1	-	-	-
PairDiffs.1	0.048	-0.442	-
MeanDiv.Hs.bar.	-	-	-
PoolDiv.Ht.	-	-	-
Fst	0.208	0.819	-0.0888
PrivTo0.1	-	-	-
PrivTo2	-	-	-
PairDiffs.2	0.0498	-0.0659	-
MeanDiv.Hs.bar..1	-	-	-
PoolDiv.Ht..1	-	-	-
Fst.1	-0.07	0.466	0.00966
PrivTo0.2	-	-	-

Table C.9: Estimated coefficients for all regression models that make up MLR, with LASSO used for variable selection.

Predictor Variable	Birdsell 1	Birdsell 2	Northern Sunda
PrivTo3	-	-	-
PairDiffs.3	0.0146	0.00166	-0.0728
MeanDiv.Hs.bar..2	-	-	-
PoolDiv.Ht..2	-	-	-
Fst.2	-0.205	-0.118	0.322
PrivTo0.3	-	-	-
PrivTo4	-	-	-
PairDiffs.4	0.237	0.392	-0.00468
MeanDiv.Hs.bar..3	-	-	-
PoolDiv.Ht..3	-0.00283	0.0322	-
Fst.3	-0.74	-1.11	-0.000759
PrivTo0.4	-	-	-
PrivTo5	-	0.0467	-
PairDiffs.5	-	0.0966	-
MeanDiv.Hs.bar..4	-	-	-
PoolDiv.Ht..4	-	-	0.0524
Fst.4	-0.108	-0.376	-
Haptypes.1	-	-	-
PrivHaps.1	-0.0522	0.697	-
SegSites.1	-0.434	0.215	0.0521
PairDiffs.6	-	-1.97	-
HapDiver.1	-	-	-
TajimasD.1	-0.128	-0.306	0.0731
PrivTo1.1	-	-5.87	-
PrivTo2.1	-	-	-
PairDiffs.7	-	1.78	-0.0723
MeanDiv.Hs.bar..5	-	-	-
PoolDiv.Ht..5	-	-	-
Fst.5	-	-6.37	0.288
PrivTo1.2	-	0.0147	-
PrivTo3.1	-	-	-

Table C.9: Estimated coefficients for all regression models that make up MLR, with LASSO used for variable selection.

Predictor Variable	Birdsell 1	Birdsell 2	Northern Sunda
PairDiffs.8	0.103	0.00375	-0.00451
MeanDiv.Hs.bar..6	-	-	-
PoolDiv.Ht..6	-	-	-
Fst.6	-0.576	-	-
PrivTo1.3	-	-	-
PrivTo4.1	0.0481	-	-
PairDiffs.9	0.308	8.14e-05	-0.0179
MeanDiv.Hs.bar..7	-	-	-
PoolDiv.Ht..7	-	0.231	-0.0414
Fst.7	-0.906	-	-0.0155
PrivTo1.4	-	-	-
PrivTo5.1	-	-	-
PairDiffs.10	0.0438	-0.172	0.0638
MeanDiv.Hs.bar..8	-	-	-
PoolDiv.Ht..8	-	-	-
Fst.8	-0.259	0.453	-0.265
Haptypes.2	-	-	-
PrivHaps.2	0.0616	-0.197	-0.00346
SegSites.2	-1.58	0.336	-
PairDiffs.11	-	-	-
HapDiver.2	-	-	-
TajimasD.2	0.47	-0.67	-0.0711
PrivTo2.2	0.877	0.472	-
PrivTo3.2	1.4	2.54	-1.54
PairDiffs.12	-0.0362	-0.432	0.833
MeanDiv.Hs.bar..9	-	-	-
PoolDiv.Ht..9	0.0821	-	-
Fst.9	0.684	1.62	-3.34
PrivTo2.3	-	-	-
PrivTo4.2	-	-	-
PairDiffs.13	0.237	0.125	-0.00522

Table C.9: Estimated coefficients for all regression models that make up MLR, with LASSO used for variable selection.

Predictor Variable	Birdsell 1	Birdsell 2	Northern Sunda
MeanDiv.Hs.bar..10	-	-	-
PoolDiv.Ht..10	-	-0.122	-
Fst.10	-0.998	-0.726	-
PrivTo2.4	-	-	-
PrivTo5.2	-	-	-
PairDiffs.14	0.279	0.0107	-0.0369
MeanDiv.Hs.bar..11	-	-	-
PoolDiv.Ht..11	-	-0.0395	-
Fst.11	-0.746	-0.17	-
Haptypes.3	-	-	-
PrivHaps.3	-1.04	-2.18	1.46
SegSites.3	-1.29	-1.18	0.403
PairDiffs.15	-	-	-
HapDiver.3	-	-	-
TajimasD.3	0.753	0.977	-0.717
PrivTo3.3	-	-	-
PrivTo4.3	-	-	-
PairDiffs.16	-0.439	-0.262	-0.00368
MeanDiv.Hs.bar..12	-	-	-
PoolDiv.Ht..12	0.0238	0.00328	-
Fst.12	2.85	2.24	-
PrivTo3.4	-	-	-
PrivTo5.3	-	-	-
PairDiffs.17	0.0764	0.0388	-0.0976
MeanDiv.Hs.bar..13	-	-	-
PoolDiv.Ht..13	-0.287	-	0.0415
Fst.13	-0.157	-0.196	0.418
Haptypes.4	-	-	-
PrivHaps.4	-	-	-
SegSites.4	0.05	0.12	-
PairDiffs.18	-	-	-

Table C.9: Estimated coefficients for all regression models that make up MLR, with LASSO used for variable selection.

Predictor Variable	Birdsell 1	Birdsell 2	Northern Sunda
HapDiver.4	-	-	-
TajimasD.4	0.206	0.259	-0.0328
PrivTo4.4	-	-	-
PrivTo5.4	0.396	0.219	-
PairDiffs.19	-0.576	-0.58	0.0328
MeanDiv.Hs.bar..14	-	-	-
PoolDiv.Ht..14	0.0316	0.0585	-
Fst.14	1.72	1.7	0.0422
Haptypes.5	-	-	-
PrivHaps.5	-	-	-
SegSites.5	-0.716	-0.363	0.0728
PairDiffs.20	-	-	-
HapDiver.5	-	-	-
TajimasD.5	0.268	0.454	-

Table C.9: Estimated coefficients for all regression models that make up MLR, with LASSO used for variable selection.

Predictor Variable	Bowdler 1	Bowdler 2	Southern Sunda
(Intercept)	1.26	1.37	-0.255
Haptypes	-	-	-
PrivHaps	-0.00484	-	0.0161
SegSites	-	-	0.387
PairDiffs	-	-	-
HapDiver	-	-	-
TajimasD	-0.0173	-	0.207
PrivTo0	-	-	-
PrivTo1	-	-	-
PairDiffs.1	0.0983	0.104	-0.193
MeanDiv.Hs.bar.	-	-	-
PoolDiv.Ht.	-	-	-

Table C.10: Estimated coefficients for all regression models that make up MLR, with LASSO used for variable selection.

Predictor Variable	Bowdler 1	Bowdler 2	Southern Sunda
Fst	-	-0.0575	0.416
PrivTo0.1	-	-	-
PrivTo2	-	-	-
PairDiffs.2	0.0433	0.0852	-0.381
MeanDiv.Hs.bar..1	-	-	-
PoolDiv.Ht..1	-	-	-
Fst.1	-0.131	-0.276	0.993
PrivTo0.2	-	-	-
PrivTo3	-	-	-
PairDiffs.3	-	0.0198	-0.00104
MeanDiv.Hs.bar..2	-	-	-
PoolDiv.Ht..2	-	-	-
Fst.2	-0.159	-0.132	-
PrivTo0.3	-	-	-
PrivTo4	-	-	-
PairDiffs.4	-	-	0.021
MeanDiv.Hs.bar..3	-	-	-
PoolDiv.Ht..3	-	-	-
Fst.3	0.105	0.115	-0.194
PrivTo0.4	-	-	-
PrivTo5	-	-	-
PairDiffs.5	-0.0468	-0.104	0.0121
MeanDiv.Hs.bar..4	-	-	-
PoolDiv.Ht..4	-	-	-
Fst.4	0.0555	0.309	-
Haptypes.1	-	-	-
PrivHaps.1	-0.0335	-0.0378	0.84
SegSites.1	-0.0709	-0.136	0.489
PairDiffs.6	-	-	-1.87
HapDiver.1	-	-	-
TajimasD.1	-	-0.000558	-0.201

Table C.10: Estimated coefficients for all regression models that make up MLR, with LASSO used for variable selection.

Predictor Variable	Bowdler 1	Bowdler 2	Southern Sunda
PrivTo1.1	-	-	-6.06
PrivTo2.1	-	-	-
PairDiffs.7	-	0.11	1.81
MeanDiv.Hs.bar..5	-	-	-
PoolDiv.Ht..5	-	-	-0.0165
Fst.5	-0.11	-0.42	-6.58
PrivTo1.2	-	-	-
PrivTo3.1	-	-	-
PairDiffs.8	0.034	0.0507	-0.196
MeanDiv.Hs.bar..6	-	-	-
PoolDiv.Ht..6	-	-	-
Fst.6	-0.227	-0.281	1.06
PrivTo1.3	-	-	-
PrivTo4.1	0.0387	-	-3.61e-05
PairDiffs.9	-	-0.00284	-0.131
MeanDiv.Hs.bar..7	-	-	-
PoolDiv.Ht..7	0.000398	-	-
Fst.7	-	0.0236	0.432
PrivTo1.4	-	-	-
PrivTo5.1	-	-	-
PairDiffs.10	0.0289	-0.0203	-0.247
MeanDiv.Hs.bar..8	-	-	-
PoolDiv.Ht..8	-	-	-
Fst.8	-0.145	0.0842	0.447
Haptypes.2	-	-	-
PrivHaps.2	0.204	0.13	-0.204
SegSites.2	-1.2	-1.26	1.03
PairDiffs.11	-	-	-
HapDiver.2	-	-	-
TajimasD.2	0.863	0.816	-0.881
PrivTo2.2	-	0.453	-

Table C.10: Estimated coefficients for all regression models that make up MLR, with LASSO used for variable selection.

Predictor Variable	Bowdler 1	Bowdler 2	Southern Sunda
PrivTo3.2	2.86	2.74	-1.14
PairDiffs.12	-0.0971	-0.083	0.0172
MeanDiv.Hs.bar..9	-	-	-
PoolDiv.Ht..9	0.139	-	-
Fst.9	0.961	0.891	-1.47
PrivTo2.3	-	-	-
PrivTo4.2	-	-	-
PairDiffs.13	0.0577	0.00338	-0.112
MeanDiv.Hs.bar..10	-	-	-
PoolDiv.Ht..10	0.136	0.122	-0.0268
Fst.10	-0.402	-0.205	0.241
PrivTo2.4	-	-	-
PrivTo5.2	-	-	-
PairDiffs.14	0.0643	0.0788	-0.0734
MeanDiv.Hs.bar..11	-	-	-
PoolDiv.Ht..11	-	0.0502	-0.0173
Fst.11	-0.306	-0.131	0.25
Haptypes.3	-	-	-
PrivHaps.3	-2.76	-2.59	1.08
SegSites.3	-	-0.204	0.885
PairDiffs.15	-	-	-
HapDiver.3	-	-	-
TajimasD.3	1.89e-05	-	-0.195
PrivTo3.3	-	-	-
PrivTo4.3	-	-	-
PairDiffs.16	0.203	0.225	0.0376
MeanDiv.Hs.bar..12	-	-	-
PoolDiv.Ht..12	-0.145	-0.13	-
Fst.12	-0.506	-0.182	-0.157
PrivTo3.4	-	-	-
PrivTo5.3	-	-	-

Table C.10: Estimated coefficients for all regression models that make up MLR, with LASSO used for variable selection.

Predictor Variable	Bowdler 1	Bowdler 2	Southern Sunda
PairDiffs.17	0.0278	0.0843	-0.00239
MeanDiv.Hs.bar..13	-	-	-
PoolDiv.Ht..13	-	-	-
Fst.13	-	-0.446	-
Haptypes.4	-	-	-
PrivHaps.4	-	-	-
SegSites.4	-0.162	-0.211	0.103
PairDiffs.18	-	-	-
HapDiver.4	-	-	-
TajimasD.4	-0.0978	-0.0859	0.0208
PrivTo4.4	-	-	-
PrivTo5.4	-0.00702	-0.0148	-
PairDiffs.19	0.0187	0.0284	0.0577
MeanDiv.Hs.bar..14	-	-	-
PoolDiv.Ht..14	-0.0591	-0.0101	-
Fst.14	-	-0.416	-0.0285
Haptypes.5	-	-	-
PrivHaps.5	-	-	-
SegSites.5	0.127	-0.0864	0.226
PairDiffs.20	-	-	-
HapDiver.5	-	-	-
TajimasD.5	-0.104	-0.191	0.0751

Table C.10: Estimated coefficients for all regression models that make up MLR, with LASSO used for variable selection.

Predictor Variable	Tindale 1	Tindale 2	Aggregated
(Intercept)	0.497	0.275	-4.15
Haptypes	-	-	-
PrivHaps	0.000122	-0.00631	-
SegSites	0.088	0.0548	-
PairDiffs	-	-	-

Table C.11: Estimated coefficients for all regression models that make up MLR, with LASSO used for variable selection.

