Scalable Algorithms for Population Genomic Inference

Sara Sheehan

Electrical Engineering and Computer Sciences
University of California at Berkeley

Technical Report No. UCB/EECS-2015-94
http://www.eecs.berkeley.edu/Pubs/TechRpts/2015/EECS-2015-94.html

May 14, 2015

Scalable Algorithms for Population Genomic Inference

by

Sara Sheehan

A dissertation submitted in partial satisfaction of the

requirements for the degree of

Doctor of Philosophy

in

Computer Science

and the Designated Emphasis

in

Computational and Genomic Biology

in the

Graduate Division

of the

University of California, Berkeley

Committee in charge:

Professor Yun S. Song, Chair
Professor Lior Pachter
Professor Rasmus Nielsen

Spring 2015

Scalable Algorithms for Population Genomic Inference

Abstract

Scalable Algorithms for Population Genomic Inference

by

Sara Sheehan

Doctor of Philosophy in Computer Science

University of California, Berkeley

Professor Yun S. Song, Chair

Since the 1920s, researchers in population genetics have developed mathematical models to explain how a species evolves. With the rise of DNA sequencing over the past decade, we now have the data to use these models to answer real questions in evolutionary biology. However, the sheer amount of data and the time complexity of the models makes inference extremely challenging. Computer science has therefore become an essential tool for bridging theoretical models and modern sequencing data.

In this thesis we present two novel algorithms that make use of DNA sequencing data in a principled yet practical way. The first method estimates the history of effective population sizes of a species using a coalescent hidden Markov model (HMM). Previous coalescent HMMs could only handle a few sequences, since the set of coalescent trees makes the state-space prohibitively large. Our algorithm uses a modified state-space to make inference computationally feasible while still retaining the essential genealogical features of a sample. We apply this algorithm, called diCal, to human data to learn more about major events in human history, such as the out-of-Africa migration. We also provide several extensions to diCal that make the computation faster, more automated, and applicable in a wider variety of scenarios.

The second method is an algorithm for jointly estimating effective population size changes and natural selection. These two factors can leave similar traces in genomic data, and the models that would describe both are computationally intractable. Our method uses a machine learning technique called deep learning to make the inference procedure robust and efficient. Deep learning automatically teases out important features of the data, but previously had not been used in population genetics. We apply this method to African *Drosophila melanogaster* data to jointly infer their population size changes and classify each region of their genome as neutral or under natural selection. We considered three types of selection: hard sweeps, soft sweeps, and balancing selection. To create a sophisticated framework for population genomic inference, in the future it would be promising to combine machine learning algorithms with biologically-inspired coalescent modeling.

Contents

List of Figures

List of Tables

Acknowledgments

When I decided to come to Berkeley for grad school, someone told me that the only downside was that afterward I wouldn't be happy anywhere else. Indeed, my time here has been a wonderful learning experience, and I have many people to thank for that. First, I am extremely fortunate to have had Yun as an adviser. Throughout every step of my PhD, Yun has been dedicated to helping me grow and excel. I am continually amazed by his problem-solving skills and ability to see a solution to a seemingly insurmountable research obstacle. Even when I was discouraged by my research, he helped me step back and gain perspective and focus on a solution. He exemplifies interdisciplinary research and has encouraged me to draw upon a wide variety of tools and techniques when tackling biological problems. I have learned so much from Yun, and am grateful to have had him as an adviser and colleague.

I would like to thank the members of my dissertation committee and qualifying exam committee, whose feedback and advice has been instrumental for my research progress: Rasmus Nielsen, Lior Pachter, Elchanan Mossel, and Ken Wachter. In addition, I would like to thank Anant Sahai, who has taught me so much about pedagogy and creative teaching. His encouragement and support of my career path was extremely valuable to me. Todd Kemp and Michael Sipser both generously advised me and helped prepare me for grad school - their early support led to my decision to come to Berkeley.

I was lucky to have some amazing collaborators during my time at Berkeley, including Kelley Harris, Jack Kamm, Eline Lorenzen, Ma'ayan Bresler, and Andrew Chan. I have also been part of a wonderful research group, whose past and present members include Junming Yin, Wei-Chun Kao, Josh Paul, Anand Bhaskar, Jonathan Terhorst, Matthias Steinrücken, Paul Jenkins, Joyce Liu, Ethan Jewett, Jeff Spence, Jonathan Fischer, Geno Guerra, Shishi Luo, Khanh Dao Duc, and Jane Yu. One of the best things about Berkeley for me has been the extended computational biology community, including CTEG (Center for Theoretical Evolutionary Genomics) and the wider designated emphasis. Brian McClendon and Xuan Quach have really tied this group together and made being a part of computational biology at Berkeley easy, effective, and enjoyable.

My two semesters as a GSI at Berkeley were amazing, and I would like to thank the course instructors (Yun and Anant), fellow GSIs, and students for making these experiences so rewarding for me. The enthusiasm and curiosity of the students at Berkeley helped me discover my own love of teaching. In addition, I would like to thank my undergraduate research students, Yun Park and Robert Huang.

I am thankful to have been in EECS together with some inspiring and supportive friends, including Ma'ayan Bresler, Urmila Mahadev, Vidya Ganapati, Claire Baek, Linda Li, and Genia Vogman. I am forever grateful to my parents, who have always been supportive of my decisions, and who have made so many sacrifices so that I had access to the best education possible. Finally, I would like to thank my fiancé Iain Mathieson, who I met at a population genetics workshop during my first year in grad school. Iain has read everything I've written in grad school, and his detailed comments have greatly improved this thesis. He has been the number one supporter of my career, and for that I will always be grateful.

Chapter 1

Introduction

One of the major aims of evolutionary biology is to understand the development of the diversity of life we see today. We know that much of this phenotypic diversity is caused by differences at the DNA level, and further, we now have the technology to access this information and compare different individuals. The field of population genetics seeks to understand DNA variation at the population level. We aim to build models that will help us understand the past evolutionary events that gave rise to observed genetic variation. We can think of the DNA for an diploid individual as a pair of strings built from the characters A, C, G, T, one string from the mother and one from the father. There are many factors that affect these strings, including mutation, recombination, migration, population splits, natural selection, and population size changes. Each factor is important to model so that we obtain an accurate picture of the forces shaping genetic variation.

In chapter 1, we provide an overview of the two main phenomena investigated in this thesis: population size changes and natural selection. We also describe previous work that aims to quantify these forces. In chapter 2 and chapter 3 of this thesis, we will focus on population size changes. In humans, for example, the population size has varied considerably due to changes in environment, culture, and technology. As humans migrated out of Africa and into the rest of the world, it is likely that small "founder" populations re-expanded into new territories, creating "bottleneck" events. Accurately estimating such population size changes provides a clearer picture of human history, or the history of any other species. For endangered species in particular, accurately monitoring their population sizes through genetics could provide essential information for conservation efforts. In chapter 4, we will incorporate natural selection as well, which will require a different type of modeling. Natural selection and population size changes are often confounded, and failure to separate their effects can lead to misleading results about evolution. In chapter 5, we summarize this thesis and provide some future research directions.

1.1 What is effective population size?

The term population size normally refers to the *census* population size, or the total number of individuals in a population at a given point in time. But in population genetics, we instead think about a different quantity called *effective* population size, usually denoted N_e. Over the years there have been many different definitions of effective population size, but the original intuition is that N_e should only include individuals who would contribute genetic material to the next generation. As first described by Sewall Wright, "Obviously N applies only to the breeding population and not to the total number of individuals of all ages" [117]. Based on his work, the working definition of N_e was the size of an idealized population that would show the same level of x as observed in a natural population, where x is some population genetic parameter of interest. Often x was genetic drift, which describes the random changes in allele frequencies that occur in finite populations. An "idealized" population needs to meet several conditions, including that individuals mate randomly, and all individuals survive until they have a chance to reproduce in the next generation.

Since natural populations do not meet these conditions, N_e almost always differs from the census size, often significantly so. In a randomly mating population where the variance in the number of offspring is high, N_e will usually be less than the census size, since many individuals will not pass on genetic material to the next generation. Another factor that can affect N_e is the sex-ratio between males and females in a diploid population. An unequal ratio will bias N_e towards the size of the less prevalent sex.

1.1.1 Variance effective population size

To quantify how such factors affect N_e, we start by examining a model of evolution called the Wright-Fisher model, after Sewall Wright and Ronald Fisher. In this model, we begin with a population of size $2N$ (the 2 for diploid individuals), and assume generations are discrete. In the next generation, each "child" chooses a "parent" at random from $2N$ individuals. Let us examine the case where we have two alleles, A and B, at a single locus. Let p be the frequency of allele A in the current generation, and p' be the frequency of A in the next generation. The variance of p' in the Wright-Fisher model, and many similar models, is

$$\mathrm{Var}(p') = \frac{p(1-p)}{2N_e} \tag{1.1}$$

[30]. This N_e is the variance effective population size. For the Wright-Fisher model in particular, the number of copies of A in the next generation, X, follows a binomial distribution

$$\mathbb{P}(X = k) = \binom{2N}{k} p^k (1-p)^{2N-k}.$$

If we work out the expectation and variance of p' using this binomial distribution, we get

$$\mathbb{E}(p') = p \quad \text{and} \quad \mathrm{Var}(p') = \frac{p(1-p)}{2N}$$

[30]. Comparing this variance to Equation 1.1, we can see that in the Wright-Fisher model, the census population size N is equal to the (variance) effective population size N_e, making this model one of the few that meets all the idealized population assumptions.

This idea of using the variance of p' can also help us understand how a variable population size affects N_e. If N_i is the effective population size in generation i, for $i = 1, \cdots, G$, then the "average" N_e of the population over these G generations is the harmonic mean

$$N_e = \frac{G}{\sum_{i=1}^{G} \frac{1}{N_i}}. \tag{1.2}$$

Therefore, if there have been times of very low population sizes, N_e will be dominated by these sizes. This formula was first provided by Wright [118], and can be understood using an argument outlined by Gillespie [30]. First consider a Wright-Fisher population that has census size N_1 with probability r and size N_2 with probability $1 - r$. Then the variance is

$$\mathrm{Var}(p') = r\frac{p(1-p)}{2N_1} + (1-r)\frac{p(1-p)}{2N_2} = \frac{p(1-p)}{2}\left(\frac{r}{N_1} + \frac{1-r}{N_2}\right).$$

Comparing this variance to Equation 1.1, we can obtain the variance effective population size

$$N_e = \frac{1}{\frac{r}{N_1} + \frac{1-r}{N_2}},$$

which is the harmonic mean of N_1 and N_2. We can extend this idea to multiple population sizes, each occurring for one generation, to obtain Equation 1.2. Note that so far we have assumed a model with no mutation, so the harmonic mean should not be thought of as a perfect formula for "average" N_e, but as a general guideline.

In the case of an imbalanced sex-ratio, let N_f be the number of females and N_m be the number of males. Then the variance effective population size is

$$N_e = \frac{4N_fN_m}{N_f + N_m}.$$

As described in [30], if N_f and N_m are unequal, there will be a reduction in N_e, but typically not a severe one.

Finally, we can investigate an arbitrary offspring distribution. If σ^2 is the variance in the number of offspring for a single copy of the locus, then

$$\mathrm{Var}(p') = \frac{\sigma^2 p(1-p)}{2N},$$

so $N_e = N/\sigma^2$. This formula helps us see that N_e can be either greater or less than the census population size. In the Wright-Fisher model, $\sigma^2 = 1$, so $N_e = N$. If $\sigma^2 > 1$, then the effective population size will be less than the census size. And conversely, for a limiting case, let the number of offspring of each copy be exactly 1, so $\sigma^2 = 0$. In that case $N_e \to \infty$.

1.1.2 Early estimates of effective population size

The Wright-Fisher model is usually thought of *forward* in time, but we could consider the evolutionary process *backward* in time as well. In this case, we would imagine starting with an effective population of size $2N_e$ at the present time, and following the ancestry of these copies backward through the previous generations. If two copies find a common ancestor at a given generation, we say that they *coalesce*. At a single locus, eventually all the lineages will coalesce to a common ancestor. The time of this event is often called the TMRCA for "time to most recent common ancestor". If we rescale time by $2N_e$ and take the limit as $N_e \to \infty$, we obtain a continuous time process instead of one measured in discrete generations. This type of modeling is called coalescent theory; see Kingman [59] for an initial description of the mathematical process, or Wakeley [112] for a more general introduction.

Rescaling by N_e leads many population genetic parameters to depend on the effective population size. For example, if t is time measured in "coalescent units", then $2N_e t$ is time in generations. If we further multiply by the generation time g (years/generation), we can get time in years. In the case of a sample of size $n = 2$ (i.e. two haplotypes or one diploid individual), the expected time to coalescence under neutrality is 1 in coalescent units (exponential waiting time with rate 1), or $2N_e$ generations.

This observation leads to a simple way to estimate N_e. If the per-generation, per-locus (a locus could be one base, or a collection of adjacent bases) mutation rate is μ, then the expected number of pairwise differences between two samples is $4N_e\mu$. This is because mutations occur as a Poisson process on both lineages until the common ancestor event. Averaging across many pairs of individuals, we can obtain the average pairwise heterozygosity, denoted π. Early estimates of N_e thus relied on the formula:

$$\pi = 4N_e\mu. \tag{1.3}$$

In 1974, initial estimates of π (which were denoted M) for different human populations were calculated in [80] using protein polymorphisms. These estimates were later used in [79] to estimate N_e for all humans at around 10,000. See chapter 4 of [112] for a more detailed description of π and other measures of neutral diversity. Subsequent analysis was often consistent with this estimate of N_e, for example, Table 1 of [106] analyzes three methods, of which the latter two use variants of Equation 1.3. The ranges for these estimates are 9,000-12,000, 8,000-11,000, and 7,000-9,200 for all humans.

It should be noted that many assumptions go into this simple method for estimating N_e, most importantly a neutral coalescent (no natural selection) and that N_e has been constant since the TMRCA of the sample. Population size changes are reduced to a single number (roughly the harmonic mean, Equation 1.2). Selection can drastically change the shape of a coalescent topology, creating a potentially significant bias when estimating N_e using this type of method. In particular, protein-coding regions are likely under purifying selection, which could bias N_e estimates downward.

In 1995, Hammer [42] also obtained an estimate of (autosomal) $N_e = 10,000$ for humans using data from the Y chromosome. Not all estimates have been consistent though, and

some debate occurred surrounding an estimate of long-term $N_e = 100,000$ in [3], which was criticized in [24] in favor of $10,000$. This traditional, lower estimate has been used to argue for the single origin of modern humans in Africa. The reasoning is that if N_e is still relatively low in non-African populations, they must have undergone a relatively recent bottleneck, presumably the out-of-Africa bottleneck. If N_e is found to be high across all human populations, that could indicate more long term structure and sub-division among human populations. This debate might seem outdated now, but shows how interested researchers were in N_e and how much weight has been given to its estimates. Later estimates of long-term N_e included 18,000 in [104], and as low as 7,500 for an African population and 3,100 for Asians in [110]. A natural next step toward building a consistent species history is to model changes in effective population size.

1.1.3 Coalescent effective population size

A more mathematical definition of N_e than the number of breeding individuals is as a scaling factor in the rate of coalescence. During times of low population size, there are fewer individuals to coalesce with, so coalesces happens more quickly than expected. Correspondingly, during times of large population size, coalescence happens more slowly than expected. One way to relate this *coalescent* N_e to variance N_e is through the formula $N_e = N/\sigma^2$ we saw before. If each lineage produces exactly one offspring and thus $\sigma^2 = 0$, then going backward in time, there are never any coalescent events. Since the rate of coalescence scales inversely with the effective population size, N_e is infinite, as is consistent with the formula.

To make this more precise, we can quantify how this scaling factor affects the coalescent. First, define a population size change function

$$N(t) = \lambda(t) N_{\text{ref}}$$

where N_{ref} is a baseline effective population size, and $\lambda(t)$ is the function of scaling factors going back in time (so $t = 0$ is the present). N_{ref} can be chosen arbitrarily, and it is not necessary that $\lambda(0) = 1$. Let T_i be the time to the next coalescence event when there are currently i lineages. We can consider the effect of $N(t)$ on the distribution of the coalescence time for two lineages (T_2), which is

$$f_{T_2}(t) = \frac{1}{\lambda(t)} \exp\left(\int_0^t \frac{1}{\lambda(\tau)} d\tau \right). \tag{1.4}$$

Note that generalizing this coalescent process to n lineages is not straightforward, since when N_e is not constant, T_i depends on the sum of $T_n, \cdots, T_{i+1}$.

Coalescent N_e is further confounded by population structure, which can cause lineages from the same population to coalesce more recently, and lineages from different populations to coalesce more anciently. Migration and natural selection also affect this scaling factor. Additionally, for species that undergo genetic recombination, the coalescent history varies throughout the genome. Modeling this process is the focus of the next section.

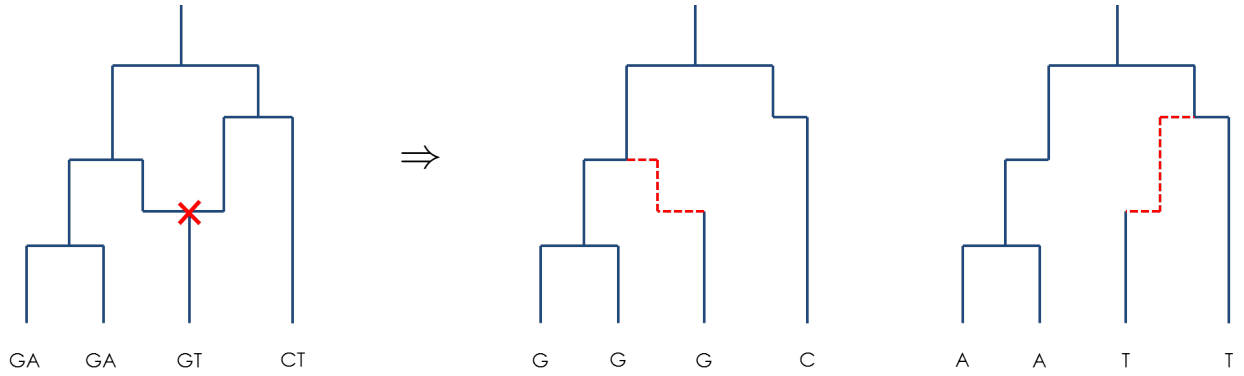


Figure 1.1: The topology on the left is an ancestral recombination graph for four haplotypes at two sites, whose sequences are labeled below each lineage. The red 'X' marks a recombination event between these two sites. The trees on the right show the topologies for the first site and second site, which are different. These are what we refer to as *local trees*.

1.2 Coalescent hidden Markov models

For many problems in population genetics, knowing the true genealogical history of a sample would provide access to a direct estimation of many parameters of interest. For a single locus, the genealogical history is a tree, but for species that undergo recombination, this tree could be different at different sites along the genome. During recombination, the two copies of each chromosome from the parents combine to create a hybrid that can be passed on to the next generation. Recombination "breakpoints" represent switches in the ancestry of the sequence. Previously, we always considered the number of lineages to be decreasing backward time time, and lineages coalesced as they found common ancestors. But with recombination, the number of lineages can *increase*, as breakpoints cause a single lineage to split in two, with each part of the sequence following a different lineage backward in time. Thus the genealogy is no longer a tree, but a graph, which we refer to as the *ancestral recombination graph*, or ARG. An example is illustrated in Figure 1.1.

We can model the ARG using the coalescent with recombination, a stochastic process that encapsulates the history of a collection of DNA sequences [51, 116]. The ancestral state associated with each genetic locus is marginally a tree with time-weighted edges, and neighboring trees in the sequence are highly correlated with each other. Sequential changes in tree structure reflect the process of genetic recombination that slowly breaks up ancestral haplotypes over time.

This view of genealogical history leads to a natural interpretation of sequence data through a graphical model, where the observations are the DNA sequences, and the hidden state is the genealogy relating them. The correlation of nearby trees further provides inspiration for using a hidden Markov model (HMM) as the graphical model of choice. The hidden state is not the entire ARG, but a tree at each locus. In practice we usually assume

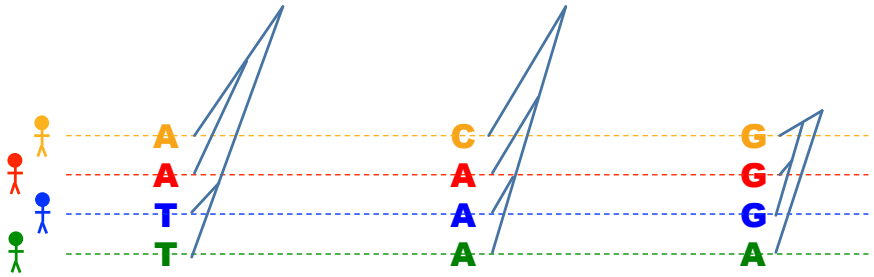


Figure 1.2: Illustration of how the local tree changes along the genome due to recombination. From the first tree to the second tree, there is a recombination event on the red lineage, and then it reconnects to a different part of the tree. From the second to the third tree, a more complicated recombination event occurred, and the tree changed more drastically.

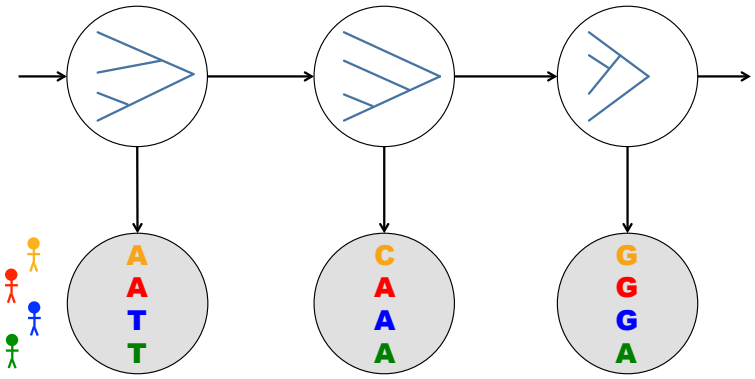


Figure 1.3: Idealized graphical model for a coalescent HMM. The observed data are the alleles at each locus (shaded circles) and the hidden state is a complete, edge-weighted tree (unshaded circles).

that this hidden tree only depends on the tree immediately preceding it (a Markov process of order 1). See Figure 1.2 for an illustration. In a graphical model framework, this process would look like Figure 1.3, which can be thought of as a "gold-standard" coalescent HMM.

However, even with the simplification of a tree as our hidden state, the state space grows super-exponentially in the number of sequences, which is not practical. This can be seen as follows. For n sequences, there are $\binom{n}{2}$ choices for the first coalescent event, $\binom{n-1}{2}$ for the second coalescent event, and so on. So there are

$$\frac{n!(n-1)!}{2^{n-1}} \sim O(n^n) \tag{1.5}$$

distinct tree topologies where we consider the order of internal nodes. Even for $n = 10$, there would be over 2 billion states. This does not even include edge-weighted trees, which we need for time information, but would further exacerbate the state space explosion. This model is not computationally practical, and it is difficult to see how it could be in the future.

As a consequence, much of the modeling work on coalescent HMMs has focused on reducing this state space. In the next section, we will cover three coalescent HMMs that have various solutions to the state space problem. We will also cover how these and other methods have estimated demographic history parameters.

1.3 Previous methods for demographic inference

Much can be learned about ancient population history from present-day DNA data. This might seem surprising, but intuitively, each of our genomes is made up of the genomes of our ancestors. Over many generations, scattered mutation and recombination events make a modern genome look like an "imperfect mosaic" of the ancestral genomes of the past. Demographic history influences the pattern of genetic variation in a population, thus genomic data from multiple individuals sampled from a present-day population contain valuable information about the past demographic history of that population. It is worth asking exactly how changes in N_e can inform us about changes in census population size. For the purposes of this thesis, we will assume that such changes are correlated, and that relative changes in N_e are indicative of relative changes in census population size.

Accurately inferring the past demographic changes in humans has several important applications, including properly accounting for population structure in association studies and reducing confounding effects in inferences about natural selection. It can also help to resolve archaeological and historical questions. Scally and Durbin [101] provide a summary of estimates of the times of migration events out-of-Africa, as well as their relationship to ancient archaeological sites. The *times* of such events have been estimated from modern-day genomic data (as well as ancient DNA and archaeological evidence), but there is less known about the *sizes* of the populations that were migrating at these times.

The ancient effective population size estimates we do have vary widely, as do the time estimates of demographic events. For example, estimates of the population divergence time between European and African humans range from 50 to 120 thousand years ago (kya); see [95] for a review of human demographic results. And humans are far from the only species for which demography raises important questions. The demographic history of *Drosophila* has been investigated and debated, and likely has very interesting dynamics [5, 40, 111, 114], including seasonal population size oscillations. Another example is the speciation time between polar bears and brown bears, where estimates range from 50 kya to 4 million years ago [9, 41, 69, 76].