Predictor Variable	Tindale 1	Tindale 2	Aggregated
HapDiver	-	-	-
TajimasD	0.0368	-	-
PrivTo0	-	-	-
PrivTo1	-	-	-
PairDiffs.1	0.0873	-0.0273	-
MeanDiv.Hs.bar.	-	-	-
PoolDiv.Ht.	-	-	-
Fst	-0.289	-0.0331	-
PrivTo0.1	-	-	-
PrivTo2	-	-	-
PairDiffs.2	-0.0151	-	-
MeanDiv.Hs.bar..1	-	-	-
PoolDiv.Ht..1	-	-	-
Fst.1	-	-	-
PrivTo0.2	-	-	-
PrivTo3	-	-	-
PairDiffs.3	-	-0.0148	-
MeanDiv.Hs.bar..2	-	-	-
PoolDiv.Ht..2	0.0159	-	-0.00901
Fst.2	0.162	0.2	-
PrivTo0.3	-	-	-
PrivTo4	-	-	-
PairDiffs.4	-0.0523	-0.00406	-
MeanDiv.Hs.bar..3	-	-	-
PoolDiv.Ht..3	-	-	-
Fst.3	0.134	-	-
PrivTo0.4	-	-	-
PrivTo5	-	-	-
PairDiffs.5	-0.0343	0.03	-
MeanDiv.Hs.bar..4	-	-	-
PoolDiv.Ht..4	-	-	-
Fst.4	0.27	-0.076	-

Table C.11: Estimated coefficients for all regression models that make up MLR, with LASSO used for variable selection.

Predictor Variable	Tindale 1	Tindale 2	Aggregated
Haptypes.1	-	-	-
PrivHaps.1	0.0058	-0.00992	-
SegSites.1	-	-	-
PairDiffs.6	0.13	-	-
HapDiver.1	-	-	-
TajimasD.1	0.0387	0.0349	-
PrivTo1.1	-	-	-0.168
PrivTo2.1	-	-	-
PairDiffs.7	-0.0196	-0.074	-
MeanDiv.Hs.bar..5	-	-	-
PoolDiv.Ht..5	-	-	-
Fst.5	0.0647	0.277	-
PrivTo1.2	-	-	-
PrivTo3.1	-	-	-
PairDiffs.8	-0.0322	-	-
MeanDiv.Hs.bar..6	-	-	-
PoolDiv.Ht..6	-	-	-
Fst.6	0.0711	-	-
PrivTo1.3	-	-	-
PrivTo4.1	-0.000254	-0.0329	-
PairDiffs.9	-	0.0412	-
MeanDiv.Hs.bar..7	-	-	-
PoolDiv.Ht..7	-	-	-
Fst.7	0.0385	-0.0891	-
PrivTo1.4	-	-	-
PrivTo5.1	-	-	-
PairDiffs.10	-	0.0845	-
MeanDiv.Hs.bar..8	-	-	-
PoolDiv.Ht..8	-	-	-
Fst.8	-	-0.392	-
Haptypes.2	-	-	-
PrivHaps.2	-0.0015	-	-

Table C.11: Estimated coefficients for all regression models that make up MLR, with LASSO used for variable selection.

Predictor Variable	Tindale 1	Tindale 2	Aggregated
SegSites.2	0.104	-	-
PairDiffs.11	-	-	-
HapDiver.2	-	-	-
TajimasD.2	-	-0.0349	-
PrivTo2.2	-	-	-
PrivTo3.2	-1.71	-1.6	-
PairDiffs.12	0.842	0.798	-
MeanDiv.Hs.bar..9	-	-	-
PoolDiv.Ht..9	0.000408	-0.0717	-1.21
Fst.9	-3.33	-3.29	-
PrivTo2.3	-	-	-
PrivTo4.2	-	-	5.93
PairDiffs.13	-0.0535	-0.0274	-
MeanDiv.Hs.bar..10	-	-	-
PoolDiv.Ht..10	-0.0376	0.017	-
Fst.10	0.114	0.0978	-
PrivTo2.4	-	-	-
PrivTo5.2	-	-	0.719
PairDiffs.14	-0.0406	-0.0728	-
MeanDiv.Hs.bar..11	-	-	-
PoolDiv.Ht..11	-	-	-
Fst.11	0.238	-	-
Haptypes.3	-	-	-
PrivHaps.3	1.64	1.54	-
SegSites.3	0.0713	0.423	-
PairDiffs.15	-	-	-
HapDiver.3	-	-	-
TajimasD.3	-0.83	-0.65	-
PrivTo3.3	-	-	-
PrivTo4.3	-	-	-
PairDiffs.16	0.109	-	-
MeanDiv.Hs.bar..12	-	-	-

Table C.11: Estimated coefficients for all regression models that make up MLR, with LASSO used for variable selection.

Predictor Variable	Tindale 1	Tindale 2	Aggregated
PoolDiv.Ht..12	-	-	-
Fst.12	0.163	-0.0607	-
PrivTo3.4	-	-	-
PrivTo5.3	-	-	-
PairDiffs.17	-0.0194	-0.0814	-
MeanDiv.Hs.bar..13	-	-	-
PoolDiv.Ht..13	0.0472	-	-0.0355
Fst.13	-0.14	0.537	-
Haptypes.4	-	-	-
PrivHaps.4	-	-	-
SegSites.4	-0.0231	-	-
PairDiffs.18	-	-	-
HapDiver.4	-	-	-
TajimasD.4	-	-0.0153	-
PrivTo4.4	-	-	-
PrivTo5.4	-	-	-
PairDiffs.19	-0.0844	-	-
MeanDiv.Hs.bar..14	-	-	-
PoolDiv.Ht..14	0.0405	-	-
Fst.14	-0.244	0.0792	-
Haptypes.5	-	-	-
PrivHaps.5	-0.00941	-	-
SegSites.5	-	0.117	-
PairDiffs.20	-	-	-
HapDiver.5	-	-	-
TajimasD.5	-0.0479	-0.0309	-

Table C.11: Estimated coefficients for all regression models that make up MLR, with LASSO used for variable selection.

C.2.4 Detailed results from validation analyses

For MLR (with both types of variable selection) and linear SVMs, we used 10% of the data as training data. There were four different randomly-selected 10% validation sets that contained the same number of observations for each migration model. We used these four different validation sets to train all three classifiers.

Since neural networks rely on large amounts of data to be effective, we re-trained the neural network on the same 70% training data.

Validation for MLR with Forward Selection

For the first validation run, the classifier had a training accuracy of 51.4% and a test accuracy of 50.0%. For the second validation run, the classifier had a training accuracy of 51.4% and a test accuracy of 49.8%. For the third validation run, the classifier had a training accuracy of 51.1% and a test accuracy of 49.9%. For the fourth validation run, the classifier had a training accuracy of 51.8% and a test accuracy of 49.7%.

For all validation runs, the class probabilities for the observed summary statistics are given in Table C.12. We note that different validation runs choose different models (Validation 1, 3, and 4 select Southern Sunda, while Validation 2 selects Birdsell 1), and the highest class probability is often approximately one.

Confusion matrices for all validation runs are given below. Similar misclassification patterns appear in all confusion matrices, and these also resemble the confusion matrices from the initial analysis.

Migration Model	Probability (Validation 1)	Probability (Validation 2)	Probability (Validation 3)	Probability (Validation 4)
Birdsell 1	1.011×10^{-35}	≈ 1.00	3.145×10^{-25}	1.756×10^{-3}
Birdsell 2	2.078×10^{-40}	5.583×10^{-10}	6.050×10^{-24}	6.953×10^{-3}
Bowdler 1	3.444×10^{-36}	2.941×10^{-19}	1.439×10^{-31}	0.122
Bowdler 2	2.210×10^{-35}	4.060×10^{-19}	2.970×10^{-31}	9.385×10^{-2}
Northern Sunda	5.496×10^{-19}	6.126×10^{-19}	5.303×10^{-19}	0.139
Southern Sunda	≈ 1.000	5.516×10^{-18}	≈ 1.00	0.397
Tindale 1	1.358×10^{-18}	3.666×10^{-18}	3.760×10^{-22}	0.133
Tindale 2	7.854×10^{-18}	1.019×10^{-18}	1.211×10^{-21}	0.107
Aggregated	4.721×10^{-47}	4.230×10^{-33}	7.553×10^{-48}	3.596×10^{-36}

Table C.12: Class probabilities of the observed summary statistics belonging to each migration model, determined by MLR with forward selection.

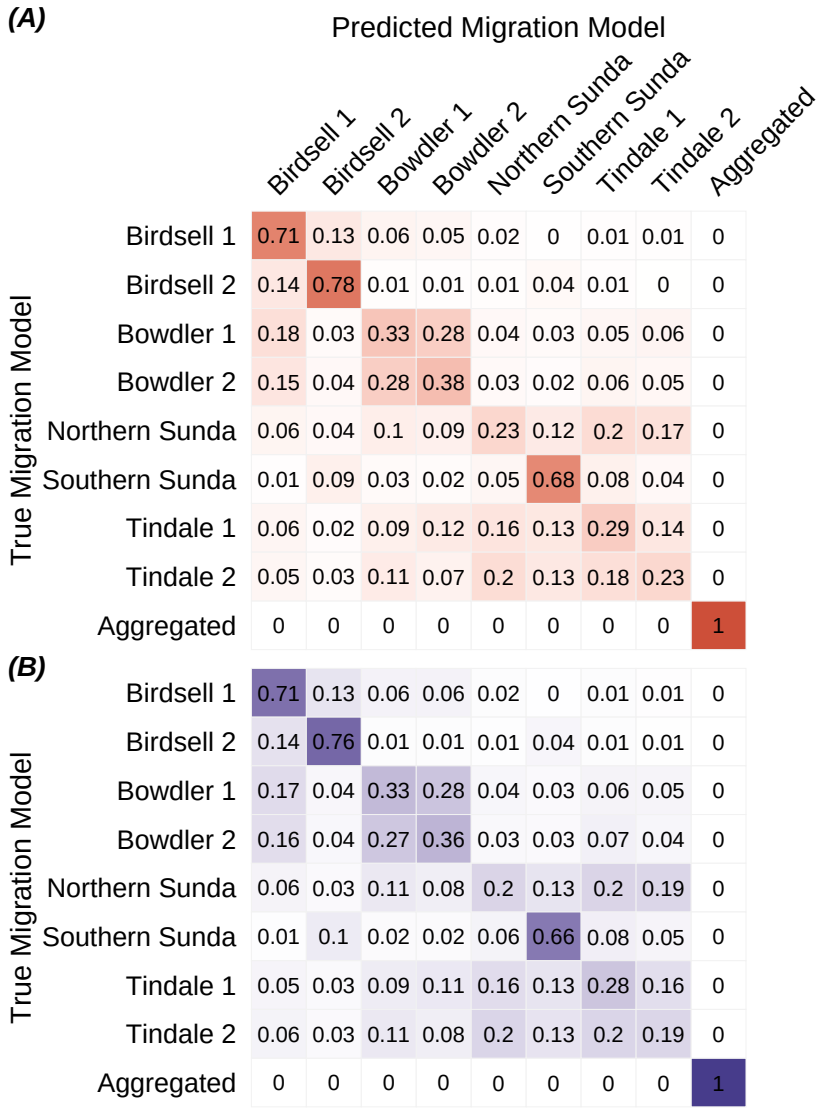


Figure C.10: Confusion matrices based on (A) the training data and (B) the test data when using MLR with forward selection (Validation 1).

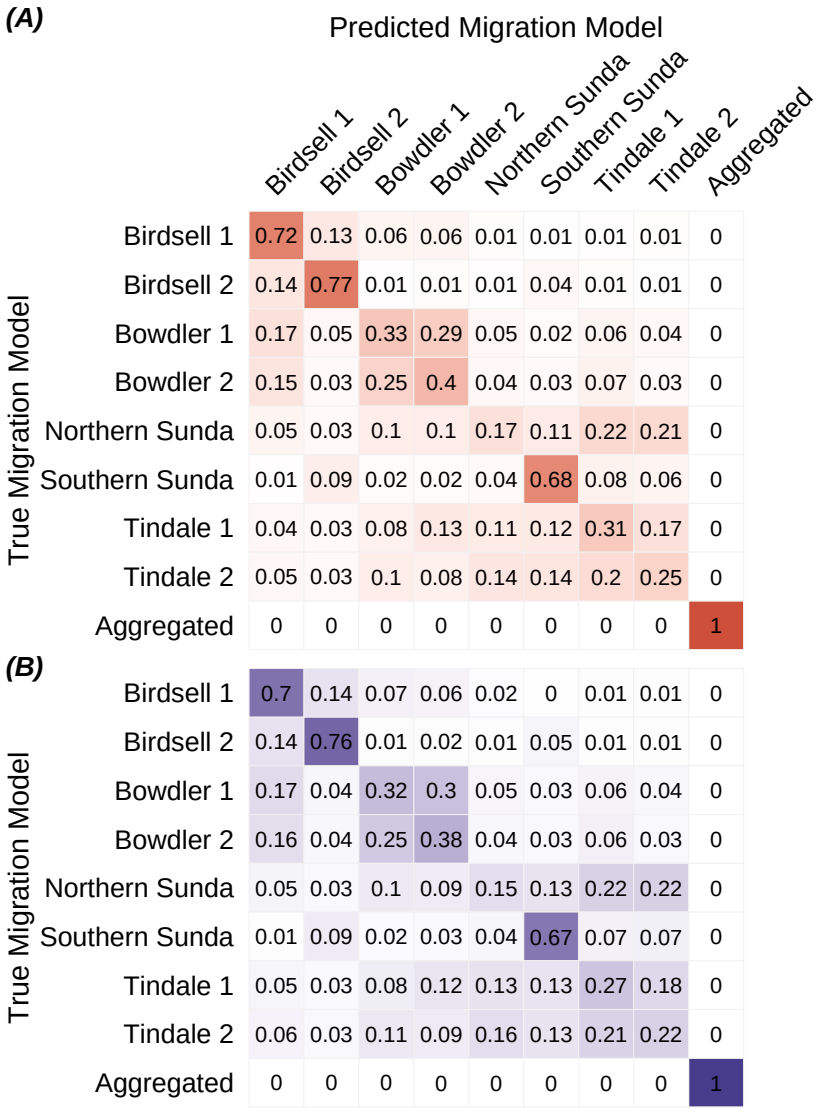


Figure C.11: Confusion matrices based on (A) the training data and (B) the test data when using MLR with forward selection (Validation 2).

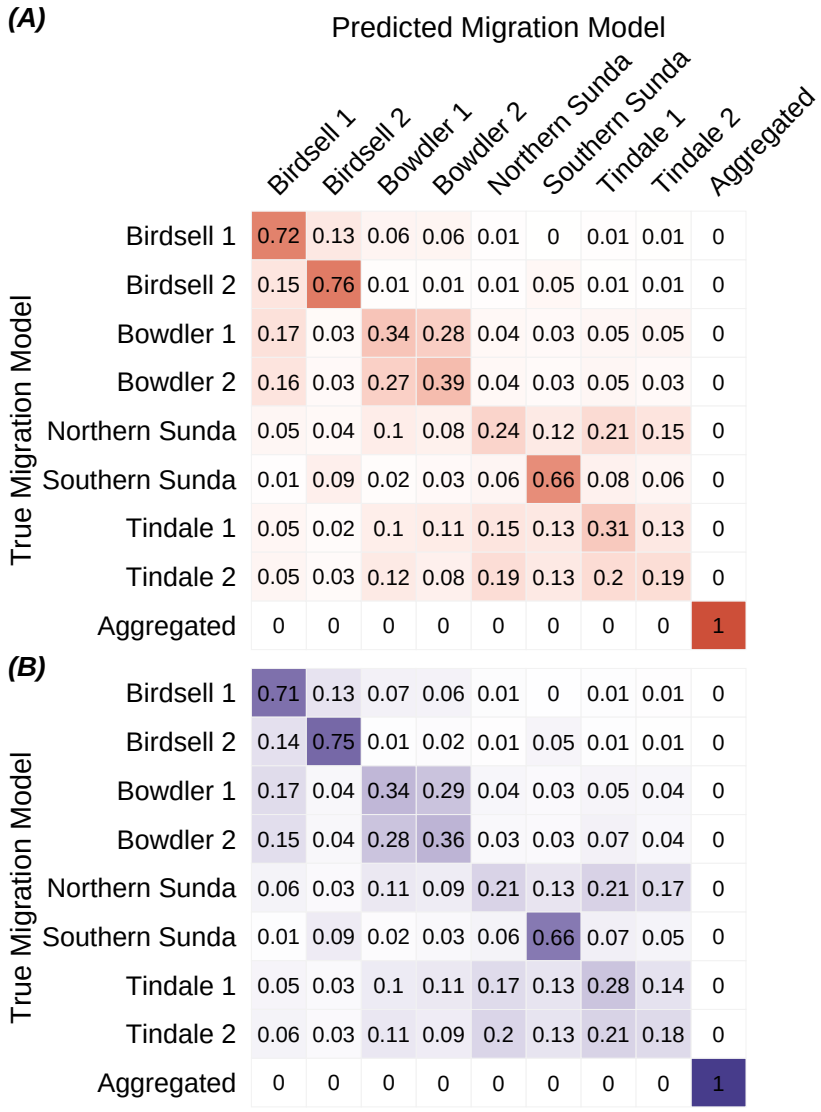


Figure C.12: Confusion matrices based on (A) the training data and (B) the test data when using MLR with forward selection (Validation 3).

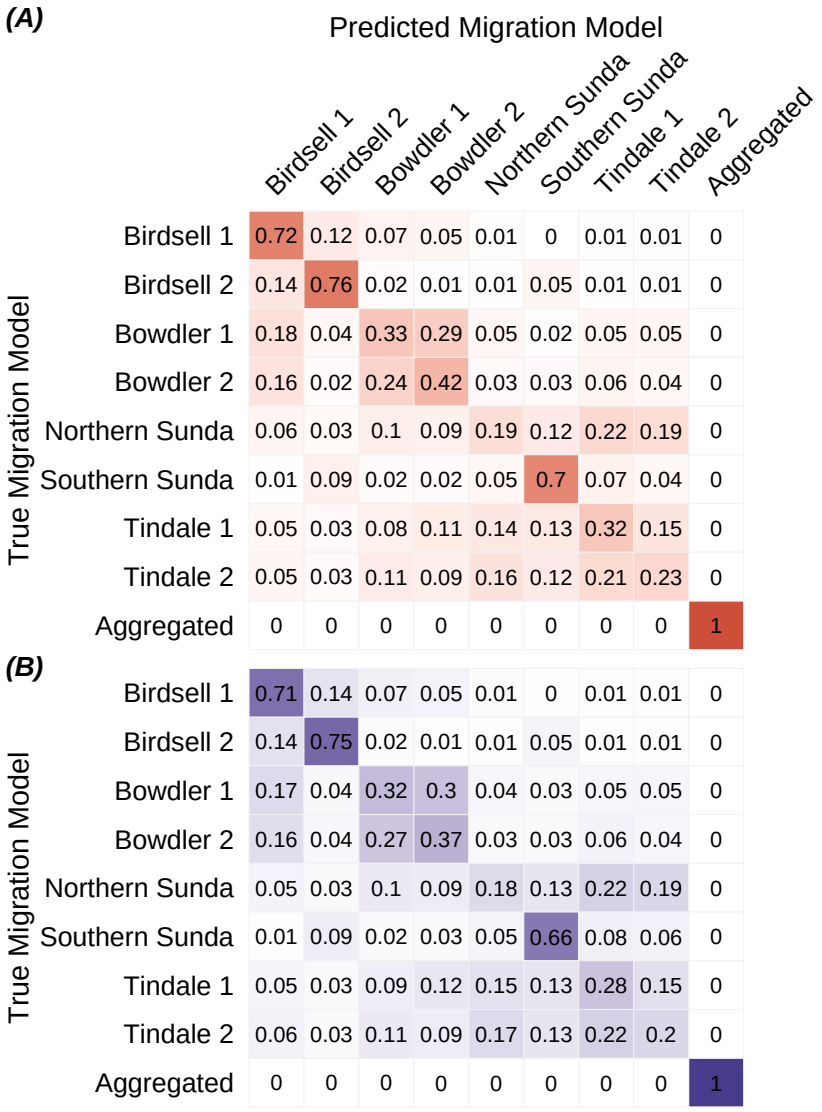


Figure C.13: Confusion matrices based on (A) the training data and (B) the test data when using MLR with forward selection (Validation 4).

Validation for MLR with LASSO

The training and test accuracy for the MLR classifier for each validation run, along with the penalization parameter λ, are given in Table C.13. For all classifiers we used λ_{1se}. The same data partitions were used as for MLR with forward selection, using 10% of the data as training data. The probabilities of the observed data belonging to each class for each of the migration models are given in Table C.14.

Validation	λ_{1se}	Training Accuracy	Test Accuracy
1	4.235×10^{-4}	51.0%	49.5%
2	3.515×10^{-4}	50.6%	49.3%
3	3.515×10^{-4}	50.7%	49.5%
4	2.012×10^{-4}	51.5%	49.5%

Table C.13: Training and test accuracy, along with λ_{1se} values for all MLR classifiers using LASSO for variable selection.

Migration Model	Probability (Validation 1)	Probability (Validation 2)	Probability (Validation 3)	Probability (Validation 4)
Birdsell 1	2.468×10^{-3}	3.535×10^{-3}	1.442×10^{-2}	2.993×10^{-3}
Birdsell 2	7.298×10^{-3}	7.251×10^{-3}	6.923×10^{-3}	5.632×10^{-3}
Bowdler 1	9.393×10^{-2}	0.104	7.807×10^{-2}	0.218
Bowdler 2	6.864×10^{-2}	9.099×10^{-2}	0.101	4.778×10^{-2}
Northern Sunda	0.175	0.130	0.140	0.140
Southern Sunda	0.349	0.323	0.232	0.384
Tindale 1	0.167	0.172	0.210	7.498×10^{-2}
Tindale 2	0.137	0.169	0.231	0.127
Aggregated	6.153×10^{-5}	3.236×10^{-5}	3.618×10^{-5}	1.370×10^{-5}

Table C.14: Class probabilities of the observed summary statistics belonging to each migration model, determined by MLR with LASSO.

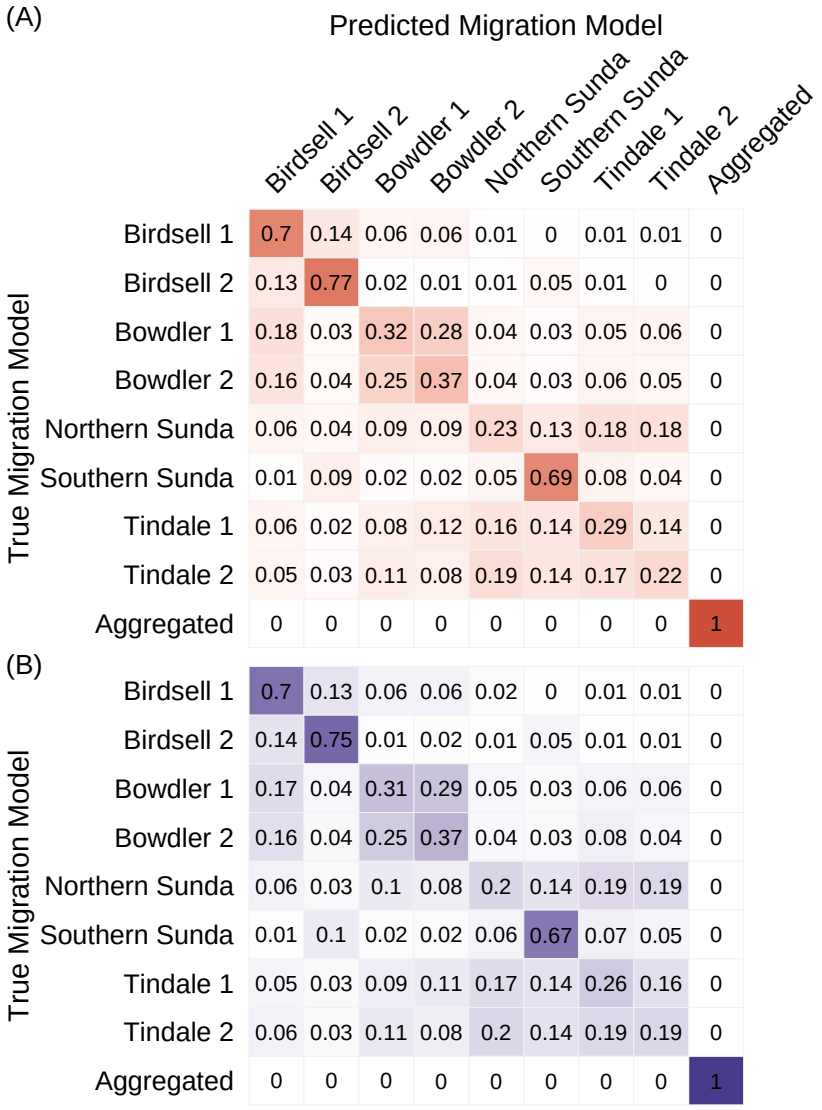


Figure C.14: Confusion matrices based on (A) the training data and (B) the test data when using MLR with LASSO (Validation 1).

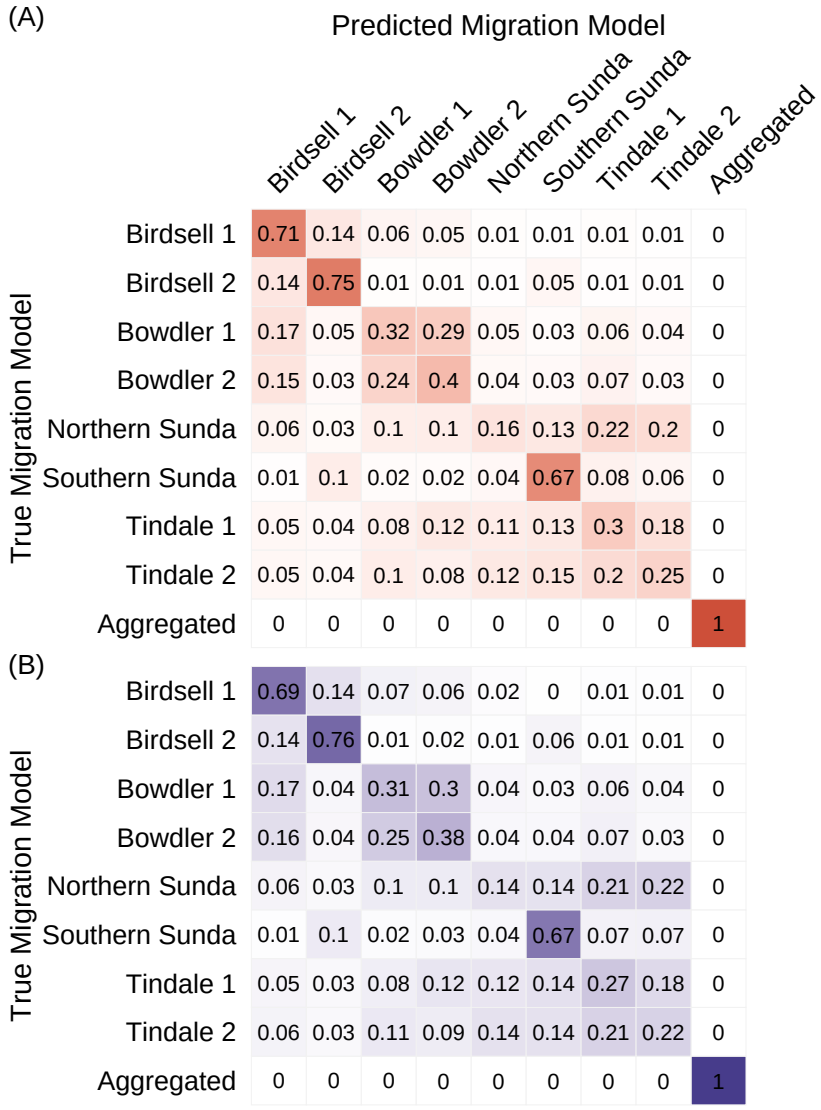


Figure C.15: Confusion matrices based on (A) the training data and (B) the test data when using MLR with LASSO (Validation 2).