1.3.1 PSMC: pairwise sequentially Markovian coalescent

PSMC is a coalescent HMM for a pair of chromosomes (or one diploid individual) [64]. In this case, the local tree (i.e. hidden state) is completely described by the coalescence time of the two lineages. To make this one-dimensional state space feasible in an HMM framework, time is discretized into a finite number of intervals. The PSMC model is used to estimate an arbitrary piecewise-constant effective population size history. Within each time interval, the population size is constant, although the population size can be constrained to be the same for multiple consecutive intervals. Assuming such a piecewise-constant size history makes the integral in Equation 1.4 much easier to work with.

Like all coalescent HMMs, the emissions for PSMC are the observed data. For two sequences, this simplifies to whether the observed genotype is homozygous or heterozygous

(phasing is not necessary). As one moves along the sequence, the distribution of homozygous and heterozygous sites is informative of the distribution of coalescence times. Intuitively, during stretches with more heterozygous sites, more mutations have accumulated, so the coalescent times will generally be higher. PSMC models coalescent times using the sequentially Markov coalescent [72, 74] for a pair of sequences. The HMM decoding produces an estimate of these coalescent times, which depend on the past population sizes.

While this elegant approach produces reasonably accurate population size estimates overall, its accuracy in the very recent past is hampered by the fact that, because of the small sample size, few coalescence events occur in that period. As a consequence, the information in the pattern of genetic variation for a pair of sequences is insufficient to resolve very recent demographic history. However, PSMC has many advantages; it is very fast and easy to use. Runtime can easily be controlled by choosing fewer discretization intervals or analyzing subsets of sequences. Multiple diploid individuals can be used pairwise to get a better distribution of coalescent events throughout the mid-ancient past. PSMC has been used to infer ancestral population sizes in a variety of non-model organisms where the number of genomes is limited and phasing is often not possible: wild boar [36], polar and brown bear [76], horse [86], Chinese alligator [113], giant panda [120], as well as Neanderthal and Denisovan archaic hominid genomes [75].

1.3.2 CoalHMM

For more than two sequences, CoalHMM is the purest implementation of the graphical model in Figure 1.3. Originally CoalHMM was used for four species: human (H), chimp (C), gorilla (G), and orangutan [47], as opposed to individuals from the same population or species. With four sequences, the number of coalescent topologies is 18 (Equation 1.5), but in [47] this is restricted by assuming orangutan is always an outgroup. Further, there are four total states: HG coalesce first, CG coalesce first, and HC coalesce first, with separate states for this last event being before or after the HC/G split. This type of state space is possible because the phylogenetic tree (species tree) is known. For example, a tree where human and orangutan coalesce first can be excluded, since this event would very unlikely given the species tree. We would not typically be able to restrict the state space in this way when using samples from the same population. Using this coalescent HMM, four parameters are inferred: HC and HCG ancestral effective population sizes, the HC/G split time, and the H/C split time.

In Dutheil et al. [22], CoalHMM was used for the same data, but was reparametrized to account for constraints on the parameters due to their relationship with the HMM probabilities (and thus the HMM likelihood). Additionally, they use a model that allows for mutation rate heterogeneity along the genome. These changes improved the inference results and also allowed them to estimate lineage-specific recombination rates.

This line of research carried on to create a model for two species (one sequence from each), with more flexibility in the coalescence times by discretizing time into intervals [71]. In some ways this model is similar to PSMC, but the state space and transitions are more complex, and the ultimate goals are different. The parameters inferred are the ancestral population

size, the divergence time, and the recombination rate. Parameter inference improved when a larger state space was used (i.e. more time intervals). Unlike PSMC, which analyzed one diploid individual, this version of CoalHMM analyzes reference sequences from each of two species. This two-species version was further extended to incorporate migration [70].

1.3.3 MSMC: multiple sequential Markovian coalescent

Building upon PSMC, MSMC uses a coalescent HMM to model the *first* coalescence event back in time as it changes along the sequence [102]. The first coalescent event is parameterized as a triplet (t, i, j), where i and j are the lineages that first coalesce, and t is their (discretized) coalescence time. This method often works best for four haplotypes, where the first coalescent event is quite informative over a large range of times. Adding more lineages typically makes the first coalescent event happen more recently, thus losing power in the ancient past. MSMC can be used to estimate a piecewise-constant population size history like PSMC, and can further be used to estimate split times between populations or species. These latter estimates can be found by examining the *cross coalescence* rate between the two populations over time.

1.3.4 Site frequency spectrum (SFS) methods and ∂a∂i

Moving away from coalescent HMMs, another class of methods that has been used to estimate demographic parameters are those based on the site frequency spectrum (SFS). For a sample of size n haplotypes, the SFS counts the number of segregating sites where j samples have the derived allele, for $j = 1, \cdots, n$. Which allele is derived is often determined by using an outgroup (for example, chimp is an outgroup for human). If these ancestral alleles are unknown, the SFS can be *folded* such that variants with j or $n - j$ copies are combined, for $j = 1, \cdots, \lfloor n/2 \rfloor$.

Considering the size of genomic data, the SFS is an extremely low-dimensional summary statistic for the data. However, it is very informative for many population genetic parameters of interest because it captures information about the genealogical history of a sample. In the unfolded case, any mutation with j derived alleles occurred on a branch of a local tree that subtended j leaves. The ratio of the SFS value for j versus the SFS for i gives us information about the (average) relative lengths of branches subtending j versus i leaves. As an example, suppose we consider the SFS value for 1. These *private* mutations must have occurred on the branches just above each sample. If we see many more such mutations that expected under neutrality, then the most recent branches will be much longer than expected, possibly indicating recent population growth. One reason the SFS is convenient to work with is that the *expected* SFS under many models is easy to compute or simulate. Likelihood methods based on the SFS are possible and popular.

One useful method based on the SFS is ∂a∂i [38], which infers demographic parameters such as effective population sizes and split times. For each putative demographic model, the expected SFS is computed using a Wright-Fisher model diffusion. This method uses

a composite likelihood approach for parameter estimation, which finds the best fit to the observed SFS assuming all sites are independent. This assumption is equivalent to assuming infinite recombination between any pair of sites. Most SFS methods implicitly make this assumption - is is difficult to incorporate linkage properly without ending up in a coalescent HMM situation such as the methods described above.

In Gravel et al. [32], an extension to ∂a∂i was used on low-coverage whole-genome data and high-coverage exome data to fit a one-bottleneck model followed by exponential growth in European and Asian populations. They estimated that the timing of the out-of-Africa migration was about 51 kya, and that the effective population size in the ancient past was about 7,300, which then increased to about 14,500 around 150 kya.

The SFS is desirable to work with, since it is straightforward to compute and accommodates many individuals easily. Myers, Fefferman, and Patterson [78] investigate the limits of inferring population size changes from the SFS alone and show that two distinct population size histories may yield exactly the same expected SFS. However, such population histories are generally unrealistic, and under mild assumptions, population size change functions are identifiable by the SFS in the limit of a long genome [6]. Presumably identifiability is also provably possible using linkage information, although it is an open question of exactly how this could be shown.

1.3.5 G-PhoCS: generalized phylogenetic coalescent sampler

Another type of demographic inference approach builds upon phylogenetic methods designed for multiple species. In this realm, G-PhoCS [37] is a Bayesian, coalescent-based method for individuals from different populations. This approach uses Markov chain Monte Carlo (MCMC) to integrate over different possible genealogies relating the sampled individuals. It was inspired by the phylogenetic method MCMCcoal [97], but extensive development was required to create a method for multiple populations from the same species, including migration and unphased data.

G-PhoCS is applied to six diploid genomes, each from a different population: Korean, Han Chinese, European, Yoruban, Bantu, and San, with chimp as an outgroup. They analyze 37,574 loci, each of length 1 kb. For computational tractability they assume that loci are independent and that there is no recombination within a locus. They estimate that Eurasians and Africans diverged around 38 to 64 kya, and that the effective population size of humans in the ancient past was about 9,000.

1.3.6 Identity by descent (IBD) and state (IBS) methods

If two individuals share the same allele at a given locus, we say they are identical by state (IBS). If they further share the same alleles over a contiguous stretch of length L, then we say they share an IBS tract of length L. If two individuals have inherited a genomic region from the same recent ancestor with no subsequent recombination, they are identical by descent (IBD), regardless of whether unshared mutations have accumulated on each lineage.

The distribution of lengths of shared IBS or IBD tracts between pairs of unrelated individuals is informative of recent demographic history. Expected IBS or IBD tract length distributions can be computed and fit to empirical distributions. Recently, Palamara et al. [87] used the empirical distribution of IBD sharing in pairs of 500 Ashkenazi Jewish individuals to infer two rapid population expansions separated by a severe founder event over the past 200 generations. This approach first requires the non-trivial step of inferring IBD tracts from data. The accuracy of existing IBD detection methods has not been fully characterized when the population under consideration has undergone a complex demographic history, although diCal-IBD [107] provides a significant improvement in this direction. An IBS tract length distribution is easier to compute from data, but the analytic distribution is more challenging to compute in a coalescent framework (since lack of mutation must be taken into account). However, Harris and Nielsen [43] compute the expected distribution of IBS tract lengths exactly under the SMC. They fit this distribution to the empirical IBS distribution using a likelihood-based approach, and use it to estimate demographic history in humans.

1.3.7 Multiple sequences: all pairs versus full coalescent

For many demographic inference methods, the challenge is extending them to many sequences while still retaining linkage information. It has been suggested that instead of extending a full coalescent model to multiple sequences, one could instead use all *pairs* of sequences. However, the density of coalescent events with two sequences is fundamentally shifted toward the more ancient past and has a high variance. In Figure 1.4, we compare the theoretical distribution of coalescent events in time for different sample sizes. To ensure a distribution focused on the recent past, modeling the coalescent topology for many sequences is necessary.

The methods highlighted in this section were chosen due to their relationship with the methods described in chapter 2 and chapter 3. There are many other related inference methods. Although not an explicit parameter inference method yet, one final method worth mentioning here is ARGweaver [98]. This method iteratively "threads" individuals through an existing sequence of local trees, thus inferring an approximate ARG. This style of approach is moving toward fully principled inference, and will hopefully be used for parameter inference in the future.

1.4 Variable effective population size and natural selection

Many demographic inference methods assume that natural selection will not significantly bias the results. For species such as those in the *Drosophila* genus, where selection is ubiquitous throughout the genome, ignoring the impact of selection can bias demographic inference. For humans, it is unknown exactly how much selection is biasing demographic inference. Jointly estimating population sizes and selection is a very challenging problem, in part because these

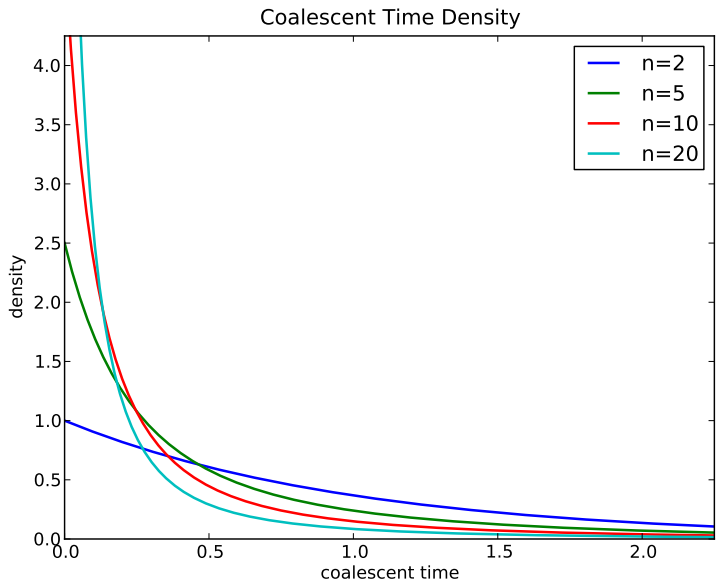


Figure 1.4: The distribution of coalescent events for different sample sizes. We can see that if we want a strong distribution of events in the recent past, a large sample size is needed.

---

factors can sometimes leave similar signatures in the genome. An illustration of this problem in shown in Figure 1.5.

Often a population bottleneck will reduce genetic variation (this can be seen in out-of-Africa populations, which show reduced diversity compared to African populations). However, background/purifying selection also removes genetic variation, and positive selection can do the same thing. In contrast, balancing selection can maintain variation within a population, and exponential growth increases *rare* variation within a population. Separating out all these factors can be extremely difficult. One helpful assumption is that selection acts *locally* at a single locus, whereas demographic changes affect the entire genome in a *global* fashion. This is not necessarily a perfect assumption. For small genomes, it is possible the entire genome is under background selection. Additionally, a large effective population size will make selection more effective, but at the same time, strong selection can reduce the local effective population size. For example, Gossmann, Woolfit, and Eyre-Walker [31] estimate the effective population size locally along the genome, and report that it is correlated with the density of selected sites. See [11] for a review of factors that affect N_e, selection in particular.

When attempting to infer a variable effective population size history, we advocate for estimating a global/average effective population size throughout the genome. If we estimated different population size change histories at different locations throughout the genome, it might be difficult to unify these histories and use them to understand evolutionary events such as bottlenecks. Further, a single region of the genome generally will not contain enough information to infer a multi-parameter effective population size change history.

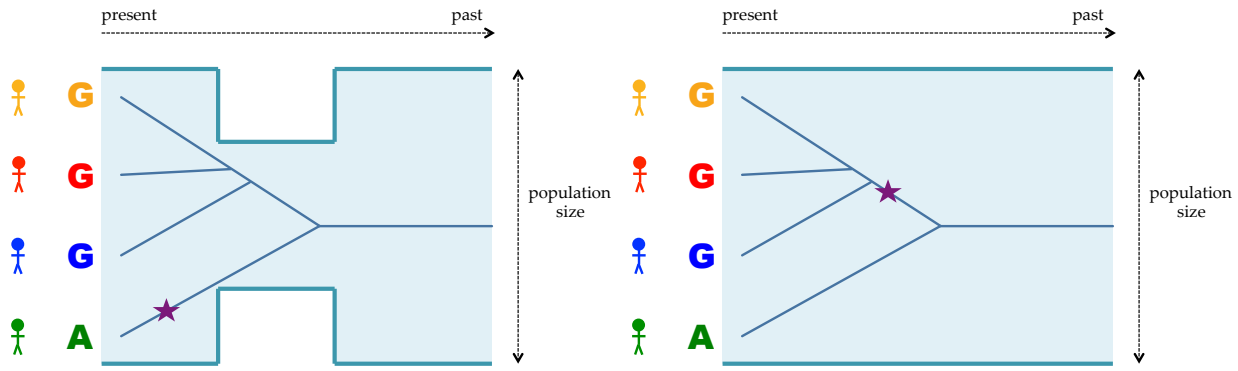


Figure 1.5: Illustration of a bottleneck versus a selected site. In both figures, the ancestral tree at this site is exactly the same. This could be explained by a bottleneck as shown on the left, where the number of lineages increases during the re-expansion. The mutation could have occurred on the branch leading to the green individual: $G \to A$. Alternatively, this tree could be explained by a mutation from $A \to G$ that turned out to be selectively advantageous, as shown on the right. The G allele rose in frequency, which can look like a population expansion.

There is little previous work addressing both demography and selection. Several machine learning methods have been developed to classify regions of the genome into neutral versus selected. The methods in [91], [92], and [100] use support vector machines (SVMs), and the methods in [67] and [68] use boosting. Often these methods demonstrate robustness to different demographic scenarios, but do not explicitly jointly infer demography. Galtier, Depaulis, and Barton [28] develop a likelihood-based method for distinguishing between a bottleneck and positive selection. They apply this method to *Drosophila* to conclude that a series of selective sweeps was more likely than a bottleneck, but do not explicitly infer parameters for both selection and demography.

For a thorough review of this topic, see [65], entitled "Joint analysis of demography and selection in population genetics: where do we stand and where could we go?" One of their conclusions is that a two-step approach is currently the most feasible, where demographic parameters are estimated first using the whole genome and, given this history, outliers are noted as selected sites. If the number of outliers is small, one can assume that the demographic estimates have not been affected too much by selection. If this is not the case, the two-step approach is not as useful. They highlight several methods with promise, but say that so far "None of these methods truly constitutes a joint estimation method of demography and selection."

1.5 Conclusions

In the field of population genetics, much progress has been made toward a unified inference framework, but even separately, effective population size and natural selection inference remain far from solved. One reason that different demographic methods often infer conflicting histories is that they make different trade-offs between the mathematical precision of the model and scalability to larger datasets. This is even true within the class of coalescent HMMs, which are much more similar to each other than to methods that infer demography from summary statistics [38, 43, 87] or MCMC [37]. One issue is the choice of a time discretization, which is often a delicate process. For example, sometimes a shallower, longer bottleneck will fit the data just as well as a more severe, shorter bottleneck.

In chapter 2 of this thesis, we will present a coalescent HMM method called diCal, that estimates variable effective population size for a single population, using an arbitrary number of individuals. The goal of this chapter is to use more individuals to improve inference in the recent past. In chapter 3, several extensions of diCal will be presented, including efforts to make this method faster and more automated. Much of this chapter will focus on the time discretization, both how to incorporate more intervals and how to group intervals effectively. We will also discuss a method that uses diCal to create local trees along the genome.

One difficulty with using coalescent-based methods for inference is that a likelihood (or approximate likelihood) must be computed, which is often very computationally expensive. Approximate Bayesian computation (ABC) is one class of methods that avoids likelihood computation, which will be discussed more in chapter 4. Another class of methods comes from machine learning, which attempts to learn interesting features of a dataset, with little to no prior knowledge of how these features might relate to the components of the original data. In essence machine learning solves an inverse problem, construction a function from the data to parameters of interest.

This is appealing for population genetics, where we also do not always know which features of our data are informative for certain parameters. The SFS has proven to be informative for many parameters, but it is also a dimensionality reduction technique that loses a lot of information. At the same time, plugging our data into a machine learning method in some sense throws away much the coalescent modeling that we know to be biologically inspired. These trade-offs will be discussed in more detail in chapter 4, with some hope for a hybrid solution.

In chapter 4 we will present a tailored *deep learning* method for population genetics. We will demonstrate that deep learning provides some answers to the problems with ABC, and can provide an alternative to traditional likelihood-free inference in population genetics. We apply this method to the problem of jointly inferring a bottleneck and natural selection in *Drosophila* and show that the resulting history is more realistic than those proposed before. Finally, in chapter 5 we will summarize this work and provide some future directions and open challenges that build upon this thesis.

Chapter 2

Effective population size inference from multiple genome sequences

In this chapter, we present a method for estimating an effective population size that varies over time, using genomic sequence data from multiple individuals. Multiple genomes from the same population contain more information about the recent past, but are also more computationally challenging to study jointly in a coalescent framework. Our method can efficiently make use of the information contained in multiple individuals, while retaining the key generality of PSMC in incorporating an arbitrary piecewise-constant population size history. More precisely, the computational complexity of our method depends quadratically in the number of sequences, and the computation involved is parallelized.

As more sequences are considered, we expect to see a larger number of coalescence events during the recent past and should be able to estimate recent population sizes at a higher resolution. With only two sequences, the distribution of coalescence events is shifted toward the ancient past, relative to the distribution of the time a new lineage joins a coalescent tree for multiple sequences. Thus, even if all sequences are considered pairwise, the resolution in the recent past may not be as clear as that achieved by jointly modeling multiple sequences.

The input to our method, which is based on a coalescent HMM, is a collection of haplotype sequences. At present, our method assumes that mutation and recombination rates are given, and it employs the expectation-maximization (EM) algorithm to infer a piecewise-constant history of effective population sizes, with an arbitrary number of change points.

Our work generalizes the recently developed sequentially Markov conditional sampling distribution (SMCSD) framework [90] to incorporate variable population size. This approach provides an accurate approximation of the probability of observing a newly sampled haplotype given a set of previously sampled haplotypes, and it allows one to approximate the joint probability of an arbitrary number of haplotypes. Through a simulation study, we demonstrate that we can accurately reconstruct the true population histories, with a significant improvement over PSMC in the recent past. We also apply our method to the genomes of multiple human individuals of European and African ancestry to obtain a detailed population size change history during recent times. Our software, called *diCal*

(Demographic Inference using Composite Approximate Likelihood), is publicly available at `http://sourceforge.net/projects/dical/`. This chapter follows Sheehan, Harris, and Song [103].

2.1 Notation and a review of the SMCSD framework

Our work stems from the SMCSD framework [90], which describes the conditional genealogical process of a newly sampled haplotype given a set of previously sampled haplotypes. In what follows, we briefly review the key concepts underlying the SMCSD model.

We consider haplotypes each of length L from the same genomic region. Suppose we have already observed n haplotypes $\mathcal{O}_n = \{h_1, \ldots, h_n\}$ sampled at random from a well-mixed population. In this chapter, we use the terms "site" and "locus" interchangeably. Recombination may occur between any pair of consecutive loci, and we denote the set of potential recombination breakpoints by $B = \{(1,2), \ldots, (L-1, L)\}$. Given a haplotype h, we denote the allele at locus ℓ by $h[\ell]$, and the substring $(h[\ell], \ldots, h[\ell'])$ by $h[\ell : \ell']$ (for $\ell \leq \ell'$).

As described in Paul and Song [88], given the genealogy $\mathcal{A}_{\mathcal{O}_n}$ for the already observed sample $\mathcal{O}_n$, it is possible to sample a *conditional genealogy* $\mathcal{C}$ for the additional haplotype according to the following description: An ancestral lineage in $\mathcal{C}$ undergoes mutation at locus ℓ at rate $\theta_\ell/2$ according to the stochastic mutation transition matrix $\mathbf{P}^{(\ell)}$. Further, as in the ordinary coalescent with recombination, an ancestral lineage in $\mathcal{C}$ undergoes recombination at breakpoint $b \in B$ at rate $\rho_b/2$, giving rise to two lineages. Each pair of lineages within $\mathcal{C}$ coalesce with rate 1, and lineages in $\mathcal{C}$ get *absorbed* into the known genealogy $\mathcal{A}_{\mathcal{O}_n}$ at rate 1 for each pair of lineages. See Figure 2.1(a) for an illustration.

Unfortunately, we do not generally have access to the true genealogy $\mathcal{A}_{\mathcal{O}_n}$, and marginalizing over all possibilities is a challenging problem. However, Paul and Song [88] show that the diffusion-generator approximation described in [34, 52, 53] implies the following approximation to $\mathcal{A}_{\mathcal{O}_n}$ which simplifies the problem considerably.

2.1.1 Approximation 1: the trunk genealogy

We approximate $\mathcal{A}_{\mathcal{O}_n}$ by the so-called *trunk* genealogy $\mathcal{A}^*_{\mathcal{O}_n}$ in which lineages do not mutate, recombine, or coalesce with one another, but instead form a non-random "trunk" extending infinitely into the past, as illustrated in Figure 2.1(b). Although $\mathcal{A}^*_{\mathcal{O}_n}$ is not a proper genealogy, it is still possible to sample a well-defined conditional genealogy $\mathcal{C}$ for the additional haplotype given $\mathcal{A}^*_{\mathcal{O}_n}$ in much the same way as described above, except that rates need to be modified. Specifically, lineages within $\mathcal{C}$ evolve backwards in time subject to the following events:

- *Mutation*: Each lineage undergoes mutation at locus ℓ with rate θ_ℓ according to $\mathbf{P}^{(\ell)}$.

- *Recombination*: Each lineage undergoes recombination at breakpoint $b \in B$ with rate ρ_b.
- *Coalescence*: Each pair of lineages coalesce with rate 2.
- *Absorption*: Each lineage is absorbed into a lineage of $\mathcal{A}^*_{\mathcal{O}_n}$ with rate 1.

The genealogical process described above completely characterizes a conditional sampling distribution (CSD), which we denote $\hat{\pi}_{\mathrm{PS}}$ [88]. Observe that the rate of absorption is the same as before, but the rates for mutation, recombination, and coalescence are each a factor of two larger than that mentioned earlier. Intuitively, this rate adjustment accounts for using the (inexact) trunk genealogy $\mathcal{A}^*_{\mathcal{O}_n}$, which remains static. Note that the adjustment follows as a mathematical consequence of the diffusion-generator approximation [34, 52, 53], and it is supported by the fact that the CSD $\hat{\pi}_{\mathrm{PS}}$ has been shown to be exact for a one-locus model with parent-independent mutation [88].

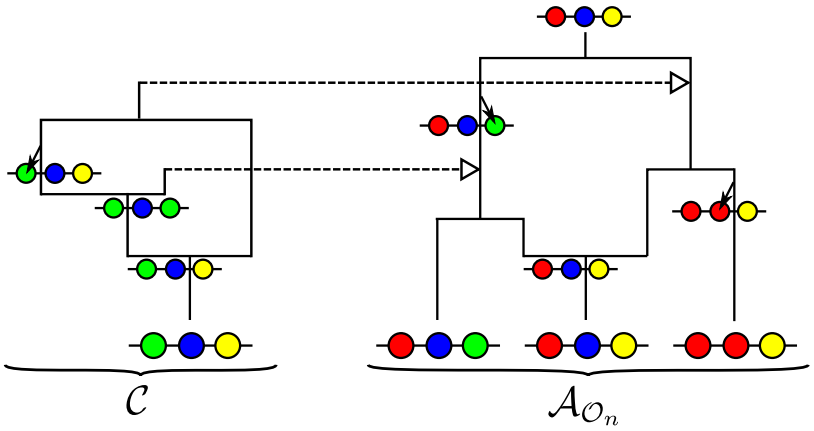
It can be deduced from the diffusion-generator approximation that $\hat{\pi}_{\mathrm{PS}}(\alpha \mid \mathcal{O}_n)$, the conditional probability of sampling an additional haplotype α given a set of previously sampled haplotypes $\mathcal{O}_n$, satisfies a recursion. Unfortunately, this recursion is computationally intractable to solve for even modest-sized datasets. To address this issue, Paul, Steinrücken, and Song [90] proposed further approximations, described below, to obtain a CSD that admits efficient implementation, while retaining the accuracy of $\hat{\pi}_{\mathrm{PS}}$.