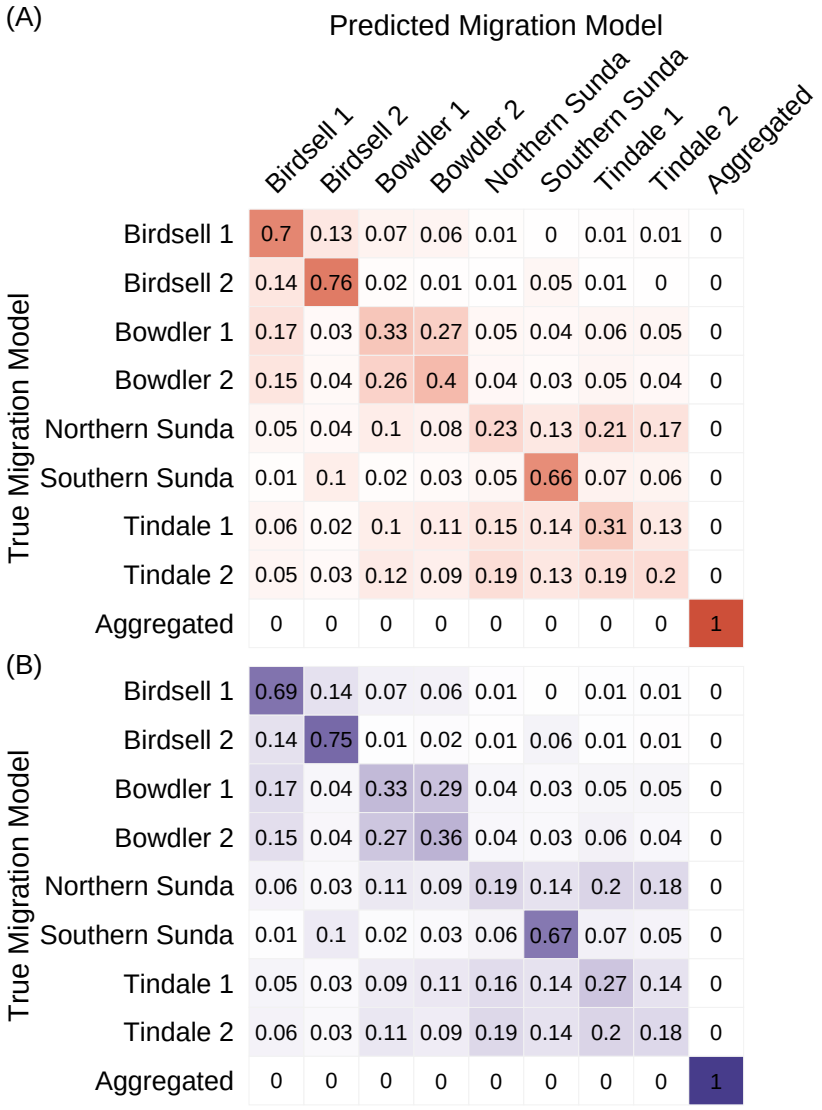


Figure C.16: Confusion matrices based on (A) the training data and (B) the test data when using MLR with LASSO (Validation 3).

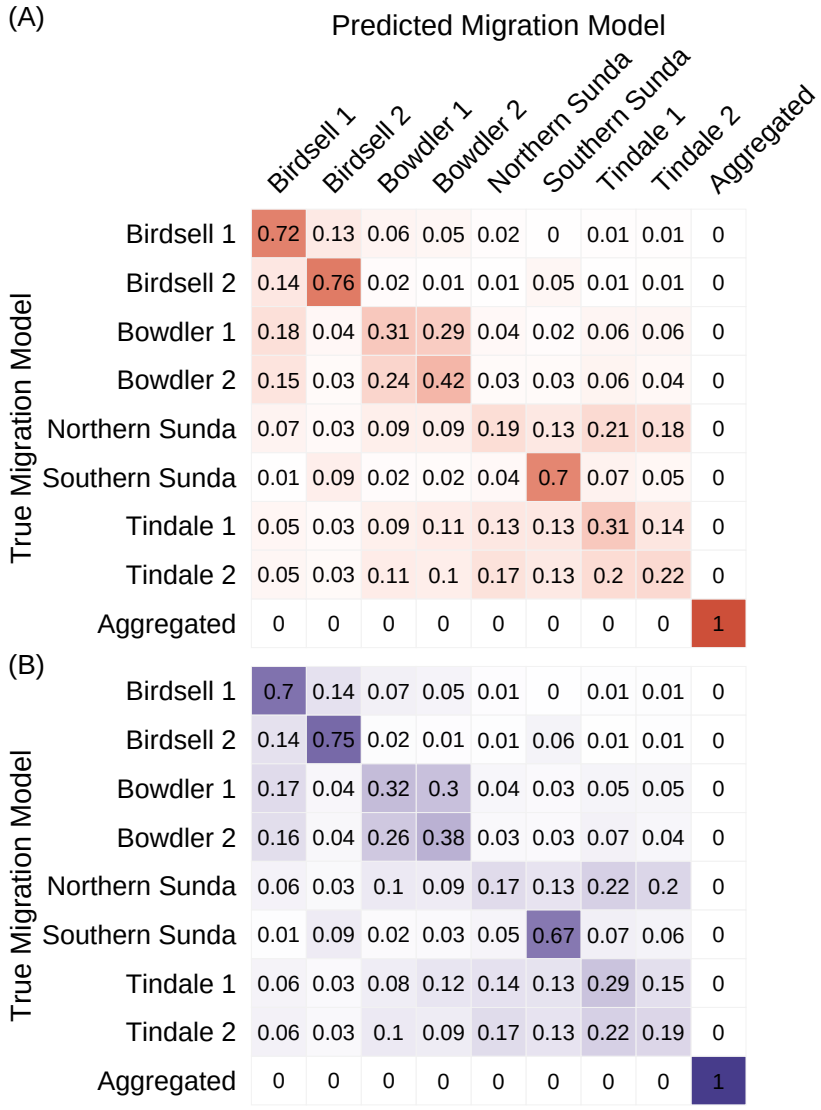


Figure C.17: Confusion matrices based on (A) the training data and (B) the test data when using MLR with LASSO (Validation 4).

Validations for a SVM with linear kernel

We trained four linear SVMs using the same data partitions as for MLR, *i.e.* using 10% of the data as training data for each validation. The cost parameter $C \in \mathbb{R}^+$ is given for each classifier in Table C.15, along with the corresponding validation accuracy and test accuracy, and the final class prediction for the observed data.

Validation	Cost	Validation Accuracy	Test Accuracy	Prediction
1	1	49.4%	49.7%	Tindale 1
2	32	49.6%	49.6%	Birdsell 1
3	8	49.2%	49.6%	Tindale 1
4	8	49.9%	49.6%	Birdsell 1

Table C.15: Cross-validated and test accuracies for the SVM trained in each validation. The cost parameter C for the linear SVMs is also given for each validation, as well as the predicted migration model for the observed summary statistics.

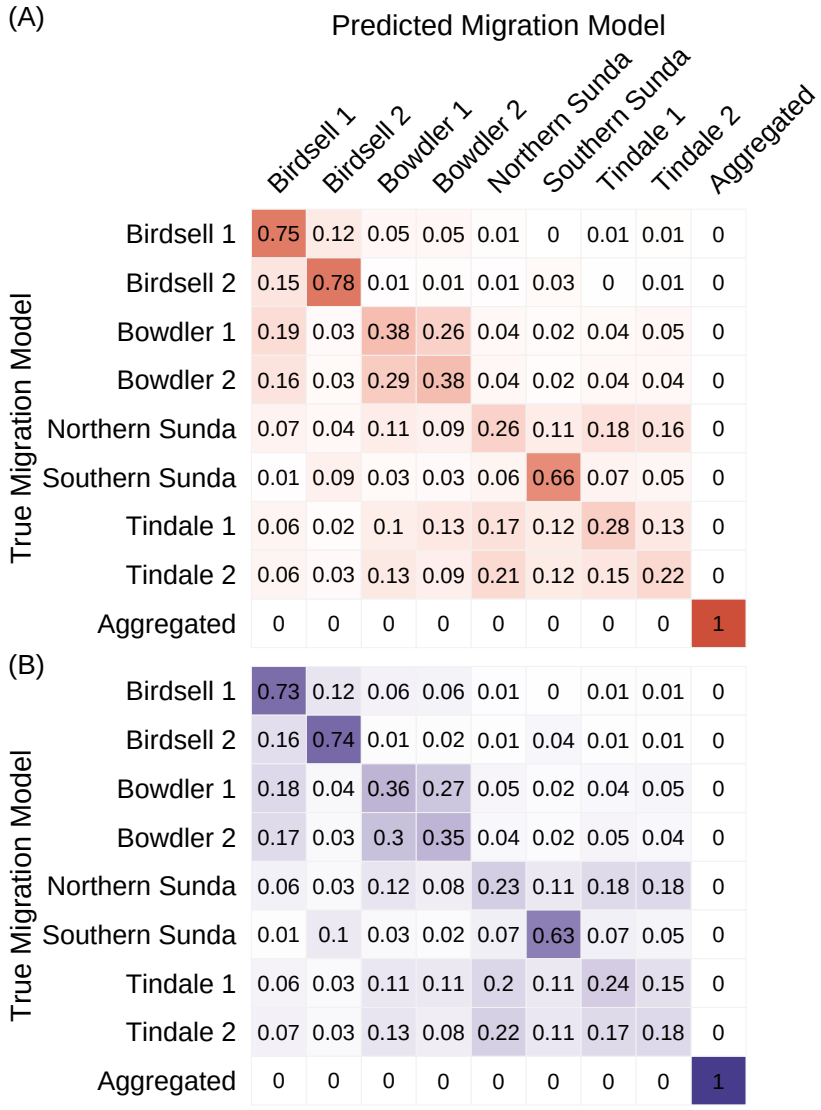


Figure C.18: Confusion matrices based on (A) the training data and (B) the test data when using a linear SVM (Validation 1).

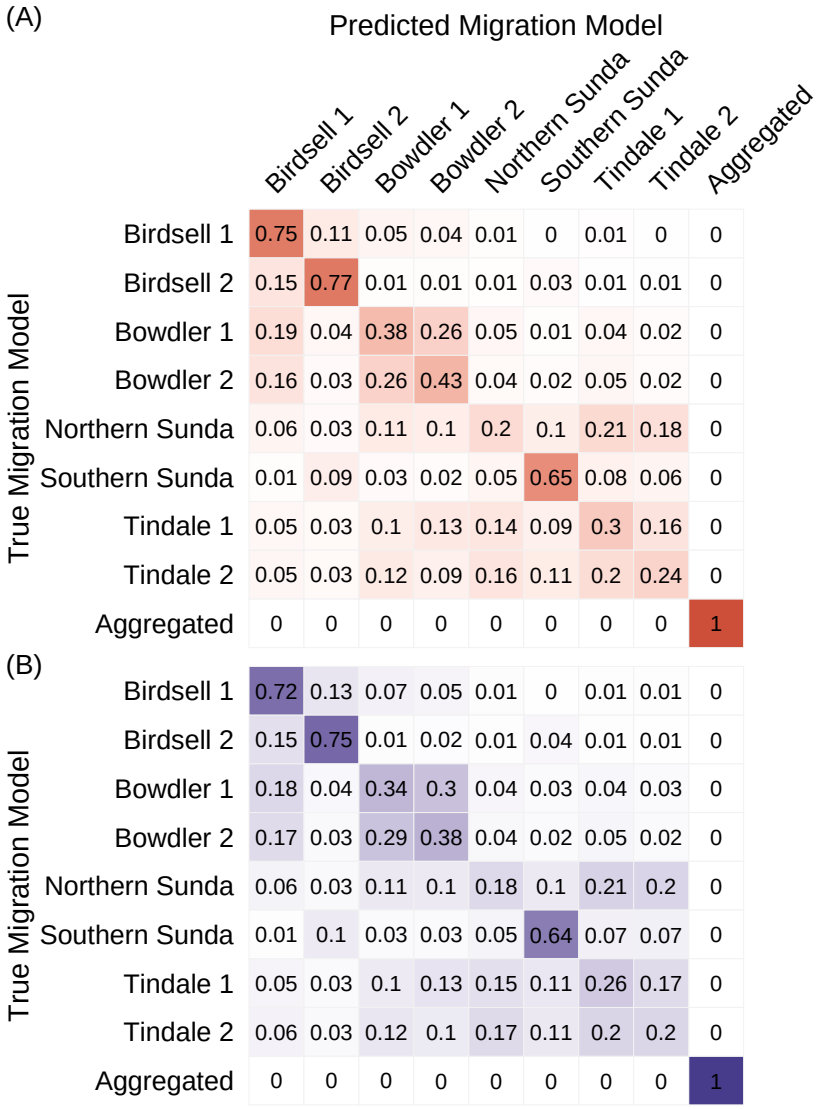


Figure C.19: Confusion matrices based on (A) the training data and (B) the test data when using a linear SVM (Validation 2).

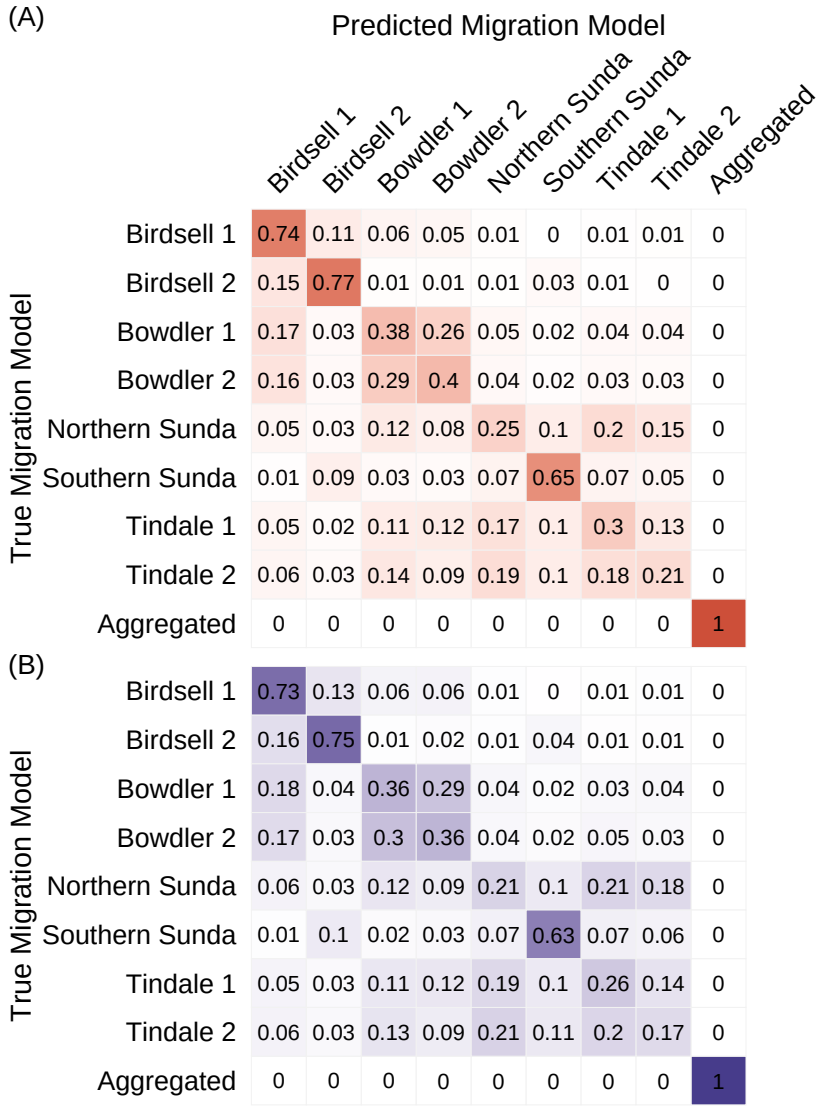


Figure C.20: Confusion matrices based on (A) the training data and (B) the test data when using a linear SVM (Validation 3).

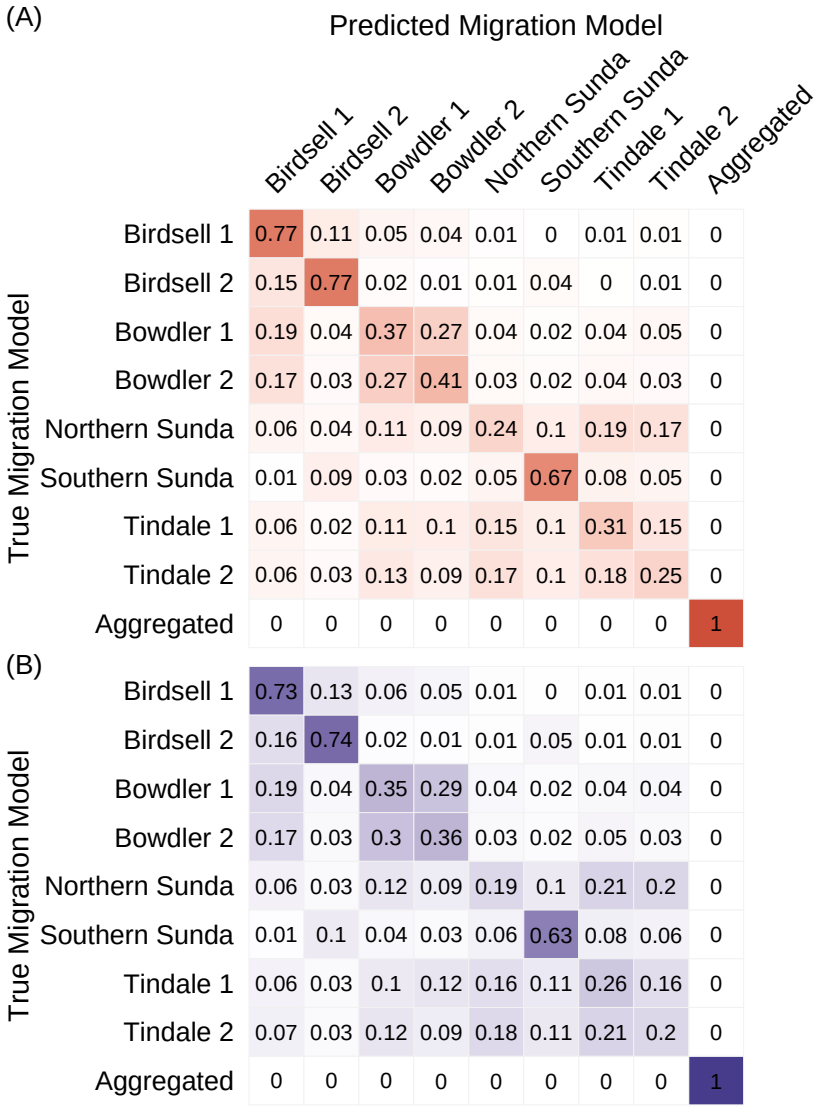


Figure C.21: Confusion matrices based on (A) the training data and (B) the test data when using a linear SVM (Validation 4).

Neural network validations

For the neural network validations, we used the same data partitions and the same parameter values as the original analysis. This is because neural networks use stochastic methods to minimize the loss function, which means it is possible to obtain different results each time.

The validation and test accuracies for each validation are given in Table C.16, and the output value of the classifier for each migration model in each validation is given in Table C.17. Recall that if a column for a migration model is missing, it means that the migration model was never predicted by the neural network.

Validation	Validation Accuracy	Test Accuracy
1	48.9%	48.6%
2	49.9%	49.7 %
3	48.7 %	48.5%
4	49.6%	49.3%

Table C.16: Validation and Training accuracies for each validation of the neural network classifier.

Migration Model	Probability (Validation 1)	Probability (Validation 2)	Probability (Validation 3)	Probability (Validation 4)
Birdsell 1	$< 2.2 \times 10^{-16}$	$< 2.2 \times 10^{-16}$	$< 2.2 \times 10^{-16}$	$< 2.2 \times 10^{-16}$
Birdsell 2	$< 2.2 \times 10^{-16}$	3.48×10^{-4}	2.00×10^{-15}	$< 2.2 \times 10^{-16}$
Bowdler 1	$< 2.2 \times 10^{-16}$	$< 2.2 \times 10^{-16}$	$< 2.2 \times 10^{-16}$	$< 2.2 \times 10^{-16}$
Bowdler 2	$< 2.2 \times 10^{-16}$	$< 2.2 \times 10^{-16}$	$< 2.2 \times 10^{-16}$	$< 2.2 \times 10^{-16}$
Northern Sunda	$< 2.2 \times 10^{-16}$	$< 2.2 \times 10^{-16}$	$< 2.2 \times 10^{-16}$	$< 2.2 \times 10^{-16}$
Southern Sunda	$< 2.2 \times 10^{-16}$	≈ 1.00	≈ 1.00	$< 2.2 \times 10^{-16}$
Tindale 1	$< 2.2 \times 10^{-16}$	$< 2.2 \times 10^{-16}$	$< 2.2 \times 10^{-16}$	$< 2.2 \times 10^{-16}$
Tindale 2	$< 2.2 \times 10^{-16}$	$< 2.2 \times 10^{-16}$	$< 2.2 \times 10^{-16}$	$< 2.2 \times 10^{-16}$
Aggregated	≈ 1.00	$< 2.2 \times 10^{-16}$	$< 2.2 \times 10^{-16}$	≈ 1.00

Table C.17: The output value of the softmax layer of neural networks for each validation. Values that were given as zero are presented in this table as less than machine precision (*i.e.* $< 2.2 \times 10^{-16}$.

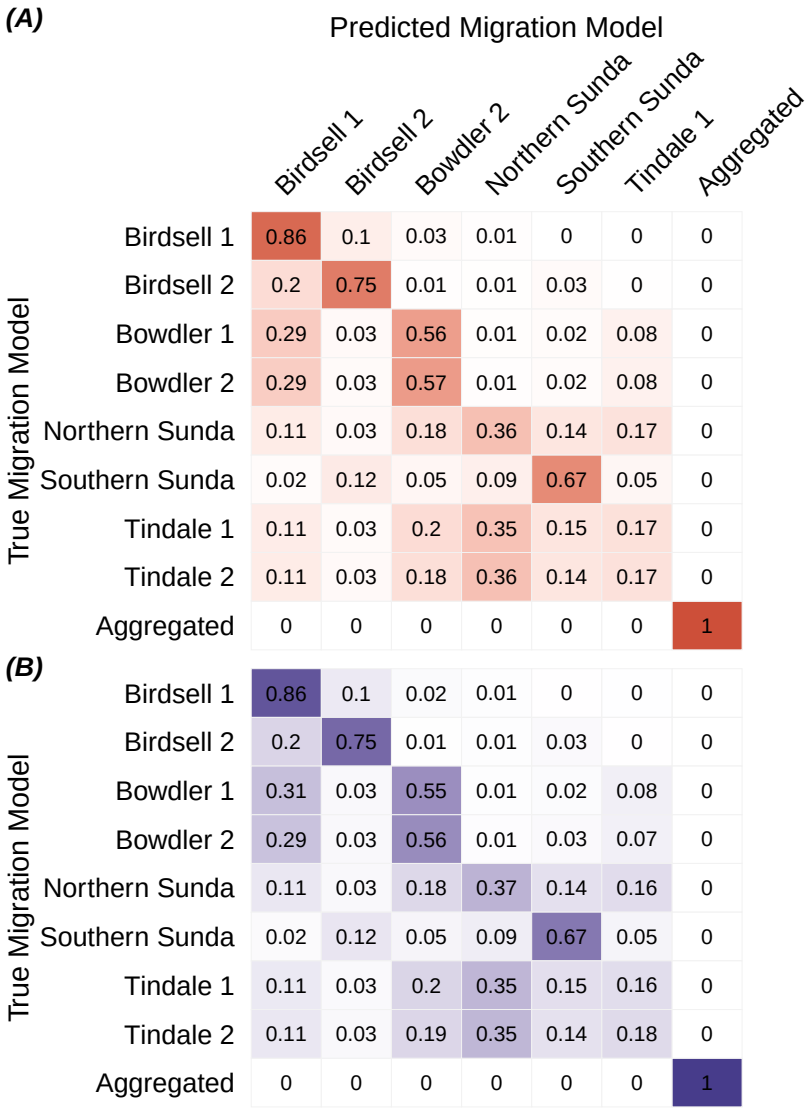


Figure C.22: Confusion matrices based on (A) the training data and (B) the test data when using a neural network with Architecture 2 (Validation 1).

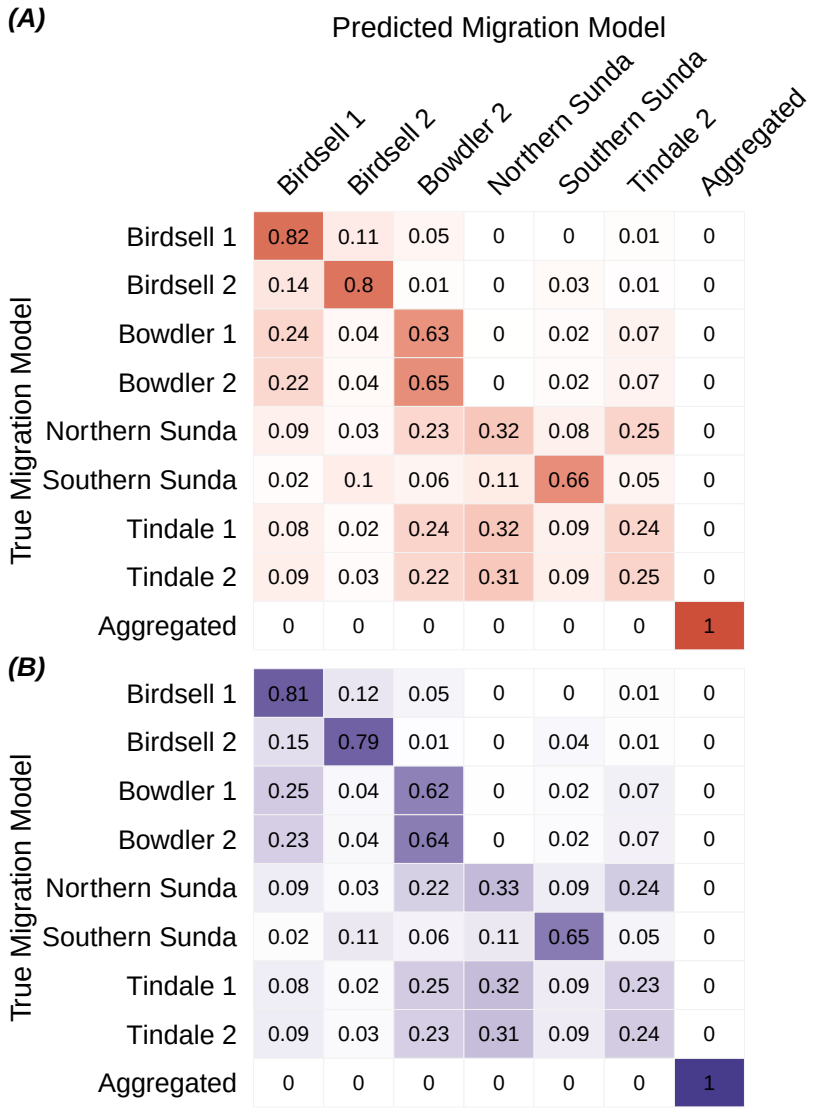


Figure C.23: Confusion matrices based on (A) the training data and (B) the test data when using a neural network with Architecture 2 (Validation 1).