2.1.2 Approximation 2: sequentially Markov CSD

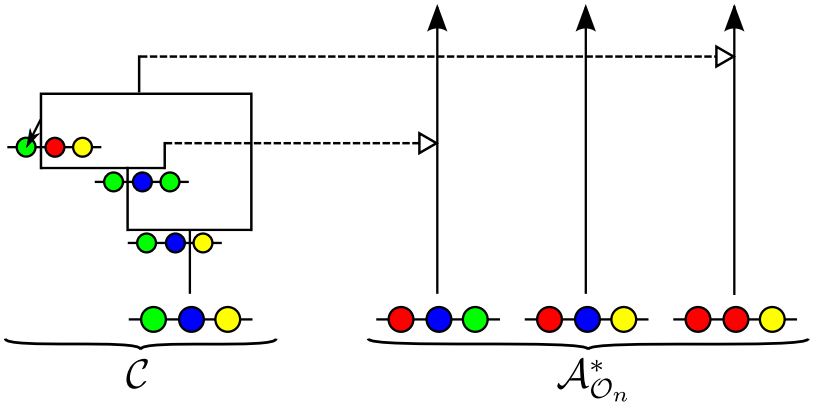
A given conditional genealogy $\mathcal{C}$ contains a *marginal conditional genealogy* (MCG) for each locus, where each MCG comprises a series of mutation events and the eventual absorption into a lineage of the trunk $\mathcal{A}^*_{\mathcal{O}_n}$. See Figure 2.1(c) for an illustration. The key insight is that we can generate the conditional genealogy as a *sequence* of MCGs across the genome, rather than backwards in time [116]. Although the sequential process is actually not Markov, it is well approximated [72, 74, 90] by a Markov process using a two-locus transition density. Applying this approximation to $\hat{\pi}_{\mathrm{PS}}$ yields the *sequentially Markov CSD* $\hat{\pi}_{\mathrm{SMC}}$.

Conditional on the MCG $\mathcal{C}_{\ell-1}$ at locus $\ell-1$, the MCG $\mathcal{C}_\ell$ at locus ℓ can be sampled by first placing recombination events onto $\mathcal{C}_{\ell-1}$ according to a Poisson process with rate $\rho_{(\ell-1,\ell)}$. If no recombination occurs, $\mathcal{C}_\ell$ is identical to $\mathcal{C}_{\ell-1}$. If recombination does occur, $\mathcal{C}_\ell$ is identical to $\mathcal{C}_{\ell-1}$ up to the time T_r of the most recent recombination event. At this point, the lineage at locus ℓ, independently of the lineage at locus $\ell-1$, proceeds backwards in time until being absorbed into a lineage of the trunk. This transition mechanism for the Markov process is illustrated in Figure 2.2. McVean and Cardin [74] use this approximation as well, while the transition process in Marjoram and Wall [72] allows the lineage to coalesce back into itself.

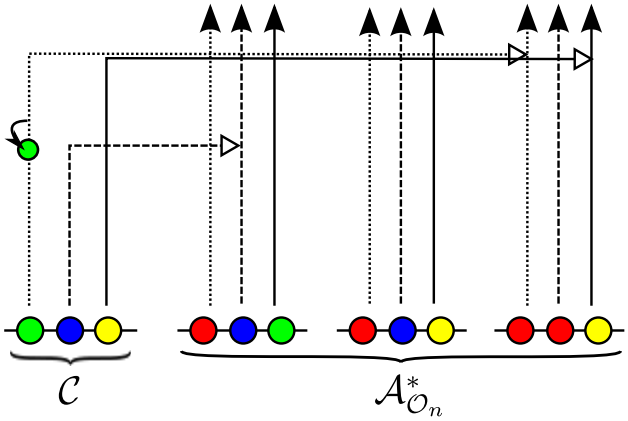
Given $\mathcal{C}_\ell$, mutations are superimposed onto it according to a Poisson process with rate θ_ℓ. The MCG is absorbed into a trunk lineage corresponding to some haplotype h, which specifies an "ancestral" allele $h[\ell]$. This allele is then propagated to the present according to the superimposed mutations and the transition matrix $\mathbf{P}^{(\ell)}$, thereby generating an allele



(a) The true genealogy $\mathcal{A}_{\mathcal{O}_n}$ for the already observed sample $\mathcal{O}_n$.



(b) Approximation by the trunk genealogy $\mathcal{A}^*_{\mathcal{O}_n}$; lineages in the trunk do not mutate, recombine, or coalesce.



(c) *Marginal conditional genealogy* for each locus.

Figure 2.1: Illustration of a conditional genealogy $\mathcal{C}$ for a three-locus model. The three loci of a haplotype are each represented by a solid circle, with the color indicating the allelic type at that locus. Mutation events, along with the locus and resulting haplotype, are indicated by small arrows. Recombination events, and the resulting haplotype, are indicated by branching events. Absorption events are indicated by dotted horizontal lines.

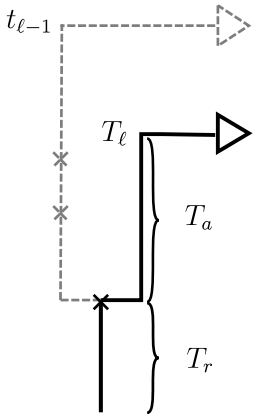


Figure 2.2: Illustration of the sequentially Markov approximation in which the absorption time T_ℓ at locus ℓ is sampled conditionally on the absorption time $T_{\ell-1} = t_{\ell-1}$ at the previous locus. In the marginal conditional genealogy $\mathcal{C}_{\ell-1}$ for locus $\ell - 1$, recombination breakpoints are realized as a Poisson process with rate $\rho_{(\ell-1,\ell)}$. If no recombination occurs, $\mathcal{C}_\ell$ is identical to $\mathcal{C}_{\ell-1}$. If recombination does occur, as in the example here, $\mathcal{C}_\ell$ is identical to $\mathcal{C}_{\ell-1}$ up to the time T_r of the most recent recombination event. At this point, the lineage at locus ℓ, independently of the lineage at locus $\ell - 1$, proceeds backwards in time until being absorbed into a lineage of the trunk. The absorption time at locus ℓ is $T_\ell = T_r + T_a$, where T_a is the remaining absorption time after the recombination event.

---

at locus ℓ of the additional haplotype α. We refer to the associated distribution of alleles as the emission distribution.

The generative process described above for the SMCSD $\hat{\pi}_{\text{SMC}}$ can be formulated as an HMM, in which the hidden state at locus ℓ corresponds to the MCG $\mathcal{C}_\ell$ excluding mutation events: we denote the hidden state at locus ℓ in the HMM by $S_\ell = (T_\ell, H_\ell)$, where $T_\ell \in [0, \infty)$ is the absorption time and $H_\ell \in \mathcal{O}_n$ is the absorption haplotype. The emission at locus ℓ corresponds to the allele $\alpha[\ell]$. See [90] for explicit expressions for the initial, transition, and emission densities in the case of a constant population size.

2.2 Incorporating variable population size

Here, we extend the SMCSD framework described in the previous section to incorporate variable population size. A history of relative effective population size is described by the function

$$\lambda(t) = \frac{N(t)}{N_{\text{ref}}}, \tag{2.1}$$

where $t \in [0, \infty)$, with $t = 0$ corresponding to the present time, N_{ref} is some reference effective population size, and $N(t)$ is the effective population size at time t in the past. The population-scaled recombination and mutation rates are defined with respect to N_{ref}.

Specifically, for $b = (\ell-1, \ell)$, we define $\rho_b = 4N_{\text{ref}}r_b$, where r_b denotes the recombination rate per generation per individual between loci $\ell - 1$ and ℓ; and $\theta_\ell = 4N_{\text{ref}}\mu_\ell$, where μ_ℓ denotes the mutation rate per generation per individual at locus ℓ.

2.2.1 Initial density

In the case of a constant population size, the absorption time T_ℓ for locus ℓ follows an exponential distribution, but with a variable population size the absorption time is described by a non-homogeneous Markov chain. See [35] for a more thorough discussion of the coalescent with variable population size. As in the constant population size case, however, the prior distribution of absorption haplotype H_ℓ is still uniform over the observed haplotypes $\mathcal{O}_n$ in the trunk genealogy. In summary, the marginal density of the hidden state $s_\ell = (t, h)$ is given by

$$\zeta^{(\lambda)}(t, h) = \frac{n_h}{\lambda(t)} \exp\left(-n \int_0^t \frac{1}{\lambda(\tau)} \, \mathrm{d}\tau\right), \tag{2.2}$$

where n_h denotes the number of haplotypes in $\mathcal{O}_n$ that are identical to haplotype h (for a sufficiently large region, usually $n_h = 1$).

2.2.2 Transition density

To obtain the transition density, we need to take into account recombination, which causes changes in the hidden state of our HMM. If no recombination occurs between loci $\ell-1$ and ℓ (prior to $T_{\ell-1}$), then $s_\ell = s_{\ell-1}$. If a recombination event occurs between loci $\ell-1$ and ℓ, the absorption time for locus ℓ will be $T_\ell = T_r + T_a$, where T_r is the time of recombination (which must be less than $T_{\ell-1}$ and T_ℓ) and T_a is the remaining additional time to absorption, as illustrated in Figure 2.2. To compute the transition density, we need to convolve the hidden variables T_r and T_a. Letting $b = (\ell - 1, \ell)$ for ease of notation, the transition density from $s_{\ell-1} = (t, h)$ to $s_\ell = (t', h')$ is given by

$$\phi^{(\lambda)}(s_\ell | s_{\ell-1}) = e^{-\rho_b t} \cdot \delta_{s_{\ell-1}, s_\ell} + \int_0^{\min(t,t')} \rho_b e^{-\rho_b t_r} \left[\frac{\zeta^{(\lambda)}(t', h')}{\int_{t_r}^\infty \zeta^{(\lambda)}(\tau) \, \mathrm{d}\tau}\right] \mathrm{d}t_r, \tag{2.3}$$

where $\zeta^{(\lambda)}(t', h')$ is defined in Equation 2.2 and $\zeta^{(\lambda)}(\tau) := \sum_{h \in \mathcal{O}_n} \zeta^{(\lambda)}(\tau, h)$. Note that $\int_0^\infty \zeta^{(\lambda)}(\tau) \, \mathrm{d}\tau = 1$.

2.2.3 Emission probability

The probability of emitting allele a at locus ℓ (i.e., $\alpha[\ell] = a$) given hidden state $s_\ell = (t, h)$ is

$$\xi^{(\lambda)}(a | s_\ell) = e^{-\theta_\ell t} \sum_{m=0}^{\infty} \frac{1}{m!} (\theta_\ell t)^m \left[(\mathbf{P}^{(\ell)})^m\right]_{h[\ell], a}. \tag{2.4}$$

This is the same emission probability as in [90], but when we discretize the state space in the following section we will have to take into account the effects of variable population size.

2.2.4 The sequentially Markov conditional sampling probability

Using the initial, transition, and emission densities described above, we can write down an integral recursion for the forward probability $f^{(\lambda)}_{\mathrm{SMC}}(\alpha[1:\ell], s_\ell)$ of observing the first ℓ alleles $\alpha[1], \ldots, \alpha[\ell]$ and the state at locus ℓ being s_ℓ. For $2 \leq \ell \leq L$,

$$f^{(\lambda)}_{\mathrm{SMC}}(\alpha[1:\ell], s_\ell) = \xi^{(\lambda)}(\alpha[\ell] \mid s_\ell) \cdot \int \phi^{(\lambda)}(s_\ell | s_{\ell-1}) f^{(\lambda)}_{\mathrm{SMC}}(\alpha[1:\ell-1], s_{\ell-1}) \, \mathrm{d}s_{\ell-1}, \tag{2.5}$$

with base case

$$f^{(\lambda)}_{\mathrm{SMC}}(\alpha[1], s_1) = \xi^{(\lambda)}(\alpha[1] \mid s_1) \cdot \zeta^{(\lambda)}(s_1).$$

Finally, the conditional probability of sampling an additional haplotype α having previously observed $\mathcal{O}_n = \{h_1, \cdots, h_n\}$ is given by

$$\hat{\pi}^{(\lambda)}_{\mathrm{SMC}}(\alpha \mid \mathcal{O}_n) = \int f^{(\lambda)}_{\mathrm{SMC}}(\alpha[1:L], s_L) \, \mathrm{d}s_L. \tag{2.6}$$

As with the constant population size HMM, a backward algorithm can also be devised to compute $\hat{\pi}^{(\lambda)}_{\mathrm{SMC}}(\alpha \mid \mathcal{O}_n)$, though we do not present it here.

2.3 Discretizing the state space

To efficiently evaluate the recursion (Equation 2.5) and the marginalization (Equation 2.6), we discretize the time component of the state space. We partition time (in units of $2N_{\mathrm{ref}}$ generations) into d intervals, demarcated by

$$t_0 = 0 < t_1 < \cdots < t_d = \infty,$$

and assume that $\lambda(t)$ defined in Equation 2.1 has a constant value λ_i in each interval $D_i := [t_{i-1}, t_i)$, for $i = 1, \ldots, d$:

$$\lambda(t) = \sum_{i=1}^{d} \mathbf{1}(t_{i-1} \leq t < t_i) \lambda_i, \tag{2.7}$$

where $\mathbf{1}(\cdot)$ is the indicator function. Using this piecewise-constant $\lambda(t)$, we can write the HMM probabilities in a more workable form, as detailed below.

2.3.1 Initial probability

For $t \in D_i$, Equation 2.7 implies that the initial density in Equation 2.2 can be written as

$$\zeta^{(\lambda)}(t, h) = \frac{n_h}{\lambda_i} e^{-n(t - t_{i-1})/\lambda_i} \prod_{j=1}^{i-1} e^{-n(t_j - t_{j-1})/\lambda_j}. \tag{2.8}$$

To obtain the initial probability in the time-discretized model, we integrate over the time interval D_i to obtain

$$\hat{\zeta}^{(\lambda)}(D_i, h) = \int_{D_i} \zeta^{(\lambda)}(t, h)\,\mathrm{d}t = \frac{n_h}{n} w^{(i)}, \tag{2.9}$$

where

$$w^{(i)} = \left[1 - e^{-n(t_i - t_{i-1})/\lambda_i}\right] \prod_{m=1}^{i-1} e^{-n(t_m - t_{m-1})/\lambda_m},$$

which corresponds to the probability that a lineage in the conditional genealogy gets absorbed into the trunk genealogy within the interval D_i.

2.3.2 Transition probability

For the transition density from state $s_{\ell-1} = (t, h)$ to state $s_\ell = (t', h')$, we let i denote the time interval index such that $t \in D_i = [t_{i-1}, t_i)$ and let j denote the index such that $t' \in D_j = [t_{j-1}, t_j)$. After some simplification, the transition density (Equation 2.3) becomes

$$\phi^{(\lambda)}(s_\ell \mid s_{\ell-1}) = e^{-\rho_b t} \cdot \delta_{s_{\ell-1}, s_\ell} + \frac{n_h}{\lambda_j} e^{-n(t' - t_{j-1})/\lambda_j} \left[\prod_{m=1}^{j-1} e^{-n(t_m - t_{m-1})/\lambda_m}\right] R(i, t; j, t'), \tag{2.10}$$

where $R(i, t; j, t')$ is defined in Appendix A.

To compute the transition probability in the time-discretized model, we use Bayes' rule and integrate the transition density function to obtain

$$\begin{aligned} \hat{\phi}^{(\lambda)}(D_j, h' \mid D_i, h) &= \frac{1}{\hat{\zeta}^{(\lambda)}(D_i, h)} \int_{D_j} \int_{D_i} \phi^{(\lambda)}(t', h' \mid t, h) \zeta^{(\lambda)}(t, h)\,\mathrm{d}t\,\mathrm{d}t' \\ &=: y^{(i)} \cdot \delta_{i,j} \delta_{h,h'} + z^{(i,j)} \cdot \frac{n_{h'}}{n}, \end{aligned} \tag{2.11}$$

where $\hat{\zeta}^{(\lambda)}(D_i, h)$ is defined in Equation 2.9, and explicit formulas for $y^{(i)}$ and $z^{(i,j)}$ are provided in Appendix A. The first term arises from the case of no recombination, while the second term accounts for the case when recombination does occur. Note that $y^{(i)}$ and $z^{(i,j)}$ depend only on the time interval, not on the absorbing haplotype.

2.3.3 Emission probability

Although thus far the emission density has not been affected by the population size being variable, discretizing time introduces a dependence on the function $\lambda(t)$. Let a denote the emitted allele of the newly sampled haplotype α at locus ℓ. Using Bayes' rule again and then integrating over the absorption time interval gives

$$\hat{\xi}^{(\lambda)}(a | D_i, h) = \frac{1}{\hat{\zeta}^{(\lambda)}(D_i, h)} \int_{D_i} \xi^{(\lambda)}(a | t, h) \zeta^{(\lambda)}(t, h)\ \mathrm{d}t = \sum_{m=0}^{\infty} v^{(i)}(m) \cdot [(\mathbf{P}^{(\ell)})^m]_{h[\ell], a}, \tag{2.12}$$

where a formula for $v^{(i)}(m)$ is provided in Appendix A.

2.3.4 Discretizing time and grouping parameters

To discover periods of population expansion or contraction with the SMCSD, it is necessary to specify a time discretization that has high resolution during such time periods. This is challenging in cases where we have little *a priori* knowledge of the demographic history. Ideally the (unknown) coalescence events would be distributed uniformly across the time intervals of our discretization; if very few coalescence events occur in an interval, the corresponding population size will often be overestimated, leading to run-away behavior. In our implementation, we employ a heuristic method, detailed in Appendix A, for choosing the discretization time points $t_1, \ldots, t_{d-1}$ based on the spacing of SNPs in the data, with the aim for coalescence events to be distributed evenly across the d time intervals. Alternatively, the user has the option of specifying their own discretization time points to achieve a desired resolution.

As noted in Li and Durbin [64], allowing separate population size parameters during time intervals that contain too few expected coalescence events can lead to inaccurate estimates. Following their lead, we mitigate this problem by constraining a few consecutive time intervals to have the same population size.

2.4 Modifying the trunk genealogy

The trunk genealogy approximation in [88] was derived by making an approximation in the diffusion process dual to the coalescent for a constant population size. It yields an accurate approximate CSD in the case of a population at equilibrium, and for parent-independent mutation models, the CSD becomes exact in the limit as the recombination rate approaches ∞. However, in the variable population size setting, we must modify the trunk genealogy approximation for the following reason: In the formulation described earlier, the trunk absorbs a lineage in the conditional genealogy $\mathcal{C}$ at the rate $n\mathrm{d}t/\lambda(t)$ at time t. Our HMM uses this inverse dependence and the inferred distribution of absorption times to estimate the population size scaling function $\lambda(t)$. In reality, at time t the number of ancestral lineages is $n(t) \leq n$ and a lineage in $\mathcal{C}$ gets absorbed at rate $n(t)\mathrm{d}t/\lambda(t)$. Hence, assuming that the trunk genealogy contains n lineages in every time interval causes absorption events to occur too quickly, leaving the ancient population sizes over-estimated. We later provide empirical results which support this intuition (see Figure 2.10).

To remedy the problem described above, in our work we use the expected number of lineages in the trunk to modify the rate of absorption, while still forbidding mutation, recombination, and coalescence in the trunk genealogy. Let $A_n(t)$ denote the number of lineages at time t ancestral to a sample of size n at time 0. Under the coalescent, the probability distribution of $A_n(t)$ is known in closed form [108], but using it directly to compute the expected number of lineages leads to numerically unstable results, due to alternating signs. However, one can obtain the following expression for the expectation [108, Equation 5.11]

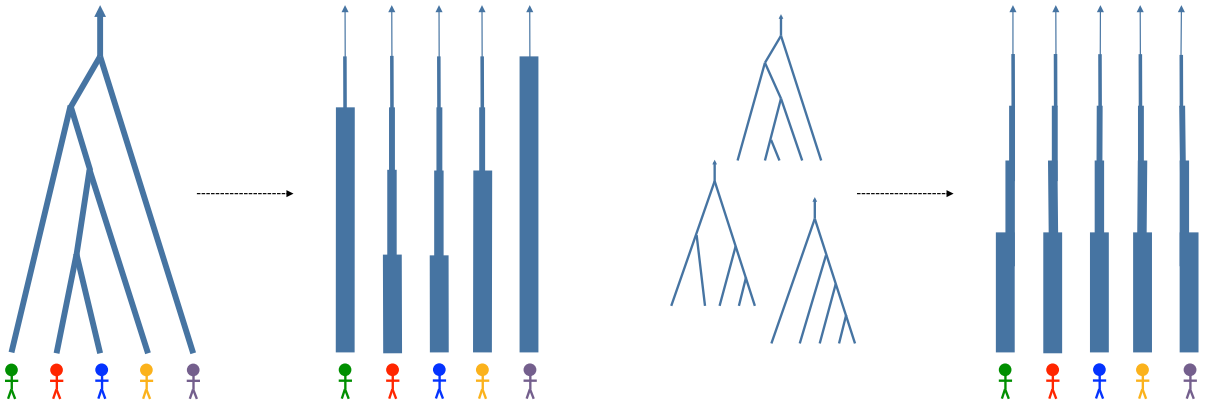


Figure 2.3: Wedding cake genealogy intuition. On the left, we show how an example tree can be thought of as a "trunk" genealogy that has each individual represented for all time. At each point back in time, we can think of the width of the lines as adding up to the total number of remaining lineages at that time (widths not to scale). If we averaged over all possible genealogies and discretized time, we would get the wedding cake genealogy (right).

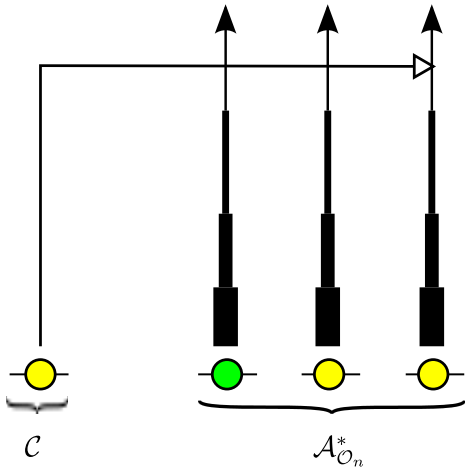


Figure 2.4: Illustration of the wedding cake genealogy approximation, in which the varying thickness of a lineage in $\mathcal{A}^*_{\mathcal{O}_n}$ schematically represents the amount of contribution to the absorption rate. As the figure depicts, the wedding cake genealogy never actually loses any of the n lineages, and absorption into any of the n lineages is allowed at all times; we are only modifying the absorption rate as a function of time.

which is numerically stable:

$$\overline{n}(t) := \mathbb{E}[A_n(t)] = \sum_{i=1}^{n} \exp\left[-\binom{i}{2}\int_0^t \frac{1}{\lambda(\tau)}d\tau\right] \frac{n(n-1)\cdots(n-i+1)}{n(n+1)\cdots(n+i-1)}(2i-1). \quad (2.13)$$

For simplicity, we assume that throughout time interval $D_i = [t_{i-1}, t_i)$, there are $\overline{n}(t_{i-1})$ lineages, creating what we call a "wedding cake genealogy." Figure 2.3 illustrates the relationship of the wedding cake genealogy to an example genealogy, and Figure 2.4 illustrates how a left-out lineages can join the wedding cake genealogy.

To modify the HMM formulas, we simply replace each n in Equations 2.9, 2.11, and 2.12 with the appropriate $\overline{n}(\cdot)$ from Equation 2.13, except in the ratio n_h/n multiplying $w^{(i)}$ in

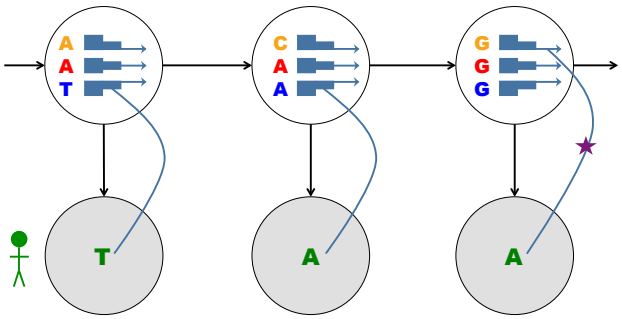


Figure 2.5: An illustration of the diCal HMM. The previously sampled haplotpyes form a wedding cake genealogy at each locus (yellow, red, blue haplotypes). Each individual haplotype is left out in turn (represented by the green individual here), and the hidden state is the time and haplotype that this left-out lineage joins onto.