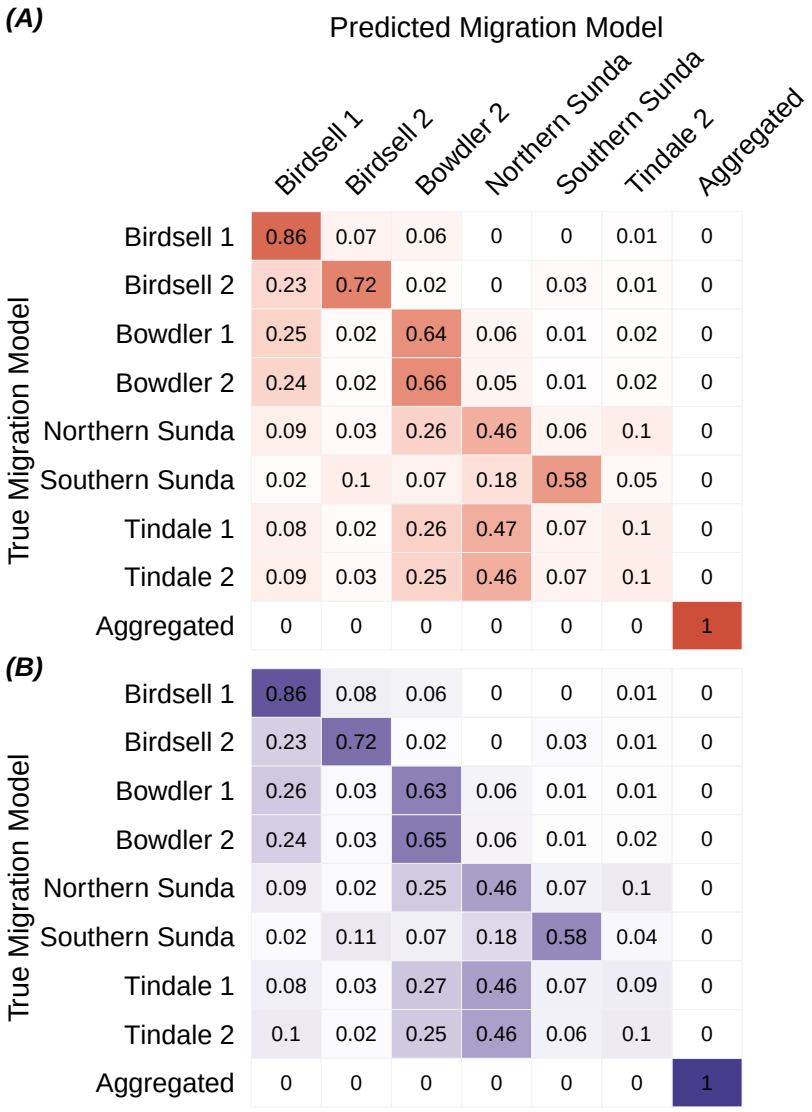


Figure C.24: Confusion matrices based on (A) the training data and (B) the test data when using a neural network with Architecture 2 (Validation 1).

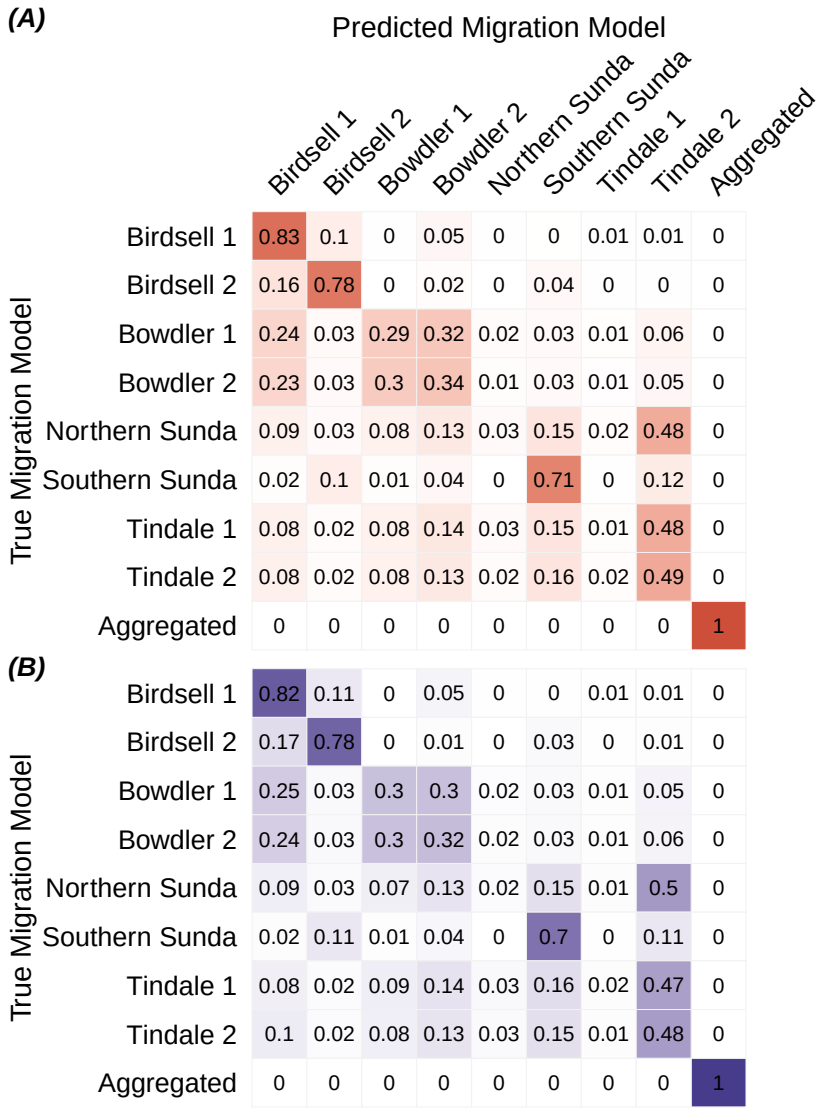


Figure C.25: Confusion matrices based on (A) the training data and (B) the test data when using a neural network with Architecture 2 (Validation 1).

C.3 Extended Results from Chapter 7

C.3.1 UMAP dimension reduction for summary statistics assuming homogeneous post-settlement migration

The UMAP dimension reduction displayed in Figure C.26 was performed with the parameters `n_neighbors` = 100, `min_dist` = 0.5, and `random_state` = 65640. We notice that the summary statistics from different migration models are generally indistinguishable in this lower-dimensional space, with the exception of the aggregated model.

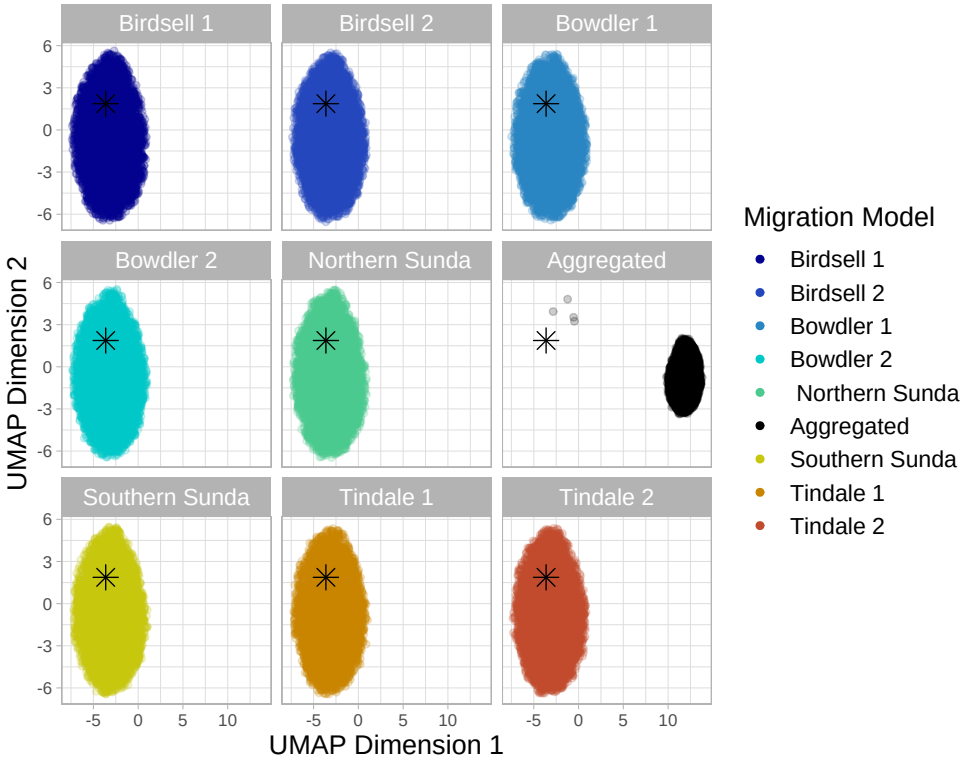


Figure C.26: The UMAP dimension reduction of the summary statistics from migration models, assuming homogeneous post-settlement migration patterns. The plot was facetted by migration model due to the significant overlap of the migration models. The observed summary statistics are given as the black star in all panels.

C.3.2 Distances between the centers of simulated summary statistics for each migration model

Manhattan distance between geometric medians

(Between)	Birdsell 2	Bowdler 1	Bowdler 2	Aggregated
Birdsell 1	129.6 8.0	10386.4 298.1	122.2 8.1	53.7 4.3
Birdsell 2	- -	10310.9 302.4	49.2 8.4	79.4 5.6
Bowdler 1	- -	- -	10342.8 299.7	10349.4 300.5
Bowdler 2	- -	- -	- -	126.3 9.6
Aggregated	- -	- -	- -	- -
Northern Sunda	- -	- -	- -	100.9 8.4
Southern Sunda	- -	- -	- -	- -
Tindale 1	- -	- -	- -	- -

(Between)	Northern Sunda	Southern Sunda	Tindale 1	Tindale 2
Birdsell 1	53.1 8.4	8.5 9.8	62.5 10.1	3.4 2.3
Birdsell 2	176.9 10.6	127.2 13.4	75.4 16.3	128.2 7.3
Bowdler 1	10422.8 303.1	10381.0 299.8	10343.2 298.4	10386.4 298.3
Bowdler 2	131.5 11.0	119.7 14.4	122.8 13.6	123.3 6.6
Aggregated	- -	57.8 11.2	9.8 11.9	52.4 3.6
Northern Sunda	- -	59.7 7.1	109.8 8.6	53.3 7.0
Southern Sunda	- -	- -	55.1 5.7	10.2 8.9
Tindale 1	- -	- -	- -	61.8 10.4

Table C.18: L1 (Manhattan) distance between the geometric medians of the summary statistics for each pair of migration models. The distances for the original migration models are the topmost element of each cell, while the distances for the migration models assuming homogeneous post-settlement migration are the second element given in each cell.

Euclidean Distance between Centroids

(Between)	Birdsell 2	Bowdler 1	Bowdler 2	Aggregated
Birdsell 1	7719.5 99.5	27.1 1.1	67.6 2.5	52.2 3.9
Birdsell 2	- -	7719.9 99.4	7721.1 100.3	7720.6 100.8
Bowdler 1	- -	- -	54.2 2.5	48.4 4.1
Bowdler 2	- -	- -	- -	23.7 2.6
Aggregated	- -	- -	- -	- -
Northern Sunda	- -	- -	- -	-
Southern Sunda	- -	- -	- -	- -
Tindale 1	- -	- -	- -	- -

(Between)	Northern Sunda	Southern Sunda	Tindale 1	Tindale 2
Birdsell 1	34.3 1.3	32.3 3.9	53.7 2.2	53.4 2.0
Birdsell 2	7720.0 100.0	7720.0 100.9	7720.6 100.2	7720.6 99.9
Bowdler 1	43.8 1.4	42.3 4.1	50.4 2.0	49.8 1.9
Bowdler 2	49.1 1.5	49.8 2.9	24.0 1.7	23.7 1.8
Aggregated	28.9 3.1	29.5 1.3	3.4 3.0	2.7 3.0
Northern Sunda	- -	3.2 3.2	28.9 1.4	29.0 1.3
Southern Sunda	- -	- -	29.9 2.8	30.0 2.8
Tindale 1	- -	- -	- -	1.0 0.7

Table C.19: L2 (Euclidean) distance between the centroids of the summary statistics for each pair of migration models. The distances for the original migration models are the topmost element of each cell, while the distances for the migration models assuming homogeneous post-settlement migration are the second element given in each cell.

Euclidean Distance between Geometric Medians

(Between)	Birdsell 2	Bowdler 1	Bowdler 2	Aggregated
Birdsell 1	53.6 2.7	7726.0 99.0	49.8 2.1	29.1 1.7
Birdsell 2	- -	7724.8 98.3	27.2 2.4	34.6 1.7
Bowdler 1	- -	- -	7725.3 98.9	7725.3 99.0
Bowdler 2	- -	- -	- -	44.0 1.8
Aggregated	- -	- -	- -	- -
Northern Sunda	- -	- -	- -	- -
Southern Sunda	- -	- -	- -	- -
Tindale 1	- -	- -	- -	- -

(Between)	Northern Sunda	Southern Sunda	Tindale 1	Tindale 2
Birdsell 1	23.1 2.3	3.1 3.0	29.9 3.2	1.2 0.6
Birdsell 2	67.3 3.3	51.9 4.2	32.1 4.5	53.1 2.6
Bowdler 1	7726.4 100.0	7725.9 100.1	7725.3 100.1	7725.9 99.0
Bowdler 2	54.0 3.2	48.1 4.3	42.1 4.3	49.9 2.0
Aggregated	48.6 2.6	28.6 3.4	3.6 3.6	28.3 1.6
Northern Sunda	- -	23.4 2.3	49.2 2.5	23.9 2.1
Southern Sunda	- -	- -	29.1 1.9	3.4 3.1
Tindale 1	- -	- -	- -	29.2 3.3

Table C.20: L2 (Euclidean) distance between the geometric medians of the summary statistics for each pair of migration models. The distances for the original migration models are the topmost element of each cell, while the distances for the migration models assuming homogeneous post-settlement migration are the second element given in each cell.

C.3.3 Confusion matrices for training data from Section 7.3.1

In Section 7.3.1 we used MLR with LASSO to classify the simulated summary statistics from simulations that assumed 2,500 years, 25,000 years, and 250,000 years between migration events through the southeast Asian islands. In Table C.21, we present the confusion matrices calculated from the training data. They show similar patterns of misclassification to those calculated from the test data.

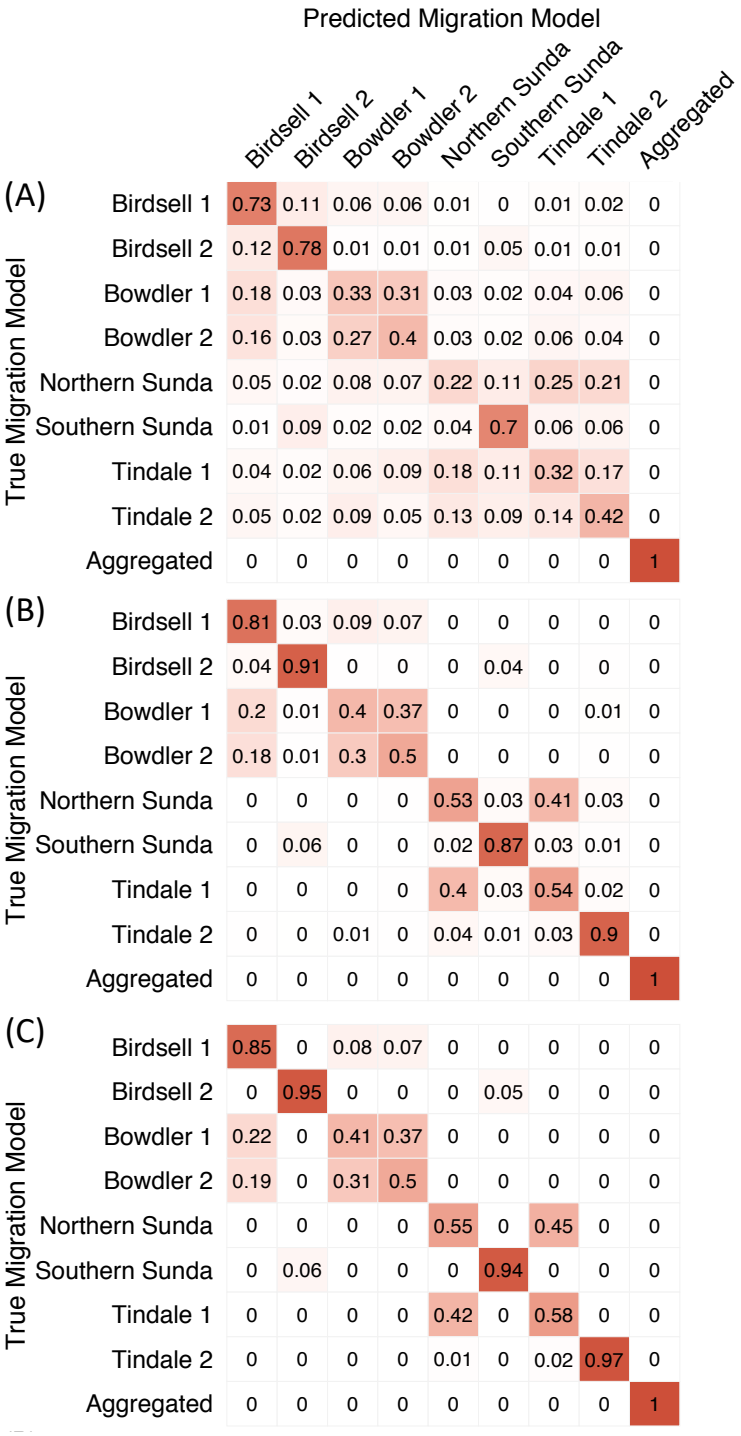


Table C.21: Confusion matrices for the true and predicted migration models of summary statistics simulated assuming (A) 2,500 years, (B) 25,000 years, and (C) 250,000 years between migration events through southeast Asia. All confusion matrices are calculated from training data. The confusion matrices calculated from the training data are nearly identical, and are presented in Appendix C.3.3. The proportion of observations that fall into each category is given, with rows summing to one. Darker squares indicate greater proportions.

C.3.4 Effects of excluding west coast migration and requiring homogeneous post-settlement migration

For MLR with LASSO, we partition the data using 70% for training data and the remaining 30% for test data. A λ value of 2.563×10^{-3} was selected through a 5-fold cross-validation process.

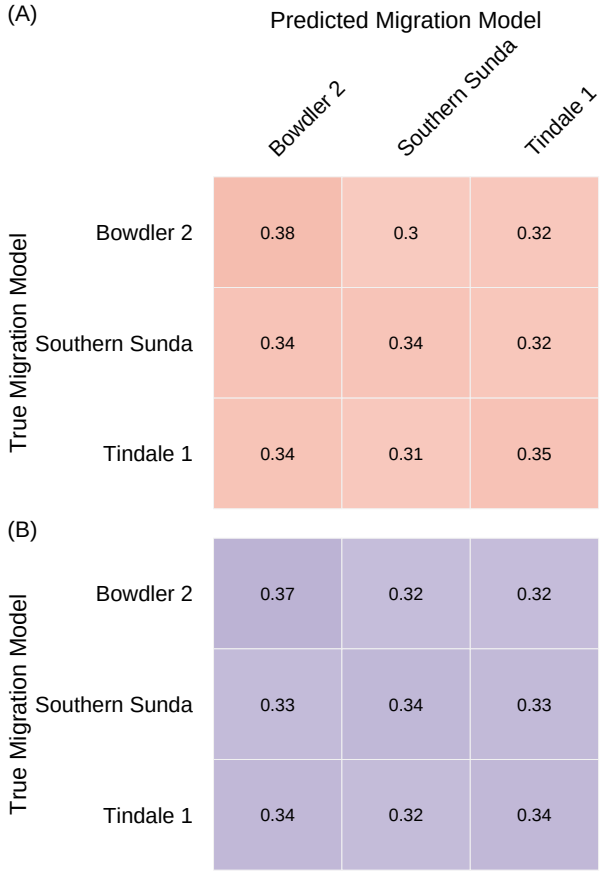


Table C.22: Confusion matrices for the simulated summary statistics, assuming no migration around the western coast of Australia, as well as homogeneous post-settlement migration patterns. Matrix (A) is calculated from the training data, while matrix (B) was calculated from the test data. The proportion of observations that fall into each category is given, with rows summing to one. Darker squares indicate greater proportions.

Bibliography

[1] Leslie C Aiello. Five years of *Homo floresiensis*. *American Journal of Physical Anthropology: The Official Publication of the American Association of Physical Anthropologists*, 142(2):167–179, 2010.

[2] Hirotugu Akaike. A new look at the statistical model identification. *IEEE transactions on automatic control*, 19(6):716–723, 1974.

[3] Stephen F Altschul, Warren Gish, Webb Miller, Eugene W Myers, and David J Lipman. Basic local alignment search tool. *Journal of Molecular Biology*, 215(3):403–410, 1990.

[4] Christian NK Anderson, Uma Ramakrishnan, Yvonne L Chan, and Elizabeth A Hadly. Serial SimCoal: a population genetics model for data from multiple populations and points in time. *Bioinformatics*, 21(8):1733–1734, 2004.

[5] Joseph B Birdsell. Some population problems involving Pleistocene man. In *Cold Spring Harbor Symposia on Quantitative Biology*, volume 22, pages 47–69. Cold Spring Harbor Laboratory Press, 1957.

[6] Joseph B Birdsell. The recalibration of a paradigm for the first peopling of Greater Australia. *Sunda and Sahul: Prehistoric Studies in Southeast Asia, Melanesia, and Australia*, pages 113–167, 1977.

[7] Joseph K Blitzstein. *Introduction to Probability*. CRC Press, Taylor & Francis Group, Boca Raton, second edition, 2019.

[8] Remco Bouckaert, Joseph Heled, Denise Kühnert, Tim Vaughan, Chieh-Hsi Wu, Dong Xie, Marc A Suchard, Andrew Rambaut, and Alexei J Drummond. BEAST 2: a software platform for Bayesian evolutionary analysis. *PLoS Computational Biology*, 10(4):e1003537, 2014.

[9] Remco R Bouckaert and Joseph Heled. Densitree 2: Seeing trees through the forest. *BioRxiv*, page 012401, 2014.

[10] Sandra Bowdler. The coastal colonisation of Australia. *Sunda and Sahul: prehistoric studies in southeast Asia, Melanesia and Australia*, pages 205–246, 1977.

[11] Adrian W Briggs, Jeffrey M Good, Richard E Green, Johannes Krause, Tomislav Maricic, Udo Stenzel, Carles Lalueza-Fox, Pavao Rudan, Dejana Brajković, Željko Kućan, et al. Targeted retrieval and analysis of five Neandertal mtDNA genomes. *Science*, 325(5938):318–321, 2009.

[12] Chih-Chung Chang and Chih-Jen Lin. LIBSVM: A library for support vector machines. *ACM transactions on intelligent systems and technology (TIST)*, 2(3):27, 2011.

[13] François Chollet et al. Keras. `https://keras.io`, 2015.

[14] Francois Chollet. *Deep learning with Python*. Manning Publications Company, 2017.

[15] Chris Clarkson, Zenobia Jacobs, Ben Marwick, Richard Fullagar, Lynley Wallis, Mike Smith, Richard G Roberts, Elspeth Hayes, Kelsey Lowe, Xavier Carah, et al. Human occupation of northern Australia by 65,000 years ago. *Nature*, 547(7663):306, 2017.

[16] Matthew Coller. SahulTime: rethinking archaeological representation in the digital age. *Archaeologies*, 5(1):110–123, 2009.

[17] Corinna Cortes and Vladimir Vapnik. Support-vector networks. *Machine Learning*, 20(3):273–297, 1995.

[18] Robin Derricourt. Getting "Out of Africa": sea crossings, land crossings and culture in the hominin migrations. *Journal of World Prehistory*, 19(2):119–132, 2005.

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

[20] Alexei J Drummond, Andrew Rambaut, Beth Shapiro, and Oliver G Pybus. Bayesian coalescent inference of past population dynamics from molecular sequences. *Molecular Biology and Evolution*, 22(5):1185–1192, 2005.

[21] Ana T Duggan, Bethwyn Evans, Françoise R Friedlaender, Jonathan S Friedlaender, George Koki, D Andrew Merriwether, Manfred Kayser, and Mark Stoneking. Maternal history of Oceania from complete mtDNA genomes: contrasting ancient diversity with recent homogenization due to the Austronesian expansion. *The American Journal of Human Genetics*, 94(5):721–733, 2014.

[22] Robert C Elston, Jaya M Satagopan, and Shuying Sun. Genetic terminology. In *Statistical Human Genetics*, pages 1–9. Springer, 2012.

[23] Laurent Excoffier, J Novembre, and Stefan Schneider. SIMCOAL: a general coalescent program for the simulation of molecular data in interconnected populations with arbitrary demography. *Journal of Heredity*, 91(6):506–509, 2000.

[24] Jack N Fenner. Cross-cultural estimation of the human generation interval for use in genetics-based population divergence studies. *American Journal of Physical Anthropology: The Official Publication of the American Association of Physical Anthropologists*, 128(2):415–423, 2005.

[25] Ronald A Fisher. XXI. On the dominance ratio. *Proceedings of the Royal Society of Edinburgh*, 42:321–341, 1923.

[26] Jerome Friedman, Trevor Hastie, and Robert Tibshirani. *The Elements of Statistical Learning*, volume 1. Springer series in statistics New York, 2001.

[27] Jerome Friedman, Trevor Hastie, and Robert Tibshirani. Regularization Paths for Generalized Linear Models via Coordinate Descent. *Journal of Statistical Software*, 33(1):1–22, 2010.

[28] Qiaomei Fu, Alissa Mittnik, Philip LF Johnson, Kirsten Bos, Martina Lari, Ruth Bollongino, Chengkai Sun, Liane Giemsch, Ralf Schmitz, Joachim Burger, et al. A revised timescale for human evolution based on ancient mitochondrial genomes. *Current Biology*, 23(7):553–559, 2013.

[29] Sibylle M Gomes, Martin Bodner, Luis Souto, Bettina Zimmermann, Gabriela Huber, Christina Strobl, Alexander W Röck, Alessandro Achilli, Anna Olivieri, Antonio Torroni, et al. Human settlement history between Sunda and Sahul: a focus on East Timor (Timor-Leste) and the Pleistocenic mtDNA diversity. *BMC Genomics*, 16(1):70, 2015.

[30] Stéphane Guindon, Jean-François Dufayard, Vincent Lefort, Maria Anisimova, Wim Hordijk, and Olivier Gascuel. New algorithms and methods to estimate maximum-likelihood phylogenies: assessing the performance of PhyML 3.0. *Systematic Biology*, 59(3):307–321, 2010.

[31] Stéphane Guindon and Olivier Gascuel. A simple, fast, and accurate algorithm to estimate large phylogenies by maximum likelihood. *Systematic Biology*, 52(5):696–704, 2003.

[32] Masami Hasegawa, Hirohisa Kishino, and Taka-aki Yano. Dating of the human-ape splitting by a molecular clock of mitochondrial DNA. *Journal of Molecular Evolution*, 22(2):160–174, 1985.

[33] Joseph Heled and Remco R Bouckaert. Looking for trees in the forest: summary tree from posterior samples. *BMC Evolutionary Biology*, 13(1):221, 2013.

[34] Joseph Heled and Alexei J Drummond. Bayesian inference of population size history from multiple loci. *BMC Evolutionary Biology*, 8(1):289, 2008.

[35] Chih-Wei Hsu, Chih-Chung Chang, Chih-Jen Lin, et al. A practical guide to support vector classification, 2003.

[36] Richard R Hudson, Montgomery Slatkin, and Wayne P Maddison. Estimation of levels of gene flow from DNA sequence data. *Genetics*, 132(2):583–589, 1992.

[37] Timothy A Jinam, Lih-Chun Hong, Maude E Phipps, Mark Stoneking, Mahmood Ameen, Juli Edo, HUGO Pan-Asian SNP Consortium, and Naruya Saitou. Evolutionary history of continental Southeast Asians : "Early train" hypothesis based on genetic analysis of mitochondrial and autosomal DNA data. *Molecular Biology and Evolution*, 29(11):3513–3527, 2012.