Equation 2.9 and the ratio $n_{h'}/n$ multiplying $z^{(i,j)}$ in Equation 2.11 (these ratios are kept intact to preserve the relative contributions of different haplotypes). Note that the trunk genealogy never actually loses any of the n lineages, and absorption into any of the n lineages is allowed at all times; we are only modifying the absorption rate as a function of time. In the case of two sequences (one trunk lineage and one additionally sampled lineage), $\overline{n}(t) = 1$ for all t, so the wedding cake approximation does not change the model. Making the number of lineages more accurate using this approximation improves our ability to estimate absorption times and therefore population sizes.

As a final illustration of the method, Figure 2.5 shows how diCal relates to the "gold-standard" HMM from Figure 1.3.

2.5 Population size inference with Expectation-Maximization

To utilize all our data in an exchangeable way, we use a "leave-one-out" approach where we leave each haplotype out in turn and perform the SMCSD computation. More precisely, we define the leave-one-out composite likelihood (LCL) as

$$L_{\text{LCL}}(\lambda; h_1, \ldots, h_n) = \prod_{i=1}^{n} \hat{\pi}^{(\lambda)}_{\text{SMC}}(h_i | h_1, \ldots, h_{i-1}, h_{i+1}, \ldots, h_n). \tag{2.14}$$

Because we compute the conditional sampling probability through dynamic programming and the probability depends on the effective population sizes in complex ways, we cannot find the maximum-likelihood estimates analytically. Although direct optimization could be used, it is computationally expensive. Thus we employ an expectation-maximization (EM) algorithm to estimate the piecewise-constant function $\lambda(t)$. Our current implementation

assumes that the population-scaled recombination rates ρ_b and mutation rates θ_ℓ, as well as the mutation transition matrices $\mathbf{P}^{(\ell)}$, are given and fixed. For computational simplicity we currently assume that θ_ℓ and $\mathbf{P}^{(\ell)}$ are the same for each site ℓ, and ρ_b is the same for each pair of consecutive sites. The time discretization is fixed throughout the EM algorithm. The output of the algorithm is an estimated population size scaling factor λ_i for each interval $D_i = [t_{i-1}, t_i)$. To convert these scaling factors into diploid effective population sizes, one would need to multiply by N_{ref}. Similarly, the discretization times can be converted to years by multiplying them by $2N_{\text{ref}}g$, where g is an average number of years per generation.

The standard Baum-Welch algorithm gives an EM procedure for learning the parameters of an HMM in which the transition probabilities and emission probabilities are treated as unknown independent parameters. However, our HMM is more constrained than a general one, with $(dn)^2+d|\Sigma|^2$ (where Σ is the alphabet of alleles) unknown probabilities $\hat{\phi}^{(\lambda)}(D_j, h' \mid D_i, h)$ and $\hat{\xi}^{(\lambda)}(\alpha[\ell] \mid D_i, h)$ that are functions of the d parameters $\lambda_1, \ldots, \lambda_d$. During the E-step, we compute the matrix $[A_{ij}]$ of the expected number of D_i to D_j transitions. We also compute the matrix $[E_i(b)]$ of the expected number of times allele b is emitted from time interval i. Then, during the M-step we maximize the likelihood function

$$(\lambda_1^{(k+1)}, \ldots, \lambda_d^{(k+1)}) = \underset{\lambda^{(k)}}{\text{argmax}} \prod_{i,j} [\hat{\phi}^{(\lambda^{(k)})}(D_j|D_i)]^{A_{ij}^{(k)}} \prod_{i,b} [\hat{\xi}^{(\lambda^{(k)})}(b|D_i)]^{E_i^{(k)}(b)}, \tag{2.15}$$

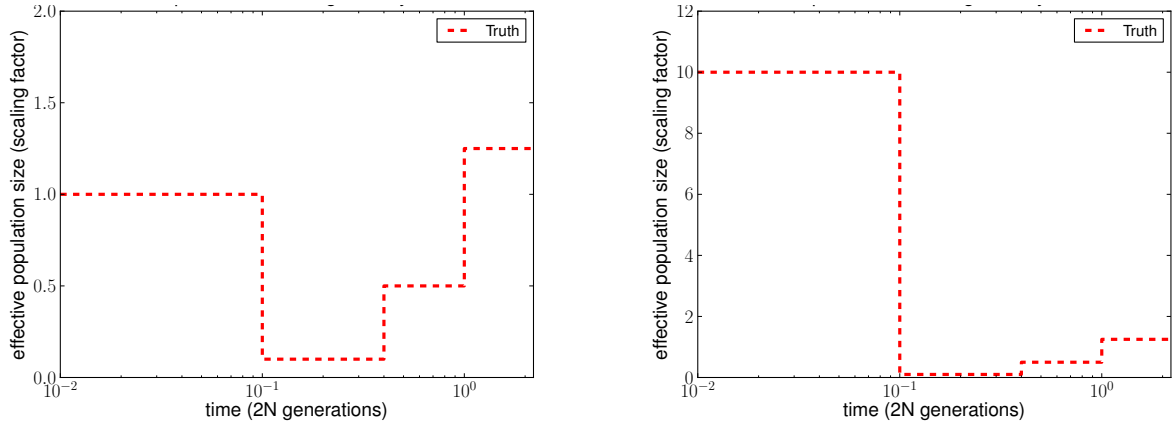
where $\hat{\phi}^{(\lambda)}(D_j|D_i)$ and $\hat{\xi}^{(\lambda)}(b|D_i)$ refer to the transition and emission probabilities where we have marginalized over the absorption haplotype.

We initialize the algorithm with $\lambda_i = 1$ for all $i = 1, \ldots, d$. To compute $[A_{ij}]$ and $[E_i(b)]$, we use the forward and backward probabilities of our HMM. The exact details of making this step computationally efficient are provided in Appendix A. After the E-step, we use the Nelder-Mead optimization routine [81] to update the parameters in the M-step. Because of local maxima in the likelihood surface, we run this optimization routine several times (≈ 10) with different starting conditions and then retain the estimates with the largest likelihood. In the analysis discussed in this paper, we ran the EM procedure for 20 iterations to obtain convergence. As pointed out in Li and Durbin [64], running the EM procedure for many iterations often leads to over-fitting.

2.6 Results

We compared the performance of our method, diCal, with that of PSMC [64] on both simulated and real data. We compared diCal using an n-haplotype leave-one-out scheme (Equation 2.14) with PSMC using the same n haplotypes paired up sequentially (i.e. haplotype 1 paired with haplotype 2, haplotype 3 with haplotype 4, etc).

Unless stated otherwise, we used 16 discretization intervals and inferred 7 free population size parameters in both PSMC and diCal. In the notation introduced in [64], the pattern we used is "3+2+2+2+2+2+3," which means that the first parameter spans the first three



(a) History S1 containing a bottleneck followed by a modest recovery.

(b) History S2 containing a bottleneck followed by a rapid expansion.

Figure 2.6: Population size histories considered in our simulation study, with time $t = 0$ corresponding to the present.

discretization intervals, the second parameter spans the next two intervals, and so on. We found that grouping a few consecutive intervals to share the same parameter significantly improved the accuracy of estimates. For example, due to an insufficient number of coalescence events, the first and last intervals are particularly susceptible to runaway behavior if they are assigned their own free parameters, but grouping with their neighboring intervals prevented such pathological behavior. See Appendix A for further details of running PSMC and our method.

2.6.1 The accuracy of population size inference on simulated data

We used `ms` [50] to simulate full ancestral recombination graphs (ARGs) under two different population histories, and then superimposed a quadra-allelic, finite-sites mutation process on the ARGs to generate sequence data over $\{A, C, G, T\}$. As illustrated in Figure 2.6, both histories contained bottlenecks in the moderately recent past. History S2 in addition contained a recent rapid population expansion relative to the ancient population size. For each history, we simulated 10 independent ARGs for $L = 10^6$ sites and $n = 10$ haplotypes, with the population-scaled recombination rate set to 0.01 per site in `ms`. To add mutations, we set the population-scaled mutation rate to 0.014 per site and used the quadra-allele mutation matrix described in Appendix A.

As shown in Figures 2.7 and 2.8, our method performed much better in the recent past than did PSMC. PSMC often had the type of runaway behavior shown in Figure 2.8, where it overestimated the most recent population size by over three orders of magnitude. We note that our method began to lose accuracy for more ancient times, most likely because ancient

Simulated History	PSMC error	diCal error
S1	0.40328	0.10283
S2	0.71498	0.29992

Table 2.1: Goodness-of-fit for PSMC and diCal, averaged over 10 simulated datasets each with a sample of $n = 10$ haplotypes. The underlying population size histories are shown in Figure 2.6. The error metric used is a normalized integral of the absolute difference between the true history and the inferred history over time. These results demonstrate that diCal is substantially more accurate than the PSMC method.

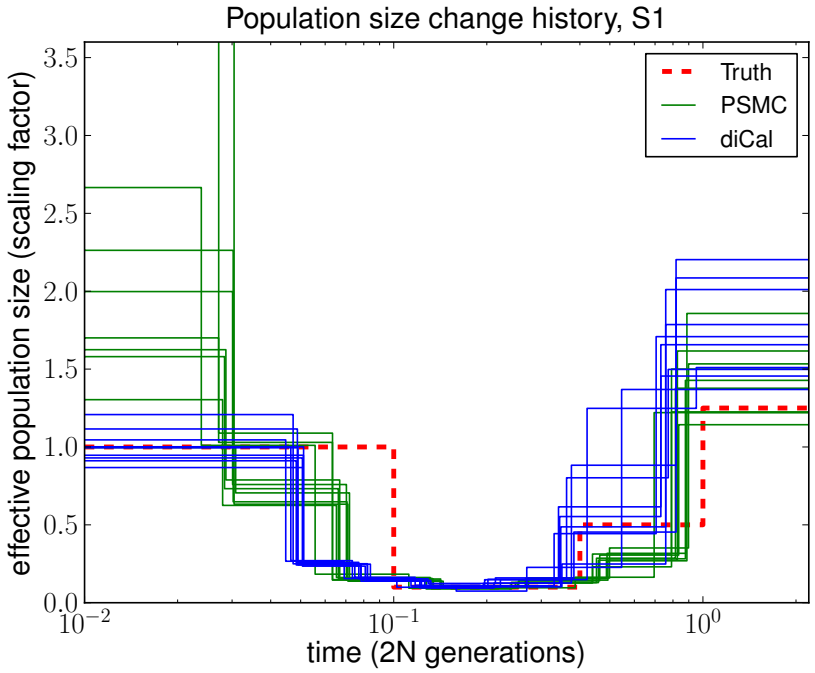
---

absorption events in a 1 Mb region are few and sparsely distributed in time in the leave-one-out SMCSD computation. Both methods tend to smooth out sudden changes in population size, which is why the inferred recovery time from a bottleneck is more recent than it should be. To quantify the improvement in accuracy of our method over PSMC, we used an error metric described in [64], which is a normalized integral of the absolute difference between the true `ms` history and the inferred history over time. The results, summarized in Table 2.1, show that our method had a substantially lower overall error than PSMC.

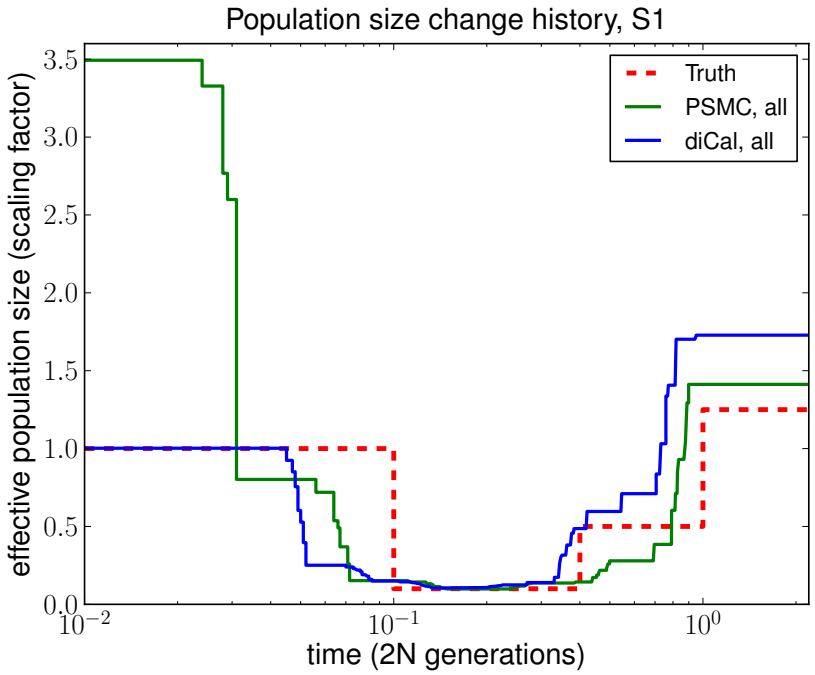
For inference using diCal, we examined the impact of considering more haplotypes on the accuracy of population size estimation. In this study, we focused on history S1 and grouped adjacent parameters to fit roughly with population size change points for illustration purposes. Figure 2.9 shows qualitatively that increasing the sample size n makes our estimate of the recent population size more accurate. Intermediate sizes changed little with increasing n, and ancient sizes were somewhat variable depending on the distribution of coalescence events. Note that for $n = 2$, our method is very similar to PSMC; we compute the transition probabilities slightly differently, but the wedding cake approximation does not change the model in this case. We used the same error metric mentioned above to quantify the advantage of increasing the sample size. As shown in Table 2.2, the overall error decreased as the sample size increased, with improvement tapering around 8 to 10 haplotypes for this particular history.

2.6.2 The impact of the wedding cake genealogy approximation

We examined the advantage of using the wedding cake genealogy approximation in the SMCSD computation, compared to assuming an unmodified trunk genealogy. Figure 2.10 illustrates that the unmodified trunk genealogy leads to overestimation of population sizes in the distant past, as discussed in Section 2.4. The wedding cake genealogy approximation, which adjusts the absorption rate by accounting for the expected number of ancestral lineages of the already observed sample, leads to a significant improvement in the accuracy of population size inference in the ancient past.

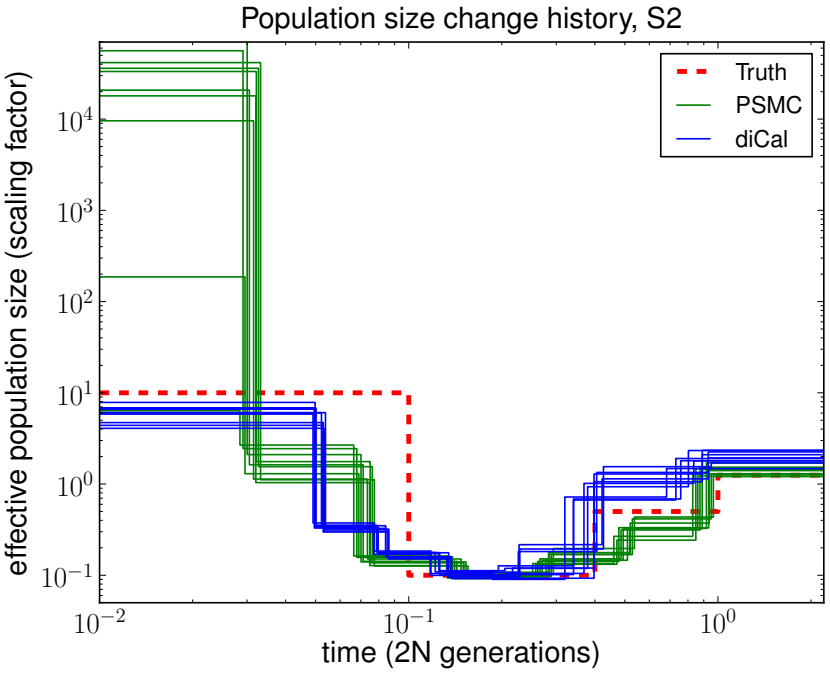


(a) Results for 10 different datasets.

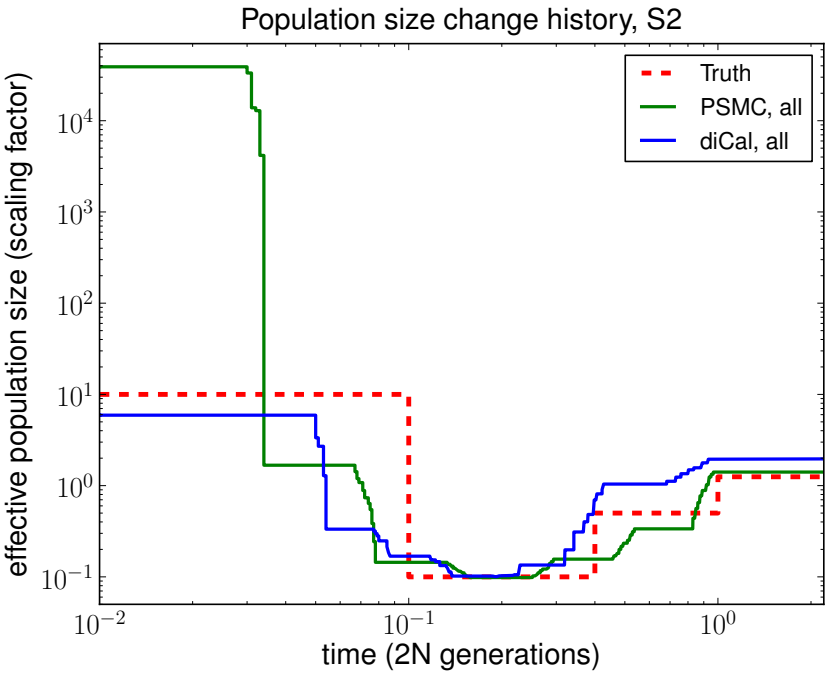


(b) Average over the 10 datasets.

Figure 2.7: Results of PSMC and diCal on datasets simulated under history S1 with sample size $n = 10$ and four alleles (A,C,G,T). PSMC significantly overestimates the most recent population size, whereas we obtain good estimates up until the very ancient past.



(a) Results for 10 different datasets.



(b) Average over the 10 datasets.

Figure 2.8: Results of PSMC and diCal on datasets simulated under history S2 with sample size $n = 10$ and four alleles (A,C,G,T). The PSMC shows runaway behavior during the recent past, overestimating the most recent time by over three orders of magnitude on average.

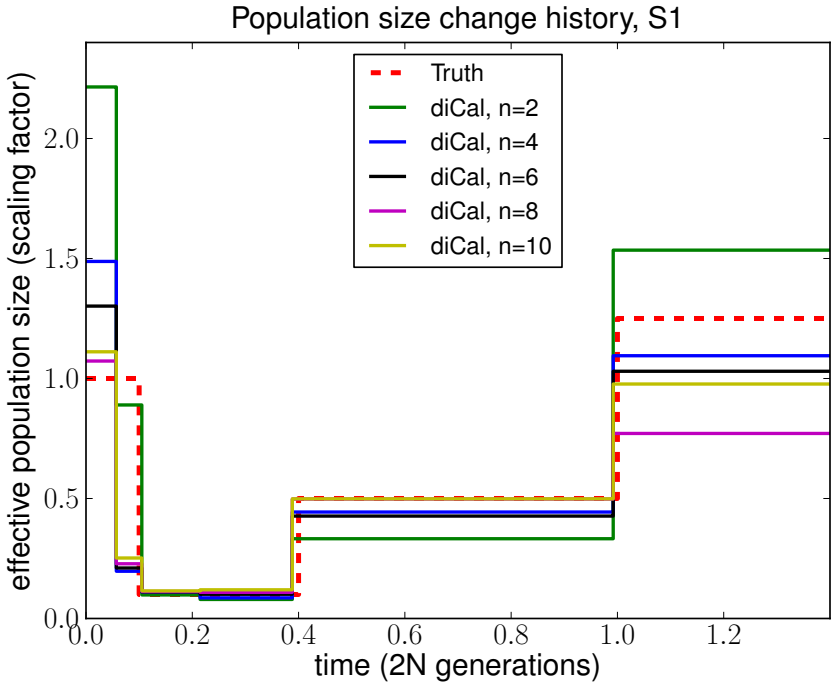


Figure 2.9: The effect of considering more haplotypes in diCal using the SMCSD-based leave-one-out likelihood approach. Data were simulated under population size history S1 with two alleles. In this study, we grouped adjacent parameters to fit roughly with population size change points for illustration purposes. This figure shows the increase in the accuracy of our method with an increasing sample size n. The recent sizes are the most dramatically affected, while intermediate sizes remain accurate even with few haplotypes.

Sample size n	diCal error
2	0.2914
4	0.1901
6	0.1446
8	0.0802
10	0.0899

Table 2.2: Goodness-of-fit for diCal on simulated bottlenecked history S1 for different sample sizes. We used the same error metric as in Table 2.1. As the sample size n increases, the error decreases, with global improvement tapering around 8 to 10 haplotypes.

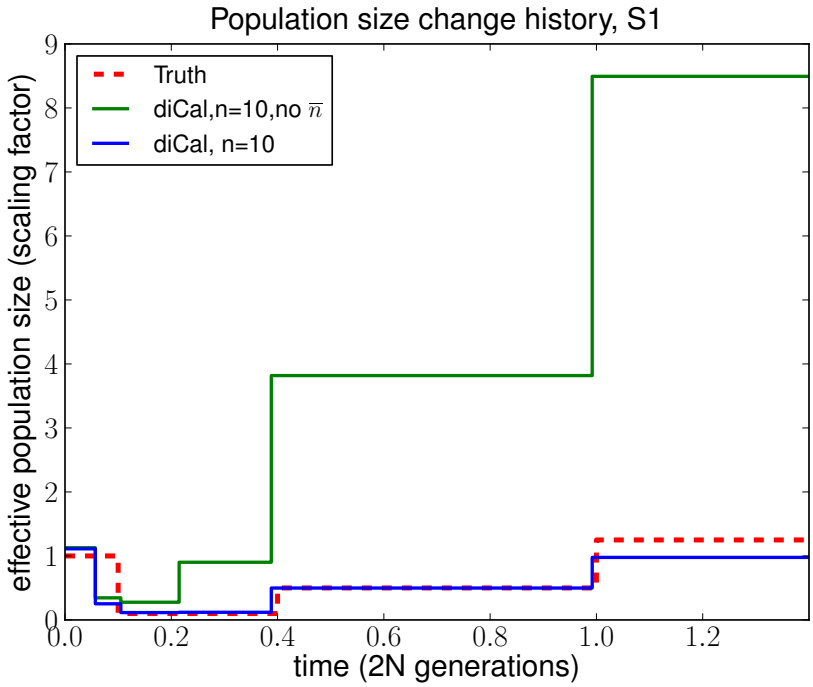


Figure 2.10: A comparison of the SMCSD-based leave-one-out likelihood approach in diCal using the wedding cake genealogy (blue line) with that using the unmodified trunk genealogy (green line). The results shown here are for $n = 10$ haplotypes simulated under history S1 with two alleles. Without the wedding cake genealogy, absorption of the left-out lineage into the trunk occurs too quickly, and the lack of absorption events in the mid to ancient past leads to substantial overestimation of the population sizes. Recent population sizes remain unaffected since during these times the absorption rates in the wedding cake genealogy and in the trunk genealogy are roughly the same. In this example, we grouped adjacent parameters to fit roughly with population size change points for illustration purposes.

2.6.3 The accuracy of estimated coalescence times

To assess the accuracy of estimated coalescence times, we produced the posterior decoding and the posterior mean of the times that a left-out haplotype got absorbed into a wedding cake genealogy. The data were simulated under the full coalescent with recombination using `ms` assuming a constant population size. The true coalescence time at each site was taken as the time the left-out lineage joined the rest of the coalescent tree at that site. As shown in Figure 2.11, we found that our estimated absorption times closely tracked the true coalescence times.

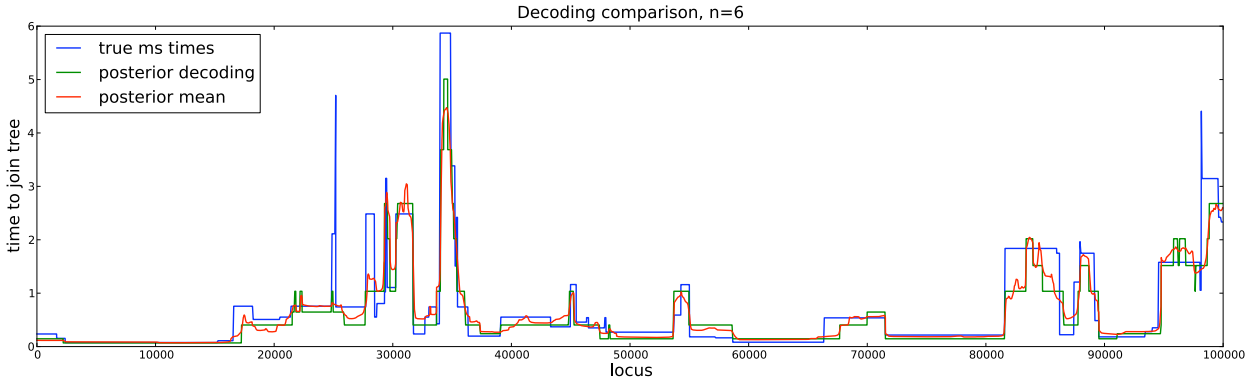


Figure 2.11: Estimated absorption times in diCal using the leave-one-out SMCSD method versus the true coalescence times for a 100 kb region. The data were simulated using `ms` for $n = 6$ haplotypes assuming a constant population size. The true coalescence time at each site, obtained from `ms`, was taken as the time the ancestral lineage of a left-out haplotype joined the rest of the coalescent tree at that site. The figure shows the true coalescence time for the nth haplotype and our corresponding inferred absorption times, obtained from the posterior decoding and the posterior mean. Our estimates generally track the true coalescence times closely.