[38] Ian T Jolliffe and Jorge Cadima. Principal component analysis: a review and recent developments. *Philosophical Transactions of the Royal Society A: Mathematical, Physical and Engineering Sciences*, 374(2065):20150202, 2016.

[39] Thomas H Jukes, Charles R Cantor, et al. Evolution of protein molecules. *Mammalian Protein Metabolism*, 3(21):132, 1969.

[40] Bekir Karlik and A Vehbi Olgac. Performance analysis of various activation functions in generalized MLP architectures of neural networks. *International Journal of Artificial Intelligence and Expert Systems*, 1(4):111–122, 2011.

[41] Shimona Kealy, Julien Louys, and Sue O'Connor. Least-cost pathway models indicate northern human dispersal from Sunda to Sahul. *Journal of Human Evolution*, 125:59–70, 2018.

[42] Thomas M Keane, Christopher J Creevey, Melissa M Pentony, Thomas J Naughton, and James O Mclnerney. Assessment of methods for amino acid matrix selection and their use on empirical data shows that ad hoc assumptions for choice of matrix are not justified. *BMC Evolutionary Biology*, 6(1):29, 2006.

[43] Motoo Kimura. A simple method for estimating evolutionary rates of base substitutions through comparative studies of nucleotide sequences. *Journal of Molecular Evolution*, 16(2):111–120, 1980.

[44] Richard G Klein. Out of Africa and the evolution of human behavior. *Evolutionary Anthropology: Issues, News, and Reviews*, 17(6):267–281, 2008.

[45] Kurt Lambeck and John Chappell. Sea level change through the last glacial cycle. *Science*, 292(5517):679–686, 2001.

[46] Sebastian Lippold, Hongyang Xu, Albert Ko, Mingkun Li, Gabriel Renaud, Anne Butthof, Roland Schröder, and Mark Stoneking. Human paternal and maternal demographic histories: insights from high-resolution Y chromosome and mtDNA sequences. *Investigative Genetics*, 5(1):13, 2014.

[47] Mark Lipson, Po-Ru Loh, Nick Patterson, Priya Moorjani, Ying-Chin Ko, Mark Stoneking, Bonnie Berger, and David Reich. Reconstructing Austronesian population history in island Southeast Asia. *Nature Communications*, 5(1):1–7, 2014.

[48] Bastien Llamas, Lars Fehren-Schmitz, Guido Valverde, Julien Soubrier, Swapan Mallick, Nadin Rohland, Susanne Nordenfelt, Cristina Valdiosera, Stephen M Richards, Adam Rohrlach, et al. Ancient mitochondrial DNA provides high-resolution time scale of the peopling of the Americas. *Science Advances*, 2(4):e1501385, 2016.

[49] Arong Luo, Huijie Qiao, Yanzhou Zhang, Weifeng Shi, Simon YW Ho, Weijun Xu, Aibing Zhang, and Chaodong Zhu. Performance of criteria for selecting evolutionary models in phylogenetics: a comprehensive study based on simulated datasets. *BMC Evolutionary Biology*, 10(1):242, 2010.

[50] Anna-Sapfo Malaspinas, Michael C Westaway, Craig Muller, Vitor C Sousa, Oscar Lao, Isabel Alves, Anders Bergström, Georgios Athanasiadis, Jade Y Cheng, Jacob E Crawford, et al. A genomic history of Aboriginal Australia. *Nature*, 538(7624):207, 2016.

[51] L. McInnes, J. Healy, and J. Melville. UMAP: Uniform Manifold Approximation and Projection for Dimension Reduction. *ArXiv e-prints*, February 2018.

[52] Leland McInnes, John Healy, Nathaniel Saul, and Lukas Grossberger. UMAP: Uniform Manifold Approximation and Projection. *The Journal of Open Source Software*, 3(29):861, 2018.

[53] David Meyer, Evgenia Dimitriadou, Kurt Hornik, Andreas Weingessel, and Friedrich Leisch. *e1071: Misc Functions of the Department of Statistics, Probability Theory Group (Formerly: E1071), TU Wien*, 2019. R package version 1.7-2.

[54] Bui Quang Minh, Minh Anh Thi Nguyen, and Arndt von Haeseler. Ultrafast approximation for phylogenetic bootstrap. *Molecular Biology and Evolution*, 30(5):1188–1195, 2013.

[55] Vladimir N Minin, Erik W Bloomquist, and Marc A Suchard. Smooth skyride through a rough skyline: Bayesian coalescent-based inference of population dynamics. *Molecular Biology and Evolution*, 25(7):1459–1471, 2008.

[56] Nano Nagle, Mannis Van Oven, Stephen Wilcox, Sheila van Holst Pellekaan, Chris Tyler-Smith, Yali Xue, Kaye N Ballantyne, Leah Wilcox, Luka Papac, Karen Cooke, et al. Aboriginal Australian mitochondrial genome variation–an increased understanding of population antiquity and diversity. *Nature Scientific Reports*, 7:43041, 2017.

[57] Vinod Nair and Geoffrey E Hinton. Rectified linear units improve restricted Boltzmann machines. In *Proceedings of the 27th international conference on machine learning (ICML-10)*, pages 807–814, 2010.

[58] Masatoshi Nei and Wen-Hsiung Li. Mathematical model for studying genetic variation in terms of restriction endonucleases. *Proceedings of the National Academy of Sciences*, 76(10):5269–5273, 1979.

[59] Masatoshi Nei and Fumio Tajima. DNA polymorphism detectable by restriction endonucleases. *Genetics*, 97(1):145–163, 1981.

[60] Lam-Tung Nguyen, Heiko A Schmidt, Arndt von Haeseler, and Bui Quang Minh. IQ-TREE: a fast and effective stochastic algorithm for estimating maximum-likelihood phylogenies. *Molecular Biology and Evolution*, 32(1):268–274, 2014.

[61] Rasmus Nielsen and Montgomery Slatkin. *An introduction to population genetics: theory and applications*. Sinauer Associates Sunderland, MA, 2013.

[62] Kasih Norman, Josha Inglis, Chris Clarkson, J Tyler Faith, James Shulmeister, and Daniel Harris. An early colonisation pathway into northwest Australia 70-60,000 years ago. *Quaternary Science Reviews*, 180:229–239, 2018.

[63] James F O'Connell, Jim Allen, Martin AJ Williams, Alan N Williams, Chris SM Turney, Nigel A Spooner, Johan Kamminga, Graham Brown, and Alan Cooper. When did *Homo sapiens* first reach Southeast Asia and Sahul? *Proceedings of the National Academy of Sciences*, 115(34):8482–8490, 2018.

[64] E. Paradis. pegas: an R package for population genetics with an integrated–modular approach. *Bioinformatics*, 26:419–420, 2010.

[65] E. Paradis and K. Schliep. ape 5.0: an environment for modern phylogenetics and evolutionary analyses in R. *Bioinformatics*, 35:526–528, 2018.

[66] F. Pedregosa, G. Varoquaux, A. Gramfort, V. Michel, B. Thirion, O. Grisel, M. Blondel, P. Prettenhofer, R. Weiss, V. Dubourg, J. Vanderplas, A. Passos, D. Cournapeau, M. Brucher, M. Perrot, and E. Duchesnay. Scikit-learn: Machine learning in Python. *Journal of Machine Learning Research*, 12:2825–2830, 2011.

[67] Allison Piovesan, Maria Chiara Pelleri, Francesca Antonaros, Pierluigi Strippoli, Maria Caracausi, and Lorenza Vitale. On the length, weight and GC content of the human genome. *BMC research notes*, 12(1):106, 2019.

[68] Oliver G Pybus, Andrew Rambaut, and Paul H Harvey. An integrated framework for the inference of viral population history from reconstructed genealogies. *Genetics*, 155(3):1429–1437, 2000.

[69] R Core Team. *R: A Language and Environment for Statistical Computing*. R Foundation for Statistical Computing, Vienna, Austria, 2019.

[70] Ryan J Rabett. The success of failed *Homo sapiens* dispersals out of Africa and into Asia. *Nature Ecology & Evolution*, 2(2):212–219, 2018.

[71] Andrew Rambaut, Alexei J Drummond, Dong Xie, Guy Baele, and Marc A Suchard. Posterior summarization in Bayesian phylogenetics using Tracer 1.7. *Systematic Biology*, 67(5):901, 2018.

[72] David Reich, Richard E Green, Martin Kircher, Johannes Krause, Nick Patterson, Eric Y Durand, Bence Viola, Adrian W Briggs, Udo Stenzel, Philip LF Johnson, et al. Genetic history of an archaic hominin group from Denisova Cave in Siberia. *Nature*, 468(7327):1053, 2010.

[73] David E Rumelhart, Geoffrey E Hinton, and Ronald J Williams. Learning internal representations by error propagation. Technical report, California University San Diego, La Jolla Institute for Cognitive Science, 1985.

[74] Sriram Sankararaman, Swapan Mallick, Nick Patterson, and David Reich. The combined landscape of Denisovan and Neanderthal ancestry in present-day humans. *Current Biology*, 26(9):1241–1247, 2016.

[75] Gideon Schwarz et al. Estimating the dimension of a model. *The annals of statistics*, 6(2):461–464, 1978.

[76] Montgomery Slatkin. Rare alleles as indicators of gene flow. *Evolution*, 39(1):53–65, 1985.

[77] Pedro Soares, Alessandro Achilli, Ornella Semino, William Davies, Vincent Macaulay, Hans-Jürgen Bandelt, Antonio Torroni, and Martin B Richards. The archaeogenetics of Europe. *Current Biology*, 20(4):R174–R183, 2010.

[78] Julien Soubrier, Mike Steel, Michael SY Lee, Clio Der Sarkissian, Stéphane Guindon, Simon YW Ho, and Alan Cooper. The influence of rate heterogeneity among sites on the time dependence of molecular rates. *Molecular Biology and Evolution*, 29(11):3345–3358, 2012.

[79] Nitish Srivastava, Geoffrey Hinton, Alex Krizhevsky, Ilya Sutskever, and Ruslan Salakhutdinov. Dropout: a simple way to prevent neural networks from overfitting. *The journal of Machine Learning Research*, 15(1):1929–1958, 2014.

[80] Korbinian Strimmer and Oliver G Pybus. Exploring the demographic history of DNA sequences using the generalized skyline plot. *Molecular Biology and Evolution*, 18(12):2298–2305, 2001.

[81] Fumio Tajima. Statistical method for testing the neutral mutation hypothesis by DNA polymorphism. *Genetics*, 123(3):585–595, 1989.

[82] Koichiro Tamura and Masatoshi Nei. Estimation of the number of nucleotide substitutions in the control region of mitochondrial DNA in humans and chimpanzees. *Molecular Biology and Evolution*, 10(3):512–526, 1993.

[83] Simon Tavaré. Some probabilistic and statistical problems in the analysis of DNA sequences. *Lectures on mathematics in the life sciences*, 17(2):57–86, 1986.

[84] Norman B Tindale. Prehistory of the Aborigines: Some interesting considerations. In *Ecological Biogeography of Australia*, pages 1761–1797, 1981.

[85] Ray Tobler, Adam Rohrlach, Julien Soubrier, Pere Bover, Bastien Llamas, Jonathan Tuke, Nigel Bean, Ali Abdullah-Highfold, Shane Agius, Amy ODonoghue, et al. Aboriginal mitogenomes reveal 50,000 years of regionalism in Australia. *Nature*, 544(7649):180, 2017.

[86] Mannis Van Oven. PhyloTree Build 17: Growing the human mitochondrial DNA tree. *Forensic Science International: Genetics Supplement Series*, 5:e392–e394, 2015.

[87] Guido Van Rossum and Fred L. Drake. *Python 3 Reference Manual*. CreateSpace, Scotts Valley, CA, 2009.

[88] Aki Vehtari, Andrew Gelman, Daniel Simpson, Bob Carpenter, Paul-Christian Bürkner, et al. Rank-normalization, folding, and localization: An improved $\widehat{R}$ for assessing convergence of MCMC. *Bayesian Analysis*, 2020.

[89] W. N. Venables and B. D. Ripley. *Modern Applied Statistics with S*. Springer, New York, fourth edition, 2002. ISBN 0-387-95457-0.

[90] John Wakeley. The effects of subdivision on the genetic divergence of populations and species. *Evolution*, 54(4):1092–1101, 2000.

[91] John Wakeley. *Coalescent Theory: An Introduction*. Macmillan Learning, 2016.

[92] Alfred Russell Wallace. On the physical geography of the Malay Archipelago. *The Journal of the Royal Geographical Society of London*, 33:217–234, 1863.

[93] GA Watterson. On the number of segregating sites in genetical models without recombination. *Theoretical Population Biology*, 7(2):256–276, 1975.

[94] Kira E Westaway, J Louys, R Due Awe, Michael J Morwood, Gilbert J Price, J-x Zhao, Maxime Aubert, R Joannes-Boyau, TM Smith, Matthew M Skinner, et al. An early modern human presence in Sumatra 73,000–63,000 years ago. *Nature*, 548(7667):322–325, 2017.

[95] Sewall Wright. Evolution in Mendelian populations. *Genetics*, 16(2):97, 1931.

[96] Ziheng Yang. Maximum likelihood phylogenetic estimation from DNA sequences with variable rates over sites: approximate methods. *Journal of Molecular Evolution*, 39(3):306–314, 1994.

[97] Ziheng Yang. Maximum likelihood estimation on large phylogenies and analysis of adaptive evolution in human influenza virus A. *Journal of Molecular Evolution*, 51(5):423–432, 2000.

[98] Xiguo Yuan, David J Miller, Junying Zhang, David Herrington, and Yue Wang. An overview of population genetic data simulation. *Journal of Computational Biology*, 19(1):42–54, 2012.

[99] Andrey Zharkikh. Estimation of evolutionary distances between nucleotide sequences. *Journal of Molecular Evolution*, 39(3):315–329, 1994.