2.6.4 Results on real data

We applied our method to European (CEU) and African (YRI) subsamples from the 1000 Genomes Project [14]. To minimize potential confounding effects from natural selection, we chose a 3 Mb region on chromosome 1 with no genes and then used the middle 2 Mb for analysis. We used the human reference (version 36) to create a full multiple sequence alignment of 10 haplotypes (5 diploid individuals) for each of the CEU and YRI populations. Although we filtered out unphased individuals and sites, the final sequences are based on low-coverage short read data, so phasing and imputation errors could impact the accuracy of our coalescence time inference. We assumed a per-generation mutation rate of $\mu = 1.25 \times 10^{-8}$ per site, which is consistent with recent studies of *de novo* mutation in human trios [2, 60, 99], and a mutation transition matrix estimated from the human and the chimp reference genomes (shown in Appendix A). For simplicity, we assumed that the per-generation recombination rate r between consecutive bases is constant and equal to μ. The generation time was assumed to be 25 years. For a reference population size, we used $N_{\text{ref}} = 10,000$.

The results of PSMC and our method are shown in Figure 2.12. PSMC displayed runaway behavior and produced rather unrealistic results; see Figure 2.12(a), for which we truncated the y-axis at 40,000 for ease of comparison with Figure 2.12(b). The dataset may be too small for PSMC to work accurately. We note that PSMC was able to produce more reasonable results on simulated datasets, probably because they were generated with much higher mutation and recombination rates, thus representing a larger genomic region for humans.

As shown in Figure 2.12(b), our method inferred that CEU and YRI had very similar histories in the distant past up until about 117 kya; discrepancies up to this point are most likely due to few observed ancient coalescence events with the leave-one-out approach. We inferred that the European population underwent a severe (out-of-Africa) bottleneck starting about 117 kya, with the effective population size dropping by a factor of about 12 from $\approx$ 28,000 to $\approx$ 2,250. Furthermore, at the level of resolution provided by our time discretization, our results suggest that the European population has recovered from the bottleneck to an average effective size of $\approx$ 12,500 for the past 16 thousand years.

In contrast to previous findings (e.g., [64]), our method did not infer a significant drop in the YRI population size during the early out-of-Africa bottleneck phase in Europeans. Instead, the African effective population size seems to have decreased more gradually over time (possibly due to changes in structure) to an average effective size of $\approx$ 10,000 for the past 16 thousand years. We note that our results for real data are fairly robust to the choice of discretization, given that a sufficient number of coalescence events occur within each set of grouped intervals.

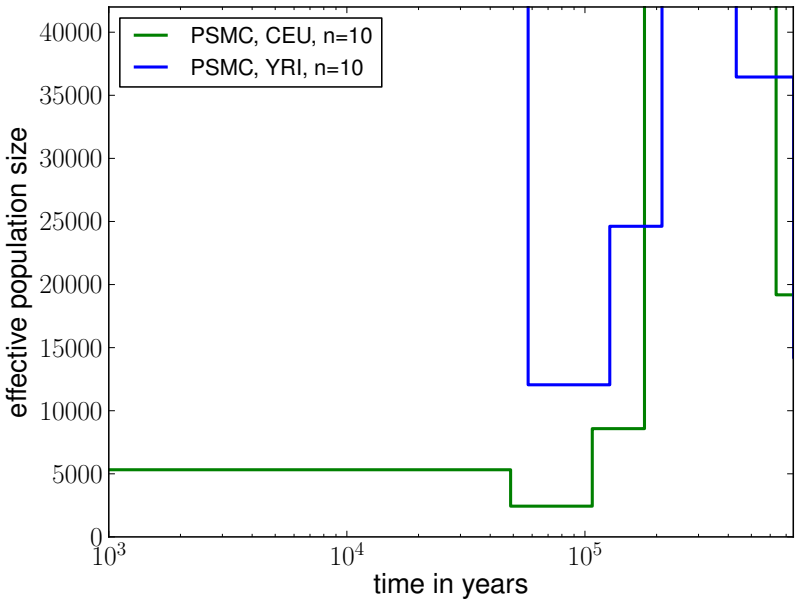
2.6.5 Runtime

The runtime of our method is $O(Ld(d+n)n)$, where n is the number of sequences, L is the number of bases in each sequence, and d is the number of time discretization intervals; the runtime for each CSD computation is $O(Ld(d+n))$, and each sequence is left out in turn (although this step is parallelizable). The runtime for PSMC is $O(Ld^2P)$, where P is the number of pairs of sequences analyzed. In practice, PSMC can run much faster when consecutive sites are grouped into bins of size 100; a bin is considered heterozygous if it contains at least one SNP and homozygous otherwise. Creating a reasonable binning scheme for multiple sequences is less clear, but would significantly improve our runtime and enable whole-genome analysis. One possible binning scheme is described in chapter 3.

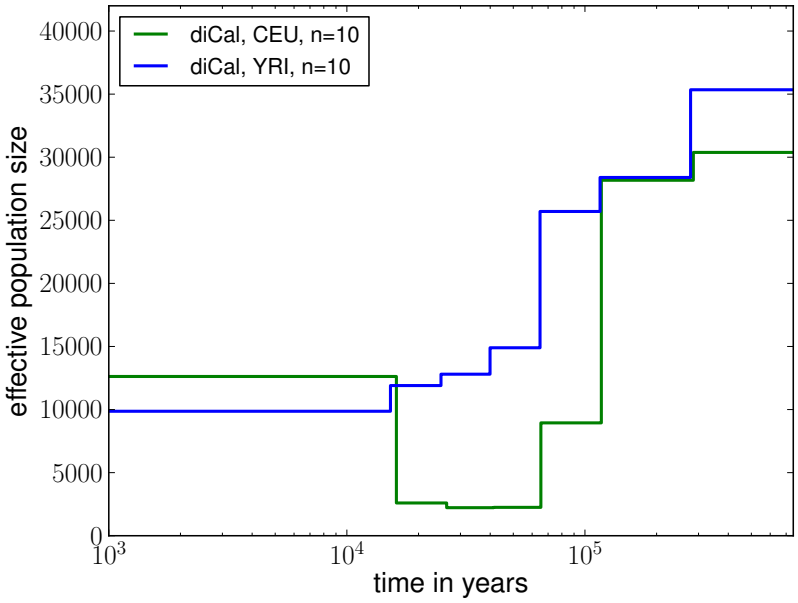
2.7 Discussion and future work

In this paper, we have generalized the recently developed sequentially Markov conditional sampling distribution framework [90] to accommodate a variable population size. One important new idea central to the success and accuracy of our method is the wedding cake genealogy approximation, which modifies the rate of absorption into the trunk by accounting for the varying number of lineages over time. On simulated data, we have shown that our method produces substantially more accurate estimates of the recent effective population size than does PSMC [64].

Applying our method to a 2 Mb intergenic region of chromosome 1 from five Europeans and five Africans, sequenced as part of the 1000 Genomes Project, and using a per-generation mutation rate of $\mu = 1.25 \times 10^{-8}$ per site, we have inferred a severe (out-of-Africa) bottleneck in Europeans that began around 117 kya, with a drop in the effective population size by a



(a) The results of PSMC, which had some runaway behavior and unrealistic results; the dataset is probably too small for PSMC to work accurately.



(b) The results of diCal. We inferred that the European population size underwent a severe bottleneck about 117 kya and recovered in the past 16,000 years to an effective size of $\approx$ 12,500. In contrast, our results suggest that the YRI population size did not experience such a significant drop during the early out-of-Africa bottleneck phase in Europeans.

Figure 2.12: Variable effective population size inferred from real human data for European (CEU) and African (YRI) populations. For each population, we analyzed a 2 Mb region on chromosome 1 from 5 diploid individuals (10 haplotypes), assuming a per-generation mutation rate of $\mu = 1.25 \times 10^{-8}$ per site.

factor of 12. In contrast, we have observed a much more mild population size decrease in the African population. We remark that our estimate of the timing of the bottleneck may not be very accurate, since we used only 16 discretization intervals and 7 free population size parameters. Furthermore, all of our inferred times and population sizes would be smaller by a factor of two if we had used $\mu = 2.5 \times 10^{-8}$. See Scally and Durbin [101] for a more thorough discussion of how new mutation rate estimates are changing the way we view ancient population history. An earlier initial human dispersal out of Africa would fit with archaeological evidence of human artifacts dated at 74 kya in India and 64 kya in China [101].

During the recent past, our results demonstrate that the effective population size of Europeans has grown in the past 16,000 years, slightly surpassing the effective population size of Africans, which does not show a growth at this resolution. Recent studies [32, 38] suggest that the European population size recently grew much faster than the African population size, although the sample size we considered is not large enough to confirm this.

The main strength of our method is in the recent past. Large-scale sequencing studies [15, 57, 82] of a subset of genes suggest that humans underwent recent explosive population growth. Our method should be well equipped to infer such recent demographic histories, but we would need to consider a very large sample to accurately infer the rate of expansion and the time of onset. Because of issues of computational speed and memory footprint, our current implementation of the SMCSD computation can handle up to about 20 haplotypes and a few megabases, but we are in the process of exploring ways to increase the scalability. One way in which we should be able to reduce our runtime is by incorporating recently developed algorithms for blockwise HMM computation [89], which have been shown to result in a speed-up of several orders of magnitude for large datasets.

All the results in this chapter make use of a leave-one-out approach (Equation 2.14) instead of the well-used product of approximate conditionals (PAC) method proposed in [66]. Briefly, the PAC approach utilizes the approximate likelihood $\hat{\pi}(h_{\sigma(1)})\hat{\pi}(h_{\sigma(2)}|h_{\sigma(1)}) \cdots \hat{\pi}(h_{\sigma(n)}|h_{\sigma(1)}, \ldots, h_{\sigma(n-1)})$, where $\hat{\pi}$ is an approximate conditional sampling distribution and σ is some permutation of $\{1, \ldots, n\}$. A well-known drawback of this approach is that different permutations may produce vastly different likelihoods. Li and Stephens [66] suggests averaging the PAC likelihood over several random permutations to alleviate this problem and this strategy seems to work reasonably well in practice. In our work, we have avoided the problem by adopting the leave-one-out approach, which yields accurate estimates of population sizes for the recent past, but likely impacts accuracy in the ancient past. Employing the PAC approach may produce accurate estimates for all times, but a challenge that needs to be addressed in the SMCSD framework is that the wedding cake genealogy, which is based on the *prior* expectation of the number of lineages, may not be accurate when there are few lineages, since coalescence times are more variable when they involve fewer lineages. We are working on improving the accuracy of the SMCSD computation by using the *posterior* absorption time distributions in a recursive fashion so that locus-specific absorption rates tailored to data can be used. This approach, together with the PAC model, should yield more accurate estimates of population sizes.

One factor that we have not investigated is the impact of variable recombination (and/or mutation) rates, although it is conceptually straightforward for our method to accommodate them. We have chosen not to incorporate recombination rate variation into our present implementation as it would make the method even more computationally expensive, since the transition probabilities would then be potentially different at each site. In addition, most fine-scale recombination maps [10, 16, 27, 74] are inferred under the assumption of a constant population size, which is exactly the assumption we are *not* making. We also note that [64] found that recombination hotspots did not impact their results significantly and that the important parameter is the average recombination rate.

A good choice of time discretization is critical to the performance of both diCal and PSMC. It is better to subdivide time more finely during periods with small population size than during periods with large population size when few coalescences occur. However, since the demography is what we are trying to infer, selecting an initial discretization is very difficult. Creating adaptive discretization schemes for coalescent HMMs is an important area of future research, which we investigate in chapter 3.

We have shown that posterior decodings of diCal enable accurate inference of coalescence times. Using this information, we provide an efficient method of sampling marginal coalescent trees from the posterior distribution in chapter 3. We expect such local tree inference to have interesting applications, including genome-wide association studies and tests of selective neutrality.

The SMCSD framework has been recently extended to incorporate structured populations with migration [105]. We are currently working on combining this extension with the work presented here to implement an integrated inference tool (to be incorporated into diCal) for general demographic models. Such a method could provide a detailed picture of the demographic history that created the diversity we see today in humans and other species.

Chapter 3

Improvements to coalescent hidden Markov models

Since its initial release, diCal has been improved in several ways. The most significant advance was the development of a forward-backward algorithm that runs in linear time in the number of time discretization intervals. Previously, all coalescent HMMs ran in quadratic time in the number of time intervals, but this improvement can be implemented for any coalescent HMM using a modified state space that models recombination more explicitly. The full details of this algorithm can be found in Harris et al. [44]. My main contributions to this work were the implementation and analysis. Section 3.1 gives a brief overview of the methods and then describes the analysis and results. The implementation for the linear-time version is available in the publicly released code at `http://sourceforge.net/projects/dical/`.

In Section 3.2, we provide a procedure for grouping adjacent sites into "bins" for faster computation, and an adaptive discretization algorithm is described in Section 3.3. In Section 3.4, we describe how diCal can be used to build local genealogies, which can assist in population size inference. Finally, minor improvements to the implementation are described in Section 3.5.

3.1 Decoding coalescent HMMs in linear time

3.1.1 Method overview

Usually when we compute the transition probabilities of an HMM, the runtime is quadratic in the number of hidden states, since we must compute the transition probability between each pair of states. The intuition behind bringing this down to a linear operation is that some of the transitions have equal probability or can be written in terms of previously computed quantities. To achieve this, we examine the transitions more closely. As in chapter 2, we will have d time intervals in the discretization and n previously sampled haplotypes. The hidden state space is the interval and the haplotype that our "left-out" haplotype coalesces with at a

particular locus. Say we are trying to compute the transition from interval k and haplotype h' at locus ℓ, to interval j and haplotype h at locus $\ell+1$, which we can denote $\phi((h,j)|(h',k))$. Let $\bar{R}$ denote the event that there is no recombination between these two loci. If there was a recombination event, it must have occurred in interval $i \in \{1, 2, \cdots \min(j,k)\}$. Denote this event R_i. If we let T_ℓ be the hidden time interval at locus ℓ, then we can write the the transition probability as

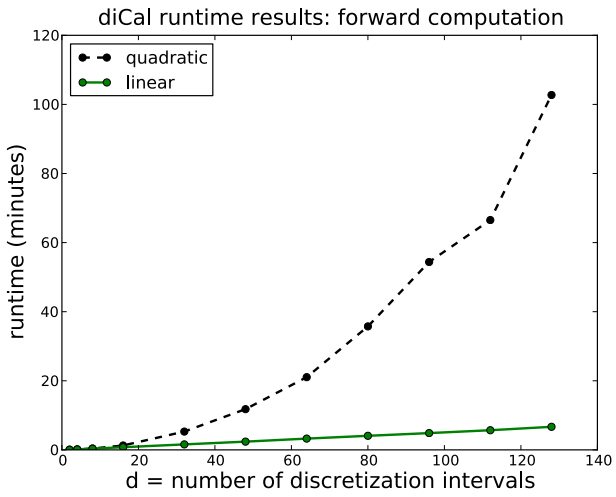
$$\phi((h,j)|(h',k)) = \mathbb{1}_{(h,j)=(h',k)} \cdot \mathbb{P}(\bar{R}|T_\ell = k) + \frac{1}{n} \sum_{i=1}^{\min(j,k)} \mathbb{P}(R_i, T_{\ell+1} = j | T_\ell = k). \tag{3.1}$$

The first term (no recombination) can be pre-computed in looked up in constant time as before, but we need to decompose the second term (recombination) further. If there is a recombination event in interval i, there are different options of how this could relate to intervals k and j. Let C_i denote the event that the recombined lineage coalesces at interval i (as well as having recombined in interval i), and let $C_{>i}$ denote the event that the lineage recombines at or before interval i, then "floats" off to coalesce during a more ancient interval than i. Then we can write the second probability from Equation 3.1 in terms of four different scenarios:

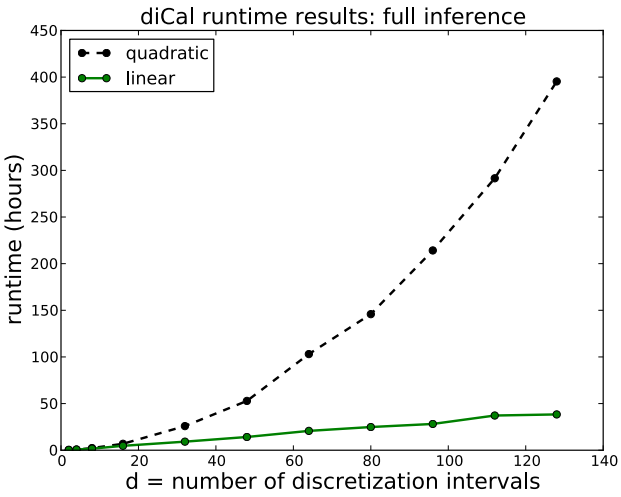
$$\begin{aligned}
\mathbb{P}(R_i, T_{\ell+1} = j | T_\ell = k) = \quad & \mathbb{1}_{i=j=k} \cdot \mathbb{P}(R_i, C_i | T_\ell = i) \\
+ \quad & \mathbb{1}_{i=j<k} \cdot \mathbb{P}(R_i, C_i | T_\ell > i) \\
+ \quad & \mathbb{1}_{i=k<j} \cdot \mathbb{P}(R_i, C_{>i} | T_\ell = i) \cdot \prod_{m=i+1}^{j-1} \mathbb{P}(C_{>m} | C_{>m-1}) \\
+ \quad & \mathbb{1}_{i<\min(j,k)} \cdot \mathbb{P}(R_i, C_{>i} | T_\ell > i) \cdot \prod_{m=i+1}^{j-1} \mathbb{P}(C_{>m} | C_{>m-1}).
\end{aligned} \tag{3.2}$$

All of these terms can be precomputed and looked up in constant time, and none of them depend on $T_{\ell+1}$. Therefore the sum from Equation 3.1 can be computed in linear time in the number of intervals, giving us the desired runtime. There are further details about the forward and backward algorithms in [44].

Before, multivariate optimization was using during the M-step to find the series of effective population sizes that maximized the HMM likelihood (using the Nelder-Mead algorithm [81]). There are $O(d)$ population sizes to estimate, and Nelder-Mead is potentially non-linear in d. To make sure that this new algorithm is still linear, we optimize each population size separately, starting from the most recent size. Due to the way we decomposed the transition (and emission) probabilities in the E-step, the population size estimate for interval i only depends on the estimates in intervals less than i. In this way the entire estimation procedure is linear in d.



(a) Runtime results (in minutes) for the forward computation.



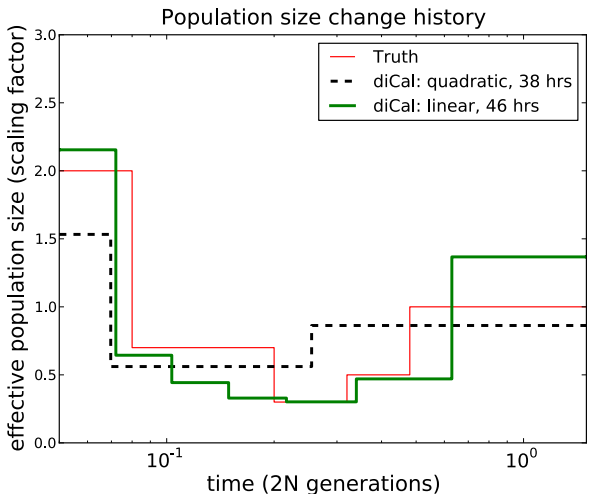
(b) Runtime results (in hours) for the entire EM inference algorithm (20 iterations) extrapolated from the time for one iteration.

Figure 3.1: Runtime results on simulated data with $L = 2$ Mb and 2 haplotypes, for varying number d of discretization intervals.

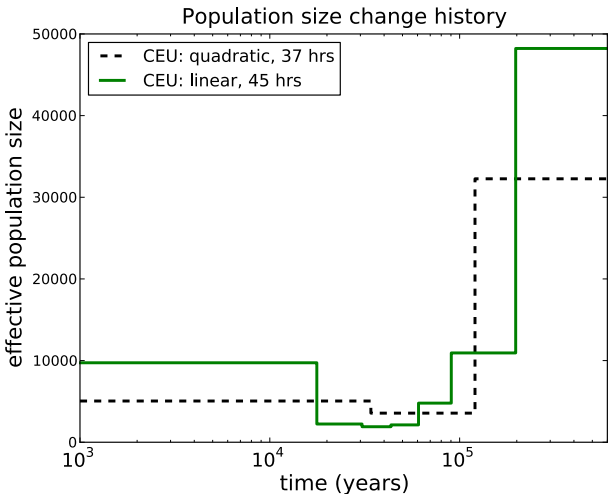
3.1.2 Results

To confirm the decrease in runtime, we ran the linear-time diCal method on simulated data with $L = 2$ Mb of loci and 2 haplotypes (in which case diCal is equivalent to PSMC), using $d = 2, 4, 8, 16, 32, 48, 64, 80, 96, 112, 128$ discretization intervals. To simulate the data, we used `ms` [50] with a population-scaled recombination rate $\rho = 0.0005$ to generate a genealogy, and then added mutations using a population-scaled mutation rate of $\theta = 0.0029$ and a mutation matrix described in [103]. Figure 3.1(a) shows the time to compute the table of forward probabilities. We also measured the time for one EM iteration and then extrapolated to 20 iterations to approximate the time required to estimate an effective population size history (Figure 3.1(b)). In both figures, the linear runtime of our new algorithm is apparent and significantly improves our ability to increase the number of discretization intervals.

To assess the gain in accuracy of population size estimates that is afforded by more discretization intervals, we ran both the linear- and quadratic-time methods on simulated data with 10 haplotypes and $L = 2$ Mb. We use the conditional sampling distribution in a leave-one-out composite likelihood approach [103] in this experiment. So that each method ran for roughly the same amount of time ($\approx$ 40 hours), we used $d = 9$ for the quadratic method and $d = 21$ for the linear method. For both methods, we ran EM for 20 iterations and estimated $d/3$ size change parameters. Larger values of d permit the inference of more accurate histories, as measured by the PSMC error function, which integrates the absolute value of the difference between the true size function and the estimated size function [64].



(a) Results on simulated data with $L = 2$ Mb and 10 haplotypes. Using the quadratic method with $d = 9$, the error was 0.148. Using the linear method with $d = 21$, the error dropped to 0.079.



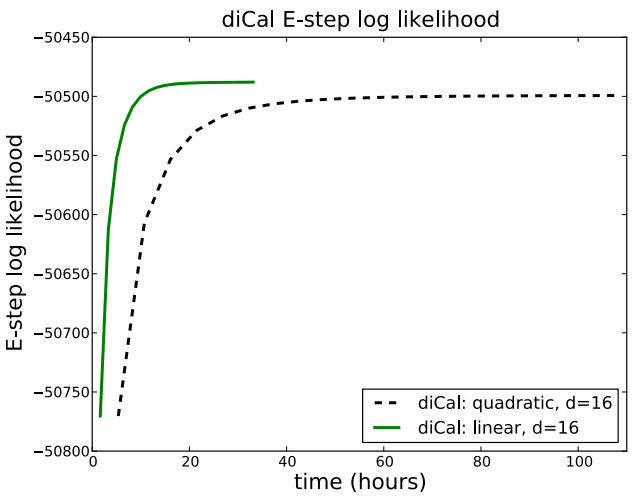
(b) Results on 10 European haplotypes over a 2 Mb region of chromosome 1. The out-of-Africa bottleneck is very apparent with $d = 21$, but is not as well characterized for $d = 9$.

Figure 3.2: Effective population size change history results. The speedup from the linear method allows us to use a finer discretization ($d = 21$) than the quadratic method ($d = 9$) for about the same amount of runtime.

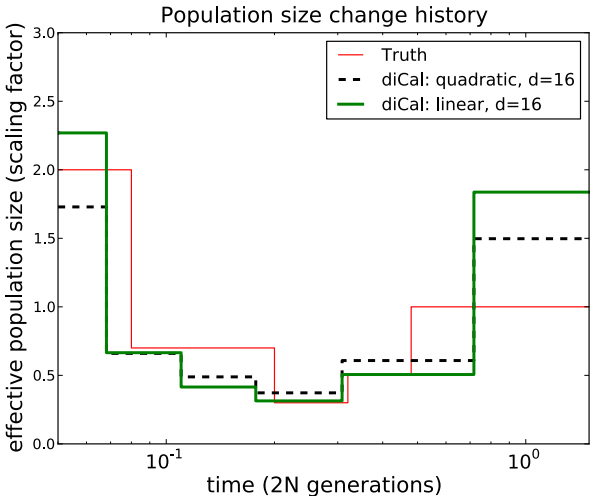
We also ran our method on 10 CEU haplotypes (Utah residents of European descent) sequenced during Phase I of the the 1000 Genomes Project [14] (Figure 3.2(b)). For the quadratic method with $d = 9$, we are unable to fully characterize the out-of-Africa bottleneck. In the same amount of computational time, we can run the linear method with $d = 21$ and easily capture this feature. The disagreement in the ancient past between the two methods is most likely due to diCal's lack of power in the ancient past when there are not many coalescence events. Using a leave-one-out approach with 10 haplotypes, the coalescence events in the ancient past tend to be few and unevenly spaced, resulting in a less confident inference.

The runtime of the full EM algorithm depends on the convergence of the M-step, which can be variable. Occasionally we observed convergence issues for the quadratic method, which requires a multivariate optimization routine. For the linear method, we used the univariate Brent optimization routine from Apache Math Commons (`http://commons.apache.org/proper/commons-math/`), which converges quickly and to a large extent avoids local maxima.

To examine the convergence of the two EM algorithms, we ran the linear and quadratic methods on the simulated data with 10 haplotypes and the same number of intervals $d = 16$. We examine the likelihoods in Figure 3.3(a). The linear method reaches parameter estimates of higher likelihood, although it is unclear whether the two methods have found different



(a) The log likelihood of the EM algorithm, plotted against time, for both the linear and quadratic methods.



(b) Population size change history results for the linear and quadratic methods, run with the same discretization using $d = 16$ and estimating 6 parameters.

Figure 3.3: Results on simulated data, using the same discretization for the linear and quadratic methods. Each method was run for 20 iterations.

---

local maxima, or whether the quadratic method is approaching the same maximum more slowly. Figure 3.3(b) shows the inferred population sizes for each method, which although similar, are not identical. We sometimes observe that the linear method overestimates the most recent population size more than the quadratic method. This is perhaps due to the univariate optimization procedure, which estimates the most recent size first, independently of the subsequent sizes.

We have also investigated the amount of memory required for each method, and although the difference is small, the linear method does require more memory to store the augmented forward and backward tables. In theory, the memory requirement each forward or backward table should be $O(ndL)$. For the quadratic method, there are two such tables, and for the linear method there are an additional five tables that require $O(dL)$ memory. A more thorough investigation of memory requirements will be important as the size of the data continues to increase.

3.2 Grouping adjacent sites (binning)

In PSMC [64], adjacent sites are grouped into "bins" to speed up computation. With two sequences, they consider an emission to be '0' if the two sequences are identical with a bin, and '1' if there is any difference between the two sequences within a bin. With more than

two sequences, binning becomes more difficult, since there could be multiple SNPs affecting different subsets of individuals within the same bin. Here we provide a binning framework for multiple sequences.

Let b be the number of bases in each bin (for humans, a typical bin size is $b = 100$). A '0' will still represent a bin in which there are no segregating sites, but now a bin with segregating sites will contain the bases of the segregating sites, but not their locations within the bin (although relative ordering will be preserved). Bins will be separated by commas. For example, input data for 4 individuals using 7 bins might look like:

$$\begin{array}{ll} 1: & 0, AC, 0, 0, G, 0, GTA \\ 2: & 0, AG, 0, 0, A, 0, GTC \\ 3: & 0, TG, 0, 0, G, 0, CGC \\ 4: & 0, TC, 0, 0, G, 0, GGC \end{array}$$

In this framework, if we see a '0', we assume that no mutation occurred, which is different from before when we integrated over all possible ways locus/bin α could become locus/bin β, regardless of whether or not $\alpha = \beta$. This means that we can no longer have a mutation matrix with non-zero diagonals, since if we see non-segregating sites, we have no information about which bases are actually present. Therefore our mutation matrix $\mathbf{P}$ should be of the form

$$\mathbf{P} = \begin{bmatrix} 0 & p_{AC} & p_{AG} & p_{AT} \\ p_{CA} & 0 & p_{CG} & p_{CT} \\ p_{GA} & p_{GC} & 0 & p_{GT} \\ p_{TA} & p_{TC} & p_{TG} & 0 \end{bmatrix}.$$

Before discretizing time, let the hidden state at locus ℓ be (t, h), and the mutation rate be θ. Then if $h[\ell] = \alpha$, the probability of emitting bin β is

$$\xi^{(\lambda)}(\beta|t,h) = \begin{cases} e^{-\theta b t} & \text{if } \alpha = \beta \\ e^{-\theta(b-1)t}(1-e^{-\theta t})[\mathbf{P}]_{\alpha_1,\beta_1} & \text{differ at one base: } \alpha_1 \neq \beta_1 \\ e^{-\theta(b-2)t}(1-e^{-\theta t})^2[\mathbf{P}]_{\alpha_1,\beta_1}[\mathbf{P}]_{\alpha_2,\beta_2} & \text{two bases: } \alpha_1 \neq \beta_1,\ \alpha_2 \neq \beta_2 \\ e^{-\theta(b-3)t}(1-e^{-\theta t})^3[\mathbf{P}]_{\alpha_1,\beta_1}[\mathbf{P}]_{\alpha_2,\beta_2}[\mathbf{P}]_{\alpha_3,\beta_3} & \alpha_1 \neq \beta_1,\ \alpha_2 \neq \beta_2, \text{ and } \alpha_3 \neq \beta_3 \\ \vdots & \vdots \end{cases} \tag{3.3}$$

In practice we will assume there is a cut-off for the number of segregating sites in one bin. To discretize the state space, we again let $D_i = [t_{i-1}, t_i)$, and integrate to find the discretized emission probabilities

$$\begin{aligned} \hat{\xi}^{(\lambda)}(\beta|D_i,h) &= \frac{1}{\hat{\zeta}^{(\lambda)}(D_i,h)} \int_{D_i} \xi^{(\lambda)}(\beta|t,h)\zeta^{(\lambda)}(t,h)\ \mathrm{d}t \\ &= \frac{\bar{n}_i}{\lambda_i(1-e^{-(t_i-t_{i-1})\bar{n}_i/\lambda_i})} \int_{t_{i-1}}^{t_i} \xi^{(\lambda)}(\beta|t,h)e^{-(t-t_{i-1})\bar{n}_i/\lambda_i}\ \mathrm{d}t \end{aligned}$$

This approach takes into account the order of the segregating sites within each bin, but we can also consider an unordered approach. In this case, the bin size is not taken into account directly as above, but indirectly via multiplying the mutation rate θ by b. Then we have

$$\xi_p^{(\lambda)}(\beta|t,h) = \frac{(\theta bt)^k e^{-\theta bt}}{k!}[\mathbf{P}]_{\alpha_1,\beta_1}\cdots[\mathbf{P}]_{\alpha_k,\beta_k}, \tag{3.4}$$

where k is the number of differences between α and β. This formulation assumes mutations occur as a Poisson process within each bin. We can integrate as before to obtain the discretized emission probabilities $\hat{\xi}_p^{(\lambda)}(\beta|D_i,h)$.

3.2.1 E-step

In the E-step, we count up the number of times we expect to see allele (not bin) $\delta \in \{A,C,G,T\}$ emitted from time state i when the trunk allele is $\sigma \neq \delta$. Let x be the left-out haplotype, and L be the number of bins. Let the number of alleles in bin ℓ be a_ℓ. Then our expected counts are

$$E(i,\sigma,\delta) = \frac{1}{\mathbb{P}(x)}\sum_{\ell=1}^{L}\sum_{h}\sum_{j=1}^{a_\ell}\mathbb{1}\left(\sigma = h[\ell][j], \delta = x[\ell][j]\right)\cdot f_\ell(i,h)b_\ell(i,h).$$

We also have a separate count of the expected number of sites with no mutation, denoted $E(i)$ for time interval i

$$E(i) = \frac{1}{\mathbb{P}(x)}\sum_{\ell=1}^{L}\sum_{h} f_\ell(i,h)b_\ell(i,h)\cdot\left((b-a_\ell) + \sum_{j=1}^{a_\ell}\mathbb{1}\left(h[\ell][j] == x[\ell][j]\right)\right).$$

The idea here is to include the $(b-a_\ell)$ non-segregating sites in bin ℓ, in addition to the sites where the trunk allele and the emitted allele are the same.

3.2.2 M-step

In the M-step, we include the expected emissions by adding the following expression onto the Q function:

$$\sum_{i=1}^{d} E(i)\cdot\log[\hat{\xi}^{(\lambda)}(0|D_i,0)] + \sum_{\sigma}\sum_{\delta\neq\sigma} E(i,\sigma,\delta)\cdot\log[\hat{\xi}^{(\lambda)}(\delta|D_i,\sigma)]$$

where $\hat{\xi}^{(\lambda)}(0|D_i,0)$ is the single site (i.e. bin size 1) emission probability of no mutation given time interval i, and $\hat{\xi}^{(\lambda)}(\delta|D_i,\sigma)$ is the single site emission probability of a mutation from σ to δ given time interval i.

3.2.3 Variations

If we use Equation 3.3, we denote the scheme "binary" (each site is either the same or different) or "ordered" (the mutations are ordered within a bin). If we use Equation 3.4, we denote the scheme "Poisson" or "unordered" (the mutations are unordered within a bin).

A final variation is a "psmc-style" version, where instead of looking at the mutations within a bin, for each pair of individuals, we just note whether or not they are different at all within a bin. Then the emission probabilities are the same as in the PSMC [64]. In the following analysis, we will compare these three binning approaches.

3.2.4 Binning results

To simulate data, we used the commandline from the supplementary material of the PSMC paper, which is meant to be a good proxy for human history and the human genome. First we looked at the decoding results using binning, as shown in Figure 3.4, and observed good performance.

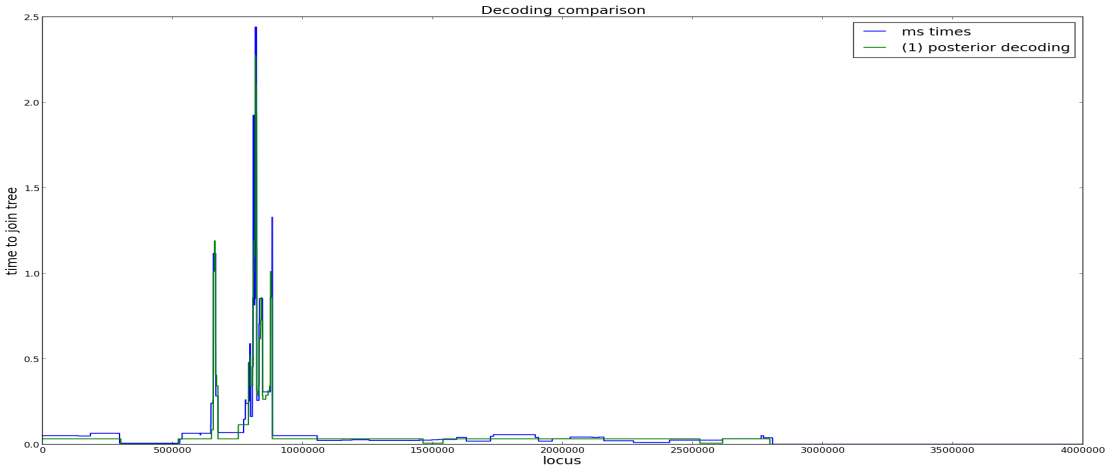
Using the psmc-style binning scheme, we ran a comparison of the results for $n = 2$ and $n = 10$. For $n = 10$, we paired up consecutive sequences for PSMC, so both methods would be given the same amount of data. The results are shown in Figure 3.5. For $n = 2$, the results are very similar, with too much runaway behavior in the recent past for both methods to do a meaningful comparison. Unfortunately, for $n = 10$, diCal's results were less accurate than the PSMC, which was not the goal. This may be due to a variety of factors, including the optimization routine in the M-step.

Figure 3.6(a) shows a comparison using the ordered binning approach for diCal, and is very similar to Figure 3.5(b). The Poisson scheme performs similarly, as shown in Figure 3.6(b).

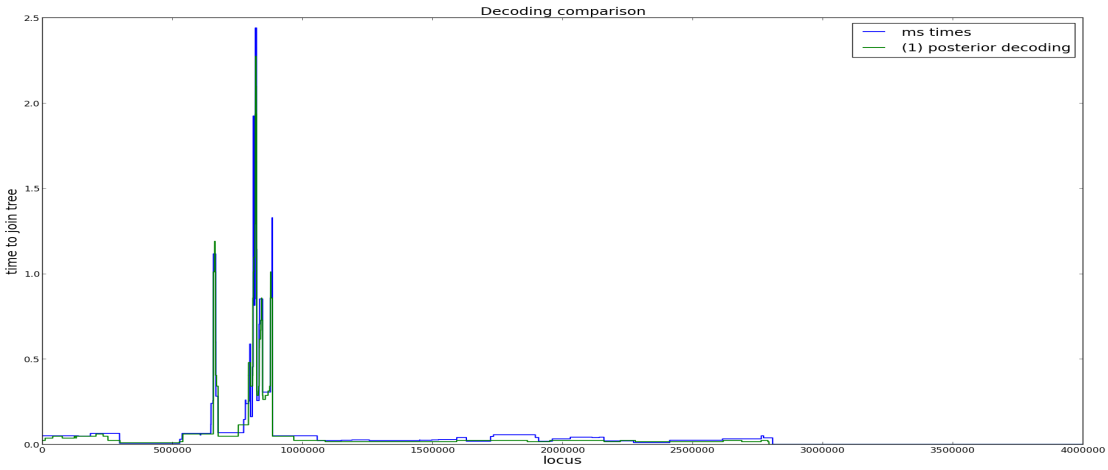
Given these results, binning is not currently supported in the released implementation, but it is still a promising area of future work due to the enormous computational savings. Currently the main issue is the most recent size - the other sizes are still accurately estimated using binning. Bins of size 1 are too computationally expensive, and often little is lost by grouping adjacent sites together, as long as the bin size is chosen such that multiple SNPs within a bin are rare. Future work on applying coalescent HMMs to many whole-genome sequences will require binning or a similar approach to be computationally feasible.

3.3 Adaptive discretization

One of the difficulties of coalescent HMMs is choosing a time discretization that that is appropriate for the species and the number of individuals. Ideally a discretization would be fine enough to capture detailed size changes, but in both [64] and [103] it was noted that very fine discretizations lead to runaway behavior. This is due to the lack of coalescent events that will be observed during a very small time interval, so it looks like the population size is very large, even infinite. One solution is to group adjacent time intervals together, and

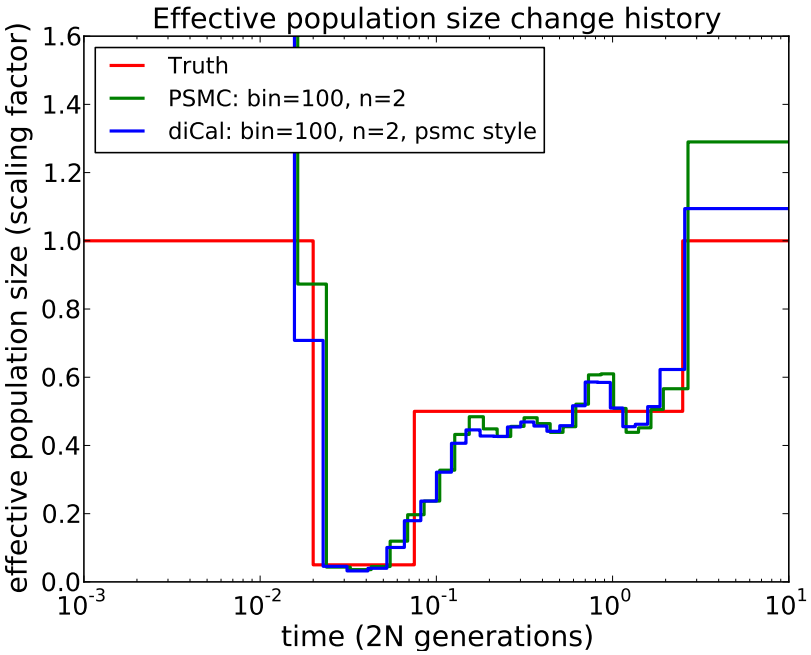


(a) *Binary (ordered)*: decoding with binning, comparing the true join on time from `ms` to the inference using the binary emission scheme.

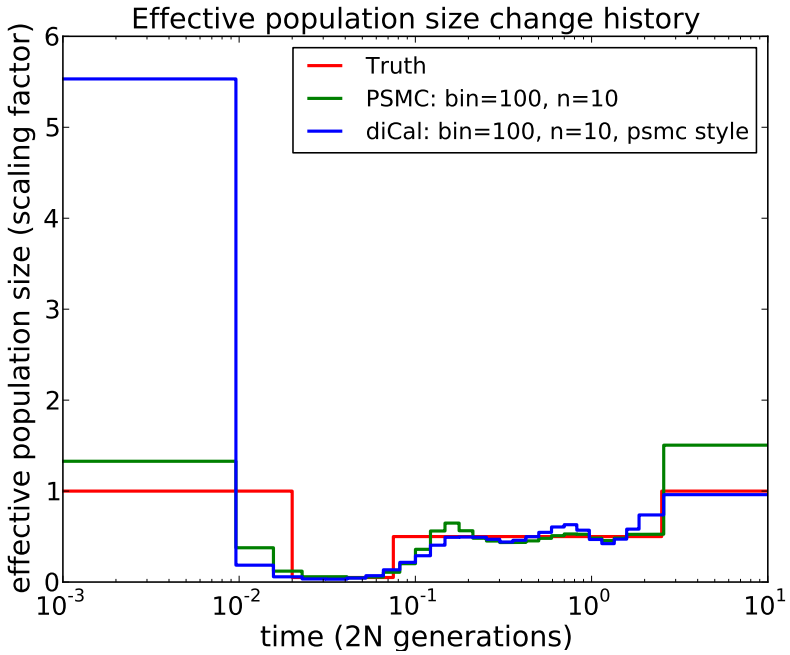


(b) *Poisson (unordered)*: decoding with binning, comparing the true join on time from `ms` to the inference using the Poisson emission scheme.

Figure 3.4: Decoding results using binning. The binary scheme is shown in (a) and the Poisson scheme in (b). In the mid-ancient past, both binning schemes perform well. In the recent past, it appears that the Poisson scheme has more resolution than the binary scheme.

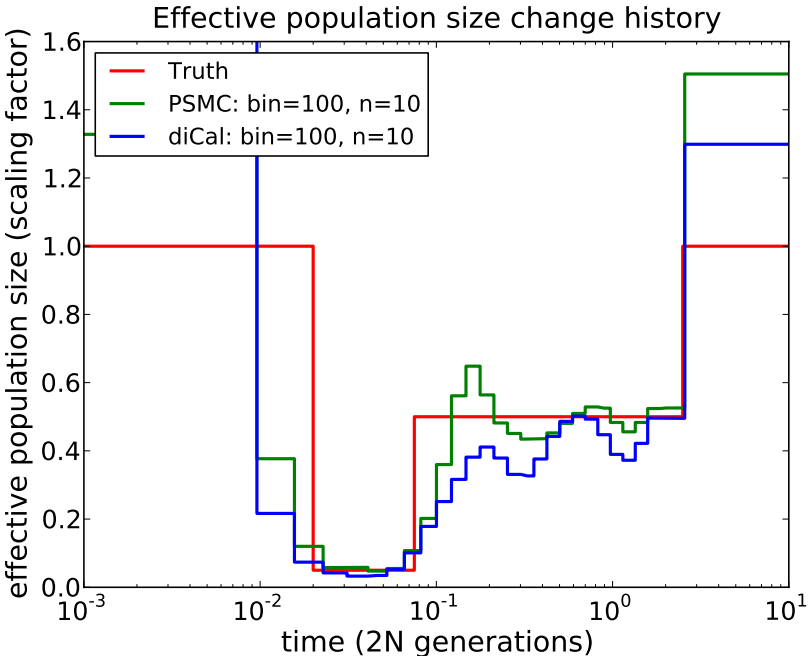


(a) PSMC vs. diCal results for $n = 2$, using the psmc-style binning scheme for diCal. Both methods show runaway behavior in the recent past, and are very similar throughout the rest of the history, with diCal being more accurate in the very ancient past.

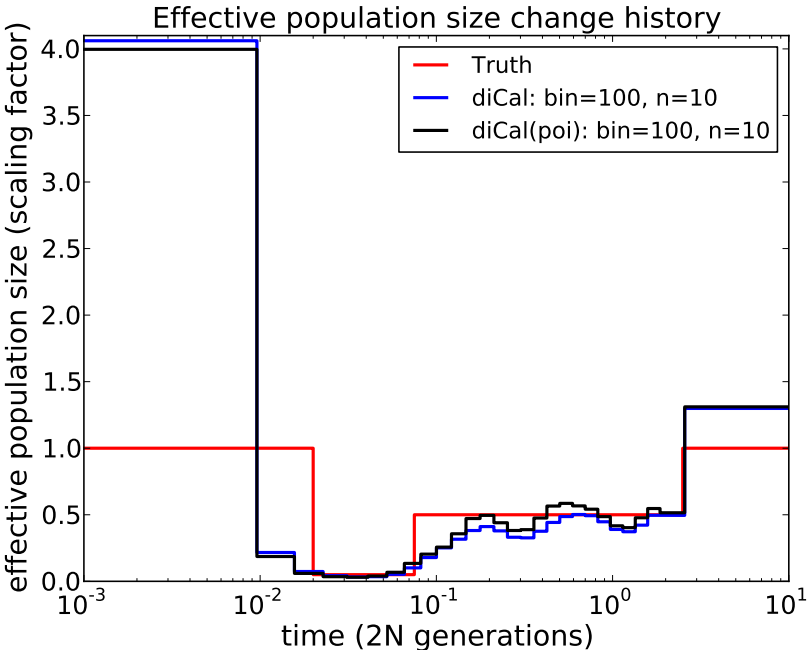


(b) PSMC vs. diCal results for $n = 10$, using the psmc-style binning scheme for diCal. In this case PSMC outperforms diCal in the recent past, and diCal shows a modest improvement over PSMC in the ancient past.

Figure 3.5: PSMC vs. diCal results, using the psmc-style binning scheme for diCal.



(a) PSMC vs. diCal results, with ordered binning for diCal, for $n = 10$. Similar to Figure 3.5(b), the PSMC outperforms diCal in the recent past.



(b) diCal results for $n = 10$, comparing the ordered (binary) and unordered (Poisson) emission schemes. There is very little difference.

Figure 3.6: Binning results for $n = 10$.

assign one population size parameter to each group. This solution has proved successful in some scenarios, but there is still the question of how to do the grouping. In this section, we will investigate how an optimal grouping can be found.

An ideal interval grouping will yield equal numbers of coalescent events in each group, so that the inference of each size parameter is supported by an equal number of observations. In our decodings however, we do not directly observe coalescent events, but we do have a proxy for when a lineage joins the tree (which represents a coalescent event). We will define an absorption "segment" as a stretch of loci where a lineage joins the tree at the same time. Switches in these segments represent recombination events. We use an adaptive procedure that aims to equalize the number of expected absorption segments within the time intervals spanned by each parameter.

More formally, first we discretize time into d intervals as we did before:

$$0 = t_0 < t_1 < \cdots < t_d = \infty,$$

and we would like to infer p size parameters. Then based on the initial decoding, we compute the expected number of segments, E_j, for each interval D_j, $j = 1, \cdots, d$. To compute E_j, we add up all the recombination transitions from any interval D_i $(i \neq j)$ to D_j:

$$E_j = \sum_{\ell=2}^{L} \sum_{i=1, i\neq j}^{d} p_r(i, j, \ell),$$

where $p_r(i, j, \ell)$ is the posterior probability of having a recombination transition from interval D_i to D_j, between locus $\ell - 1$ and ℓ. If we count up all these recombination transitions, that will be the expected number of times we *enter* interval D_j, which is a good approximation for the number of absorption segments in this interval.

Now we can find the desired number of expected segments per *parameter*:

$$E = \frac{1}{p} \sum_{j=1}^{d} E_j.$$

Our goal is to group the intervals such that each group has roughly E expected segments. Let q_k be the number of intervals spanned by parameter k, for $k = 1, \cdots, p$. So we want

$$\sum_{k=1}^{p} q_k = d.$$

To calculate the best set of q_k, we begin with the recent time intervals, adding intervals until

$$\sum_{i=1}^{q_1+q_2+\cdots+q_k} E_i \approx E \cdot k.$$

In other words, we keep adding on intervals for the k^{th} parameter until running total number of expected segments is as close as possible to $E \cdot k$. We also bound q_k by a minimum and

maximum number of intervals, so that each parameter gets a reasonable number of intervals. We use a minimum of 1 interval and a maximum of 6.

Finally, based on this optimal grouping, we compute the population size parameters in the M-step, and repeat this procedure for each E-step. The hope is that as the EM procedure converges, the grouping will converge as well. An example is shown in Figure 3.7 for $d = 25$ intervals and $p = 10$ parameters. After the very first E-step, the parameter grouping is 6+2+3+2+2+3+2+2+2+1, as shown on the x-axis for iteration 1. As the EM procedure goes on, we can see the expected segments shift and converge, to a final grouping of 6+1+1+1+1+1+1+3+9+1 at iteration 20. In the final plot, the population size change history is overlaid, and we can see that that many more coalescent events (represented by absorption segments) occur during the bottleneck. Thus we can see that the adaptive procedure is working, so that we will have more intervals grouped together during times when there are few coalescent events.

This adaptive procedure worked well in some cases, but did not always converge in others. Sometimes the grouping would alternate between two close groupings, which prevented the inference of the population sizes from converging. Depending on how we define "optimal", finding an optimal discretization is just as challenging as inferring the population sizes to begin with. Making such an adaptive procedure more robust is a promising area of research.

3.4 Using diCal to build local trees and infer population sizes

In much of population genetics, if we knew the full ARG, inference of many parameters of inference would be much easier. Estimating a series of local trees along the genome gets us most of the way toward full ARG inference. In this section we explore how diCal can be used to build local trees, and how those local trees can then be used to infer population sizes in a different way. Work in this section is joint with Jack Kamm.

3.4.1 Building local trees

In the case of $n = 2$, the posterior decoding of diCal or PSMC gives us a series of local trees at each locus. These trees are very simple, with two leaves for haplotypes h_1 and h_2. Given this series of trees, imagine adding another haplotype, h_3. At a given locus ℓ, say h_1 and h_2 coalesce at time T_ℓ. If h_3 joins either one of the other haplotypes at a time greater than T_ℓ, then the resulting tree has h_1 and h_2 coalescing first, then joining h_3. If h_3 joins either one of the other haplotypes before T_ℓ, then the tree will have that coalescent event first. We can continue building up the tree in this way, using the conditional sampling distribution to add each haplotype in turn. Figure 3.8 provides an illustration of this process.

This method works reasonably well when the mutation rate is high relative to the recombination rate. Figure 3.9 shows an example comparison between the true local tree on the left and the inferred tree on the right. Our method recovered the true topology correctly except

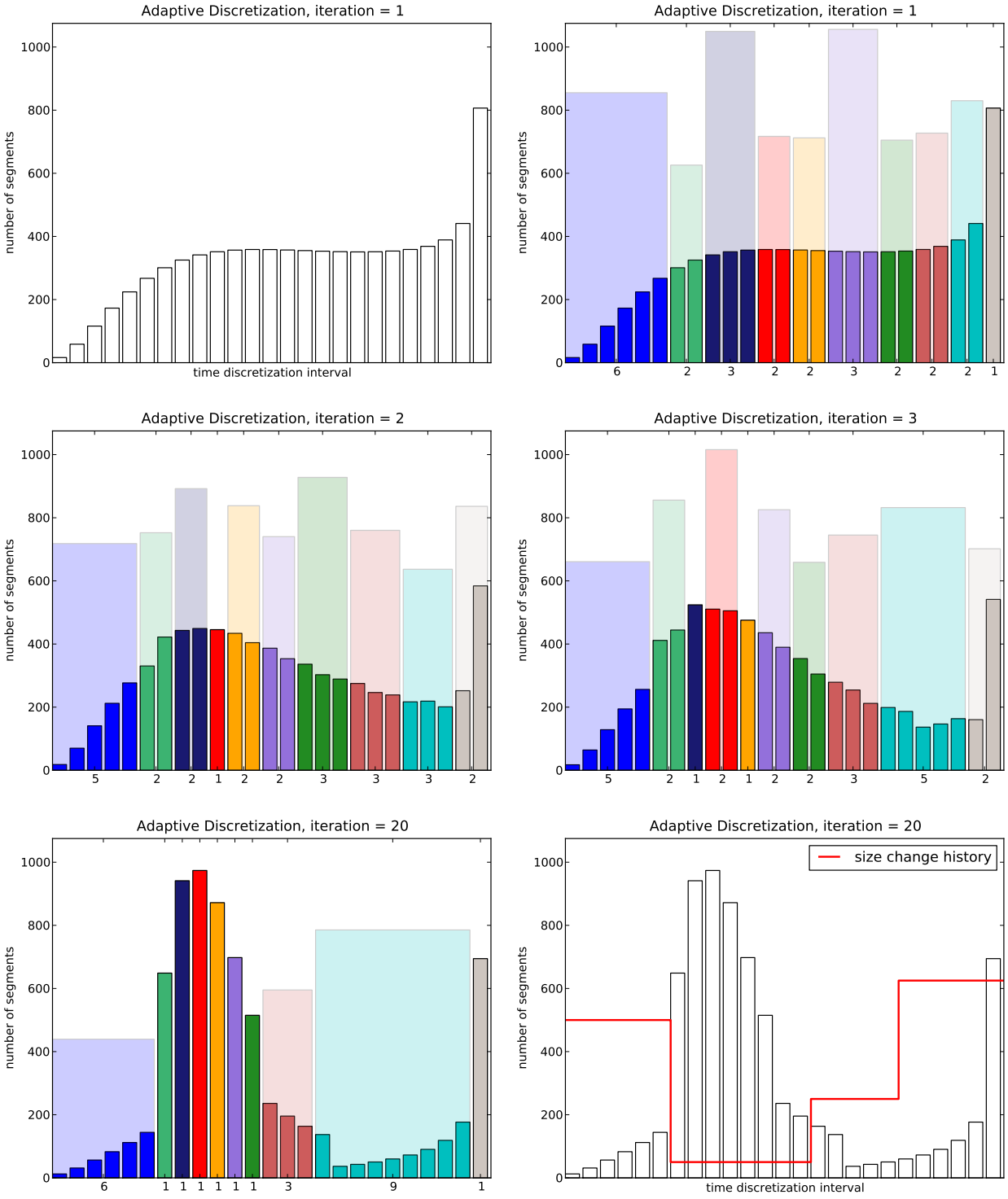


Figure 3.7: An example of the adaptive discretization procedure for $d = 25$ intervals and $p = 10$ parameters. In each plot, the number of segments for each time interval is plotted on the y-axis. Each color represents a different parameter, and the height of each shaded color block is the sum of the expected segments for that parameter. The ultimate goal is for the heights of all these shaded blocks to be the same.

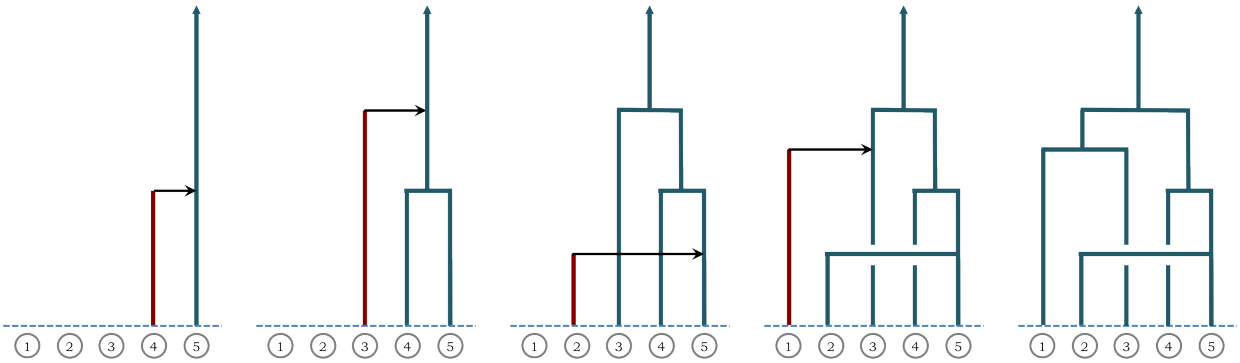


Figure 3.8: An illustration of how diCal can be used to build a local tree. First we run diCal on two lineages (4&5 here). Then we add lineage 3, and see where it joins the tree relative to 4&5. We continue adding haplotypes, using the posterior decoding from diCal to find what time and haplotype they join onto. We call this procedure the PAC-tree method, for Product of Approximate Conditionals, as opposed to the "leave-one-out" approach we used for parameter inference in chapter 2.

---

for the final coalescent event. The coalescence times are somewhat downwardly biased. In this example, we used a mutation rate of $\theta = 0.008$, and a recombination rate of $\rho = 0.001$. A more thorough investigation of how well this method works would help us determine areas for improvement, but is hammered by the difficultly of selecting an appropriate distance metric on trees.

We also wanted to investigate whether knowing the sequence of local trees would help us to infer ancestral population sizes. There are many ways the local trees could be used, but we chose to aggregate their information into the average number of remaining lineages as a function of time. Then we minimized the difference between this function and the *expected* number of remaining lineages under different population size histories, to find the best sizes. We used the integral of the absolute differences between the two functions as a distance metric. The results for different mutation rates are shown in Figure 3.10.

What is particularly interesting about Figure 3.10 is that the population size inference is not really much better when we use the true number of remaining lineages than when we use the inferred number of remaining lineages. It is possible this represents a limitation of using the number of remaining lineages for population size inference. In theory, this function should provide perfect information about the rate of coalescence back in time, which is what we need for population size inference. But in practice this function deviates from its expectation quite a bit due to the stochastic nature of the data. It is possible we need to use more data (either more loci or more lineages) to make this function more useful in practice.

Finally, we show two examples of using posterior mean decoding to smooth the function of the number of remaining lineages (this gives us a continuous distribution of coalescence times). These results are shown for a constant size history and a bottleneck in Figure 3.11.

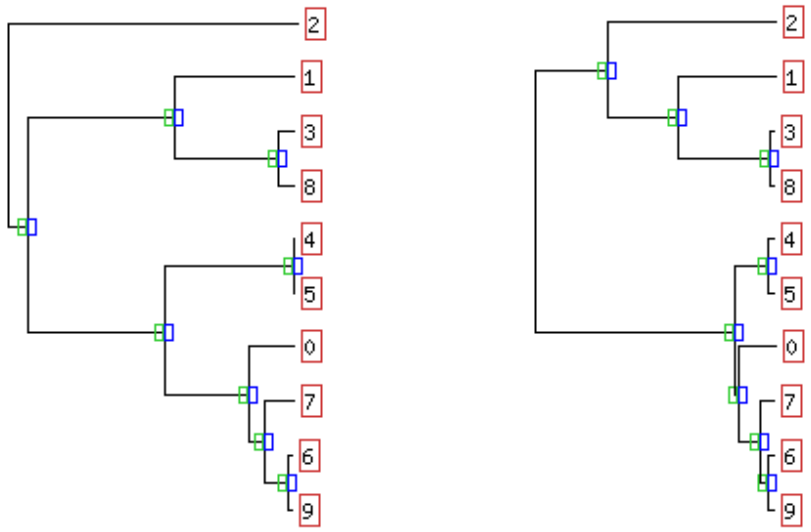


Figure 3.9: An example of local tree inference. The genealogy on the left is the true tree, and the genealogy on the right is our inferred tree. Except for the final coalescence event, we are able to reconstruct the topology correctly, although the coalescence times are generally somewhat too early.

3.5 Improvements to the diCal software

diCal was actively developed since its initial release, and this section describes some of the improvements to the software. Full details of how to run diCal can be found in the manual at `http://sourceforge.net/projects/dical/`. diCal is distributed open source under the FreeBSD (Berkeley Software Distribution) license.

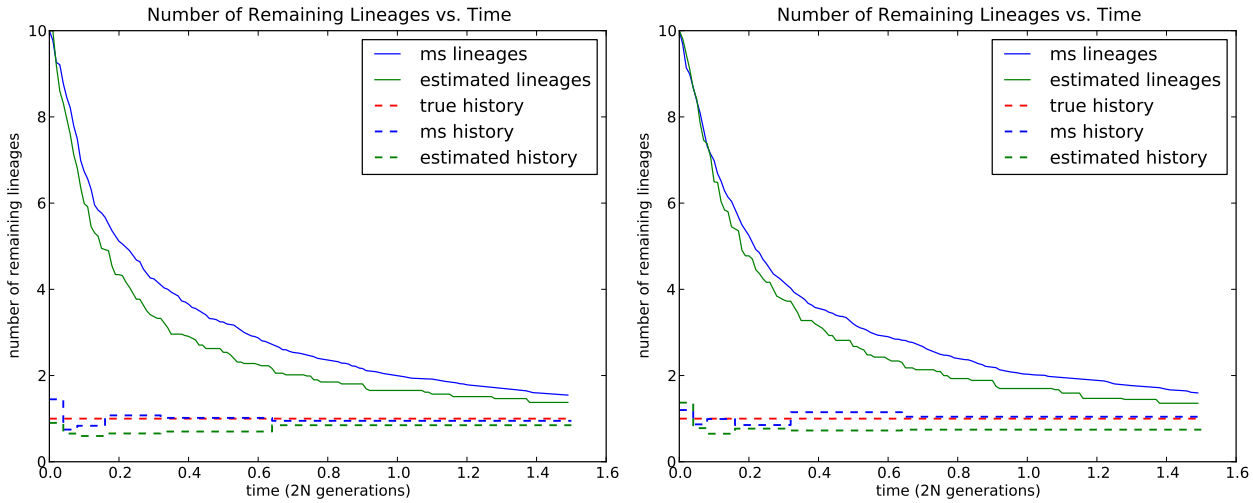
- The command line options were changed to be more consistent with PSMC [64]. An example is shown below.

  ```
  java -jar diCal.jar -F data.fa -I params.txt -n 4 -p "3+2+2+3" -t 2
  ```

  Where `-F` is the path to the input fasta file (must be phased), `-I` is the path to the parameter file, `-n` is the number of haploid sequences, `-p` is the parameter grouping, `-t` is the end time for the discretization in coalescent units. So in this example, there are $d = 10$ discretization intervals, grouped into 4 parameters (so 4 sizes will be inferred).

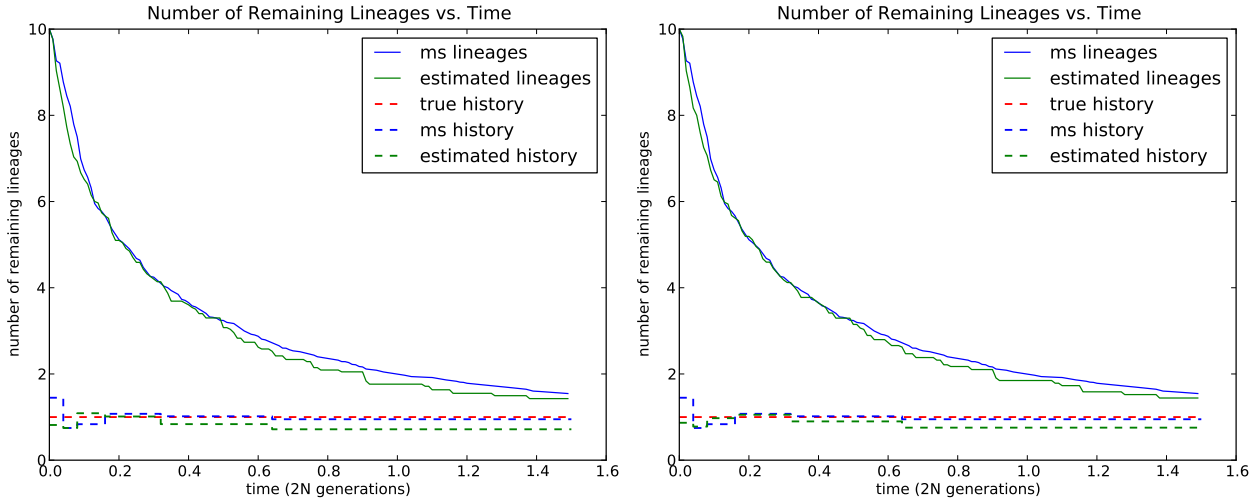
- Missing data (denoted by "N" or "n") is supported.

- In addition to fasta input files, a simplified version of the VCF file format is supported. A script is supplied to convert VCF files into this "stripped" VCF (SVCF) format. If this format is used, a reference fasta file must also be supplied. An example line of an



(a) $\theta = 0.0005$

(b) $\theta = 0.001$



(c) $\theta = 0.004$

(d) $\theta = 0.008$

Figure 3.10: These figures show the results of using our local tree inference method to infer ancestral population sizes. The solid lines represent the number of remaining lineages as a function of time, which starts at $n = 10$ at time 0, and decreases monotonically to 1 at time ∞. The dashed lines represent the population sizes over time. The solid blue line is the true average number of lineages, and the dashed red line is the true population size history we used for simulation (constant in this case). The dashed blue line is the history we infer if we use the *true* number of lineages. The solid green line is the number of remaining lineages we obtain if we use the local trees estimated from diCal as described above. And finally the dashed green line is the history we infer using these estimates. We can see that as the mutation rate θ increases, our inference becomes better.

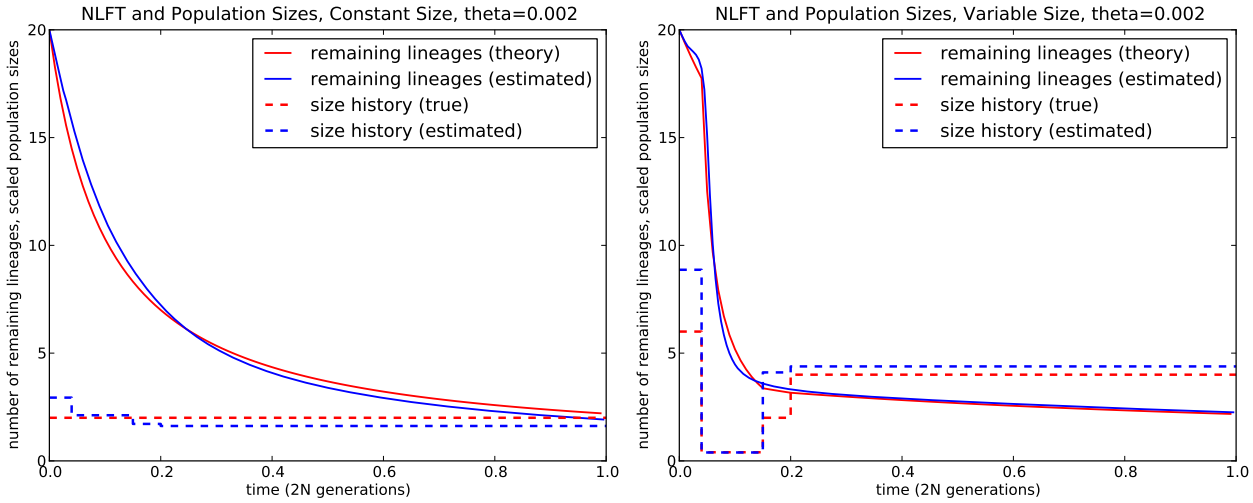


Figure 3.11: Inference results for a constant history and a bottleneck history, for $n = 20$ and $\theta = 0.002$ (twice the recombination rate). The theoretical number of remaining lineages is shown with the solid red line, and the inferred number of remaining lineages is shown with the solid blue line. Posterior mean decoding was used to smooth this function. For the bottleneck history on the right, we can see that the inferred number of lineages does not initially decrease quickly enough, which results in an overestimate of the most recent population size (dashed blue line). The discretization times were chosen to be the same as the bottleneck for proof of concept. We can see that even though the number of lineages is not inferred quite correctly during the bottleneck, the bottleneck size is quite well-estimated. This is reflective of a pattern of more accurate size inference during times of lower population size, when there are more coalescent events.

SVCF file is shown below, with one column for the chromosome or scaffold name, one column for the location of the SNP, and then the phased SNP data for each haplotype.

```
chrom1 395049 ACAAAACAAA
```

- Multiple chromosomes/scaffolds are supported.
- The E-step computation is parallelized, so that each haplotype can potentially be "left-out" at the same time. The number of cores can be specified as anything from 1 to n depending on memory and speed requirements.
- Five different types of decodings are supported, as described below.

    1. *posterior decoding*: Each line of output contains the locus, the posterior absorption time, and the posterior absorption haplotype index (with respect to the order the haplotypes appear in the fasta or VCF file).

2. *Viterbi decoding*: Similar format to the posterior decoding, with the best path absorption time and haplotype printed.
3. *posterior mean time*: For each time interval, the haplotype is marginalized out, then the posterior mean time (i.e. time a lineage joins the genealogy) is calculated.
4. *posterior decoding time*: Marginalize out the haplotypes, then choose the join-on time with the highest probability.
5. *posterior decoding time and probability*: Same as 4, but also prints out the highest probability.

- Users have the option to specify their own time discretization. Users can also specify a demographic history and simply print a decoding without doing any inference.
- The number of expected segments can be printed, which can help users detect when the method might be experiencing runaway behavior (very few segments in some intervals).

Chapter 4

Deep learning for population genetics

The coalescent is an essential tool for developing likelihood-based methods for population genetic inference. However, in chapter 2 and chapter 3, the runtime of computing even approximate likelihoods was the main barrier to the wider application of coalescent HMMs. Further, for many coalescent models, the full likelihood remains unknown, and even in cases where it is known, sufficient statistics for parameters of interest may not exist.

In this chapter we present an alternative new inference method for population genetics that leverages the power of a deep learning framework. Deep learning, a powerful modern technique from the machine learning literature, provides a principled way of learning parameters and informative features of data, even when few training datasets are used. Inspired by neural networks, deep architectures use several layers of hidden nodes to learn a rich class of functions from the input (summary statistics) to the output (parameters of interest).

Approximate Bayesian Computation (ABC) is an alternative likelihood-free inference method that has become a very useful for population genetic inference. However, this type of algorithm has several drawbacks, including poor performance in the presence of correlated summary statistics. ABC uses many simulated datasets to find those closest to the target dataset, then retains the corresponding parameters to compute an approximate posterior distribution for the parameters of interest. In contrast to ABC, deep learning requires no rejection step, does not rely on a prior for parameter estimation, and is robust to the addition of uninformative statistics.

Although our deep learning method is very general, here we focus on applying the method to the challenging problem of explicit joint inference of demography (in the form of a bottleneck) and natural selection. Our method is able to separate the global nature of demographic factors from the local nature of selection, creating a flexible and robust inference framework. Studying demography and selection is motivated by *Drosophila*, where pervasive selection confounds demographic analysis. Using hundreds of summary statistics, we apply our method to 22 African *Drosophila melanogaster* haplotypes from the DPGP [94] to infer both the overall demography, and regions of the genome under selection. We find that previous *Drosophila* demography results were biased by selection, and provide an alternative history to be used in subsequent analysis.

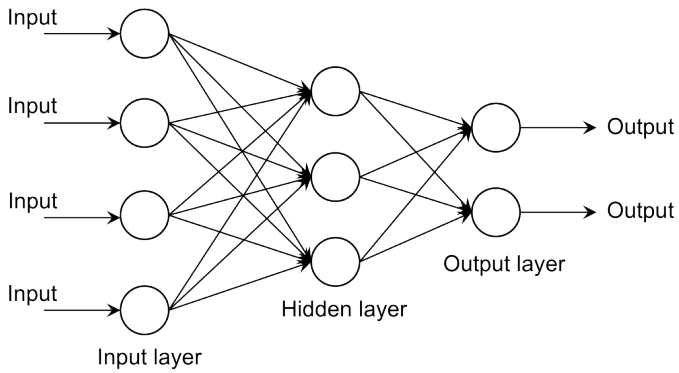


Figure 4.1: An example of a classical neural network. The single hidden layer serves to learn informative combinations of the inputs, remove correlations, and typically reduce the dimension of the data. After the optimal weight on each connecting arrow is learned through labeled training data, unlabeled data can be fed through the network to learn the output parameters.

4.1 Deep learning background

Deep learning has its beginnings in neural networks, which were originally inspired by the way neurons are connected in the brain [48]. Neural networks have been used to learn complex functions between input data and output parameters in the absence of a model. A classical neural network is shown in Figure 4.1.

It has been shown in the *universal approximation theorem* that a neural network with a single hidden layer with a finite number of units can approximate any continuous function under mild assumptions [17, 49]. However, finite can still be very large, and it can be challenging to learn and interpret the weights of such a network. So as learning problems became more complex, it became desirable to train networks with more hidden layers. Due to the non-convex nature of the function to optimize the weights, these "deep" networks proved difficult to train. Deep learning stagnated until a breakthrough in 2006 [45]. Figure 4.2 shows a plot of the number of papers that mention deep learning over time, highlighting this breakthrough.

Since then, deep learning has broken many machine learning records for classification problems, especially in the fields of vision (example: [61]) and speech recognition (example: [33]). Many variations have been developed, including *dropout*, which attempts to learn better and more robust features of the data (see [18, 46]). Deep learning has also been applied to problems in biology (examples: [63, 119]), but had not been used for population genetics before.

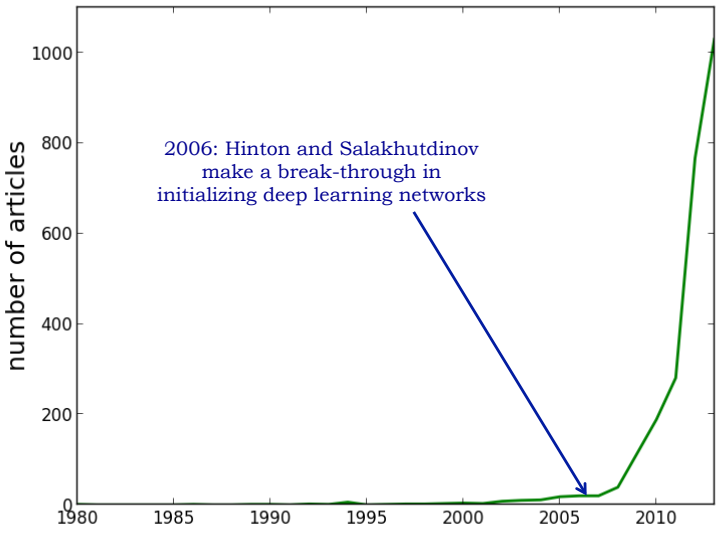


Figure 4.2: Deep learning papers over time. The first mention of deep learning was in 1980, but these deep networks proved difficult to train until 2006, when Hinton and Salakhutdinov [45] developed a novel pre-training method.

4.2 ABC background

Rejection-based methods have been used since the late 1990's (see [96, 109]) to estimate population genetic parameters when the likelihood is difficult to compute. Early improvements to ABC quickly helped make it a popular method for a variety of scenarios (see [4] for a good introduction). ABC works by simulating many datasets under a prior for the parameters of interest. Then these datasets are reduced to a vector of summary statistics that are ideally informative for the parameters. The summary statistics that are closest to the summary statistics for the target dataset are retained, and the corresponding parameters used to estimate the desired posterior distributions. The definition of "close" is usually determined by a Euclidean distance metric on the summary statistic vectors, which can create biases if statistics are not properly normalized or have different variances.

A second problem with ABC is its inability to handle uninformative or weakly informative summary statistics. Intuitively, this is because these statistics add noise to the distance metric between two datasets; two datasets simulated under similar parameters may have some uninformative statistics that are far apart, or two datasets simulated under very different parameters may have some uninformative statistics that are close together. The distance metric can be further biased by differences in the magnitude of the summary statistics. Another major problem with ABC is the rejection step, which does not make optimal use of the datasets which are not retained. The more statistics and parameters used, the more datasets must be simulated and rejected to properly explore the space, making the interaction between these two issues even more problematic. One final issue with ABC is the black-box nature of the output. Given the distances between the simulated datasets and the target dataset, and the posterior, there is no clear way to tell which statistics were the most informative.

To tackle the problem of adding summary statistics, many methods for dimensionality reduction or selecting summary statistics wisely have been proposed (see [1, 26, 55, 85], and

[8] for a dimensionality reduction comparison). However, simple reductions cannot always learn subtle relationships between the data and the parameters. Expert pruning of statistics helps some methods, but given the lack of sufficient statistics, valuable information can be eliminated, especially when trying to infer many parameters. In Blum and François [7], a dimensionality reduction step is performed on the summary statistics via a neural network similar to Figure 4.1. This reduction is similar in spirit to the work presented here, although there are many algorithmic and application differences.

To address the problem of rejecting datasets, different weighting approaches have been proposed (see [7] for a good example of how the estimation error changes as fewer datasets are rejected). The idea is to keep more datasets, but then weight each retained dataset by the its distance to the target dataset. However, few approaches utilize all the datasets in this way, and the most popular implementation of ABC (*ABCtoolbox* [115]) typically still rejects most of the simulated datasets by default.

4.3 Deep learning theory

In this section we provide the theory behind training deep networks. The notation in this section is based off of the notation in [83]. Let $x^{(i)}$ be the vector of summary statistics for dataset i, and $y^{(i)}$ be the vector of parameters that dataset i was simulated under. If we have m such datasets, then together $\{(x^{(1)}, y^{(1)}), \cdots, (x^{(m)}, y^{(m)})\}$ form the training data that will be used to learn the function from statistics to parameters. Deep neural networks are a way to express this type of complex, non-linear function. The first layer of the network is the input data, the next layers are the "hidden layers" of the network, and the final layer represents the network's prediction of the parameters of interest.

4.3.1 Cost function for a deep network

Let the weights between layer ℓ and layer $\ell+1$ be $W^{(\ell)}$, where $W^{(\ell)}_{jk}$ is the weight associated with the connection between node j in layer ℓ and node k in layer $\ell+1$. Let the biases for layer ℓ be $b^{(\ell)}$. The total number of layers (including the input and output layers) is L, and the number of hidden units in layer ℓ is denoted u_ℓ. The main goal is to learn the weights that best describe the function between the inputs and the outputs.

To learn this function, we first describe how the values of the hidden nodes are computed, given a trial weight vector. The value of hidden node j in layer ℓ is denoted $a^{(\ell)}_j$, where

$$a^{(\ell)}_j = f\big(z^{(\ell)}_j\big), \quad \text{where} \quad z = W^{(\ell-1)}_j \cdot a^{(\ell-1)} + b^{(\ell-1)}_j,$$

$W^{(\ell-1)}_j$ is the j^{th} *column* of the weight matrix $W^{(\ell-1)}$ (i.e. all the weights going into node j of layer $\ell-1$), $a^{(\ell-1)}$ are the values of all the (hidden) nodes in the previous layer, and

$$f(z) = \frac{1}{1+\exp(-z)} \tag{4.1}$$

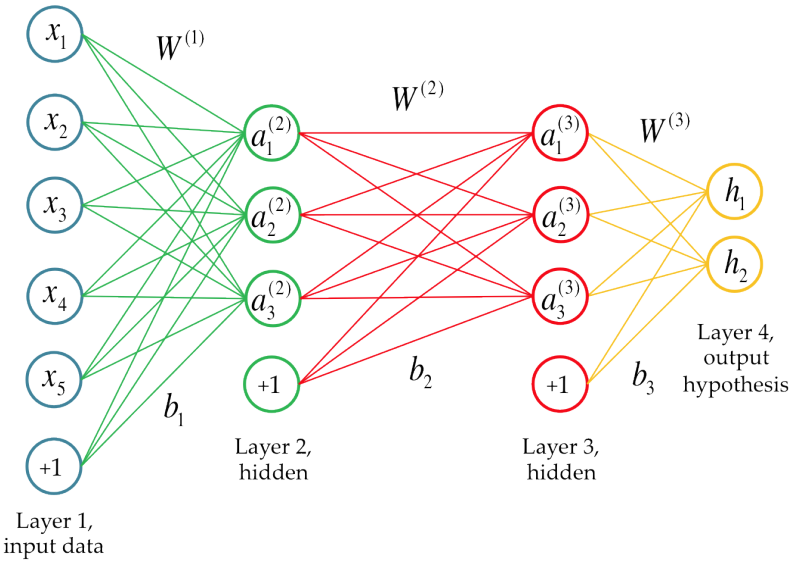


Figure 4.3: Example of a neural network with two hidden layers. The first layer is the input data (each dataset has 5 statistics), and the last layer predicts the 2 parameters of interest. The last node in each input layer (+1) represents the bias term. Here the number of layers $L = 4$, and the number of nodes (units) in each layer is $u_1 = 5$, $u_2 = 3$, $u_3 = 3$, and $u_4 = 2$ (these counts exclude the biases).

---

is the *activation function*. Here we use a logistic function, but other functions can be used. Another common activation function is the hyperbolic tangent function. An example deep network is shown in Figure 4.3.

Therefore given the input data and a set of weights, we can *feed forward* to learn the values of all hidden nodes, and a prediction of the output parameters. These predictions are usually denoted by $h_{W,b}(x^{(i)})$ for our *hypothesis* for dataset i, based on all the weights W and biases b. See Section 4.3.3 for different ways to compute the hypothesis function. To find the best weights, we define a loss function based on the L_2 norm between this hypothesis and the true parameters. This loss function is given by

$$J(W,b) = \frac{1}{m}\sum_{i=1}^{m}\frac{1}{2}\left|\left|h_{W,b}(x^{(i)}) - y^{(i)}\right|\right|^2.$$

The goal of deep learning is to find the weights (the set of W's and b's) that minimize this loss function. To efficiently find these optimal weights, we can use *backpropagation* to find the gradient. The intuition behind this approach is that once we have found the hypothesis, we then want to see how much each of the weights contributed to any differences between the hypothesis and the truth. Therefore we start at the last hidden layer, see how much each of those weights contributed, then work our way backwards, using the gradient of the

previous layer to compute the gradient of the next layer. For this we need to compute the partial derivatives of the cost function with respect to each weight.

Consider a single training example x, with associated parameters y. Let the cost for this dataset be one term of the sum above

$$J(W, b, x, y) = \frac{1}{2}\big|\big|h_{W,b}(x) - y\big|\big|^2.$$

First, we take the derivative of the cost function with respect to $z_j^{(L)}$ (the input to the activation function for output node j), which we denote $\delta_j^{(L)}$. Note that the j^{th} node of $h_{W,b}(x)$ is $f(z_j^{(L)})$.

$$\delta_j^{(L)} = \big(h_{W,b}(x) - y\big) \cdot f'\big(z_j^{(L)}\big).$$

Based on this initialization, we can recursively compute all the δ variables:

$$\delta_j^{(\ell)} = \left(\sum_{k=1}^{u_{\ell+1}} W_{jk}^{(\ell)} \delta_k^{(\ell+1)}\right) f'\big(z_k^{(\ell)}\big).$$

Now we can use the δ variables to recursively compute the partial derivatives for one dataset:

$$\frac{\partial J(W, b, x, y)}{\partial W_{jk}^{(\ell)}} = a_j^{(\ell)} \cdot \delta_k^{(\ell+1)} \quad \text{and}$$
$$\frac{\partial J(W, b, x, y)}{\partial b_k^{(\ell)}} = \delta_k^{(\ell+1)}.$$

Finally, putting all the datasets together we get

$$\frac{\partial J(W, b)}{\partial W_{jk}^{(\ell)}} = \sum_{i=1}^{m} \frac{\partial J(W, b, x^{(i)}, y^{(i)})}{\partial W_{jk}^{(\ell)}} \quad \text{and} \quad \frac{\partial J(W, b)}{\partial b_k^{(\ell)}} = \sum_{i=1}^{m} \frac{\partial J(W, b, x^{(i)}, y^{(i)})}{\partial b_k^{(\ell)}}.$$

Since we can compute the derivatives using this backpropagation algorithm, we can use the LBFGS optimization routine (as implemented in [20]) to find the weights that minimize the cost function.

4.3.2 Unsupervised pre-training using autoencoders

It is possible to train a deep network by attempting to minimize the cost function described above directly, but in practice, this proved difficult due to the high-dimensionality and non-convexity of the optimization problem. Initializing the weights randomly before training resulted in poor local minima. The breakthrough in [45] sought to initialize the weights in a more informed way, using an unsupervised pre-training routine. Unsupervised training ignores the output (often called the "labels") and attempts to learn as much as possible about the structure of the data on its own. PCA is an example of unsupervised learning. In

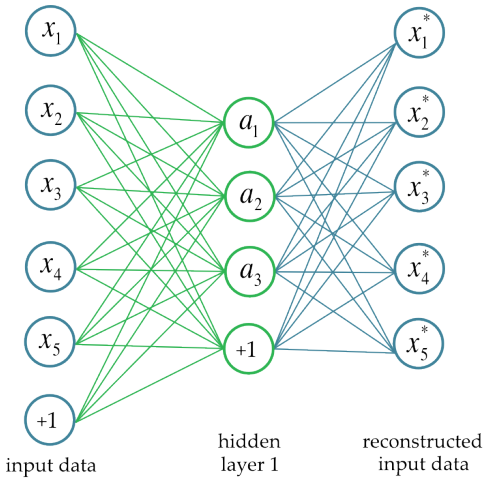


Figure 4.4: An example of an autoencoder. The input data (x) is projected into a (usually) lower dimension (a), then reconstructed ($\hat{x}$). The weights of an autoencoder are optimized such that the difference between the reconstructed data and the original data is minimal.

---

[45], the unsupervised pre-training step uses an *autoencoder* to try to learn the best function from the data to itself, after it has gone through a dimensionality reduction step (which can be thought of as trying to compress the data, then reconstruct it with minimal loss). Autoencoding provides a way to initialize the weights of a deep network that will ideally be be close to optimal for the supervised learning step as well. See Figure 4.4 for a diagram of an autoencoder.

Training an autoencoder is an optimization procedure in itself. As before, let $W^{(1)}$ be the vector of weights connecting the input x to the hidden layer a, and $W^{(2)}$ be the vector of weights connecting a to the output layer $\hat{x}$, which in this case should be as close as possible to the original input data. We again typically use the logistic function as our activation function f as described above, so we can compute the output using:

$$a_j = f(W_j^{(1)} \cdot x + b_1) \quad \text{and} \quad \hat{x}_k = f(W_k^{(2)} \cdot a + b_2).$$

If a linear activation function is used instead of a logistic function, the hidden layer becomes the principle components of the data. This makes dimensionality reduction with an autoencoder similar in spirit to PCA, which has been used frequently in genetic analysis (see [84] for an example). However, the non-linear nature of an autoencoder has been shown to reconstruct complex data more accurately than PCA. Using backpropagation as we did before, we can minimize the following autoencoder cost function using all m input datasets:

$$A(W, b) = \frac{1}{m} \sum_{i=1}^{m} \frac{1}{2} \left|\left| \hat{x}^{(i)} - x^{(i)} \right|\right|^2.$$

The resulting weights $W^{(1)^*}$ will then be used to initialize the weights between the first and second layers of our deep network. The weights $W^{(2)^*}$ are discarded. To initialize the rest of the weights, we can repeat the autoencoder procedure, but this time we will use the hidden layer a^* as our input data, and feed it through the next hidden layer. In this way we can use "stacked" autoencoders to initialize all the weights of the deep network. Finally, the supervised training procedure described in the previous section can be used to *fine-tune* the weights to obtain the best function from the inputs to the parameters of interest.

When the number of hidden units is large, we would like to constrain an autoencoder such that only a fraction of the hidden units are "firing" at any given time. This corresponds to the idea that only a subset of the neurons in our brains are firing at once, depending on the input stimulus. To create a similar phenomenon for an autoencoder, we can create a *sparsity* constraint that ensure the activation of most of the nodes is close to 0, and the activation of a small fraction, ρ, of nodes is close to 1. Let $\hat{\rho}_j$ be the average activation of the hidden node j:

$$\hat{\rho}_j = \frac{1}{m}\sum_{i=1}^{m} a_j(x^{(i)}),$$

where $a_j(x^{(i)})$ is the value of the j^{th} hidden node when activated with dataset $x^{(i)}$. To ensure sparsity, we would like $\hat{\rho}_j$ to be close to ρ, our desired fraction of active nodes. This can be accomplished by minimizing the KL divergence:

$$\sum_{j=1}^{u_2} \mathrm{KL}(\rho||\hat{\rho}_j) = \sum_{j=1}^{u_2} \rho \log\frac{\rho}{\hat{\rho}_j} + (1-\rho)\log\frac{1-\rho}{1-\hat{\rho}_j},$$

where u is the number of units in the hidden layer. We multiply this term by a sparsity weight β. In addition, a regularization term is included, which prevents the magnitude of the weights from becoming too large. To accomplish this, we add a penalty to the cost function that is the sum of the squares of all weights (excluding the biases), weighted by a well-chosen constant λ, which is often called the *weight decay parameter*. Including both sparsity and regularization, our final autoencoder cost becomes:

$$A_\lambda(W,b) = \frac{1}{m}\sum_{i=1}^{m}\frac{1}{2}\left|\left|\hat{x}^{(i)} - x^{(i)}\right|\right|^2 + \beta\sum_{j=1}^{u_2}\mathrm{KL}(\rho||\hat{\rho}_j) + \frac{\lambda}{2}\sum_{\ell=1}^{2}\sum_{j=1}^{u_{\ell-1}}\sum_{k=1}^{u_\ell}\left(W_{jk}^{(\ell-1)}\right)^2.$$

We also regularize the weights on the last layer during fine-tuning, so our deep learning cost function becomes:

$$J_\lambda(W,b) = \frac{1}{m}\sum_{i=1}^{m}\frac{1}{2}\left|\left|h_{W,b}(x^{(i)}) - y^{(i)}\right|\right|^2 + \frac{\lambda}{2}\sum_{j=1}^{u_{L-1}}\sum_{k=1}^{u_L}\left(W_{jk}^{(L-1)}\right)^2.$$

Type	Classifier	Error %	Year
Linear classifier	Pairwise linear classifier	7.6	1998, [62]
Non-Linear classifier	40 PCA + quadratic classifier	3.3	1998, [62]
Boosted stumps	Product of stumps on Haar features	0.87	2009, [56]
Support vector machine	Virtual SVM, deg-9 poly, 2-pixel jittered	0.56	2002, [19]
K-nearest neighbors	K-NN with non-linear deformation	0.52	2007, [58]
Neural network (NN)	Layers: 784-2500-2000-1500-1000-500-10	0.35	2010, [13]
Convolutional NN	35 conv. net, 1-20-P-40-P-150-10	0.23	2012, [12]

Table 4.1: Table of classification results on the MNIST hand-written digit database, from [77]. The neural network (NN) method is a classical example of deep learning, here with 6 layers, whose sizes are shown. The dimension of each image is 784, then the sizes of the hidden layers decrease from 2500 to 500, and the final layer is of size 10 for each of the classes for the digits from 0-9.

4.3.3 The final layer: parameter estimation vs. classification

In population genetics, often we want to estimate a continuous parameter. To compute our hypothesis for a parameter of interest, based on a set of weights, we could use a logistic activation function like Equation 4.1, as we did for the other layers. However, a logistic function often implies a binary scenario, which we do not have in the case of a continuous parameter. Instead, we use a linear activation function, so in the case of a single parameter, our hypothesis for dataset i becomes

$$h_{W,b}^{\text{linear}}(x^{(i)}) = W^{(L-1)} \cdot a^{(L-1)} + b^{(L-1)}.$$

In other words, it is the dot product of the activations of the final hidden layer and the weights that connect the final hidden layer to the parameters.

However, traditionally deep learning has been used for classification, where the goal is to assign each dataset to one of a discrete number of classes. For example, hand-written digit classification (with 10 classes for the digits 0-9) has been a popular machine-learning task for comparing different algorithms. Table 4.1 shows a comparison of different methods that were used to classify the MNIST hand-written digit dataset.

For such classification results, if we had two classes, we could use logistic regression to find the probability a dataset was assigned to each class. With more than two classes, we can extend this concept ans use *softmax regression* to assign a probability to each class. If we have k classes labeled $1, \cdots, k$, we can define our hypothesis as follows

$$h_{W,b}^{\text{softmax}}(x^{(i)}) = \begin{bmatrix} p(y^{(i)} = 1 | x^{(i)}; W, b) \\ p(y^{(i)} = 2 | x^{(i)}; W, b) \\ \cdots \\ p(y^{(i)} = k | x^{(i)}; W, b) \end{bmatrix} = \frac{1}{Z} \begin{bmatrix} \exp\{W_1^{(L-1)} \cdot a^{(L-1)} + b_1^{(L-1)}\} \\ \exp\{W_2^{(L-1)} \cdot a^{(L-1)} + b_2^{(L-1)}\} \\ \cdots \\ \exp\{W_k^{(L-1)} \cdot a^{(L-1)} + b_k^{(L-1)}\} \end{bmatrix},$$

where Z is the sum of all the entries, so that our probabilities sum to 1. Using this formulation, we can define our classification cost function:

$$J_{\lambda}^{\text{softmax}}(W,b) = -\frac{1}{m}\sum_{i=1}^{m}\sum_{j=1}^{k}\mathbb{1}\{y^{(i)}=k\}\log p(y^{(i)}=j|x^{(i)};W,b) + \frac{\lambda}{2}\sum_{s=1}^{u_{L-1}}\sum_{t=1}^{u_L}\left(W_{st}^{(L-1)}\right)^2.$$

Intuitively, we can think about this cost function as making the log probability of the correct class as close to 0 as possible.

4.3.4 Image example

Deep learning has proven extremely successful for images, and this section provides a concrete example to help illustrate the goals of deep learning. In Figure 4.5, we can see that our data is a set of images of faces. We want to learn three parameters, corresponding to y_1, y_2, y_3. The first two are discrete with two classes each: whether or not the person is wearing glasses and whether or not they are smiling. Eye size, represented by y_3, is a continuous parameter.

During each pre-training step, the hidden layers learned features that were easy to interpret. In the first hidden layer, the autoencoder learned "edges". This is essentially a basis of features that can be combined to create images. In the second hidden layer, the next autoencoder learned higher-level features; a mouth, an eye, a nose, and an ear. These can be thought of as more sophisticated building blocks with which to build images of faces. Now we can see that the parameters of interest can be easily found from these second-order features. To see whether the person is wearing glass and to determine their eye size, we can use they eye feature. To see if a person is smiling, we can use the mouth feature. This is a toy example, but it illustrates the goals of deep learning: to automatically learn features of the data that can provide answers to our questions.

4.4 Deep learning for demography and selection

The population genetics problem we wish to solve is jointly estimating demography and selection (see [65] for a recent review of this topic). One reason this problem is difficult is that demography (for example, a bottleneck) and selection can leave similar signals in the genome. Untangling the two factors directly has rarely been attempted - most methods that estimate selection try to demonstrate robustness to demographic scenarios, rather than estimating demographic parameters jointly with selection. Our analysis is motivated by *Drosophila*, where pervasive selection has confounded demographic inference. There is one example of a demographic history for *Drosophila*, shown in Figure 4.6 [21]. Here we focus on 22 *Drosophila melanogaster* genomes from Rwanda, Africa [94].

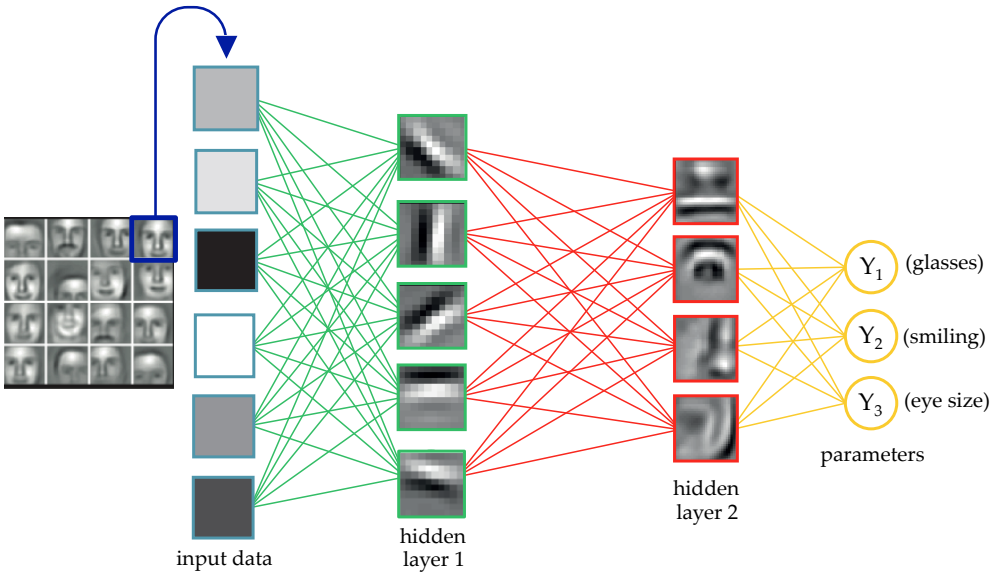


Figure 4.5: An example of the features learned from images of faces. In the first hidden layer, the autoencoder has learned a basis of "edges". In the second hidden layer, the autoencoder has learned higher-order features which can be combined to create faces. These features are also informative for our parameters of interest in the last layer. Images adapted from [54].

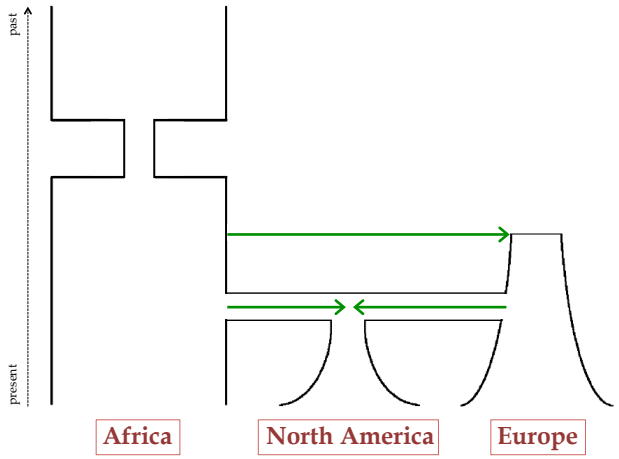


Figure 4.6: The demographic history for *Drosophila* inferred by [21], modified from their paper, but still not to scale (the bottleneck is much more severe than shown).

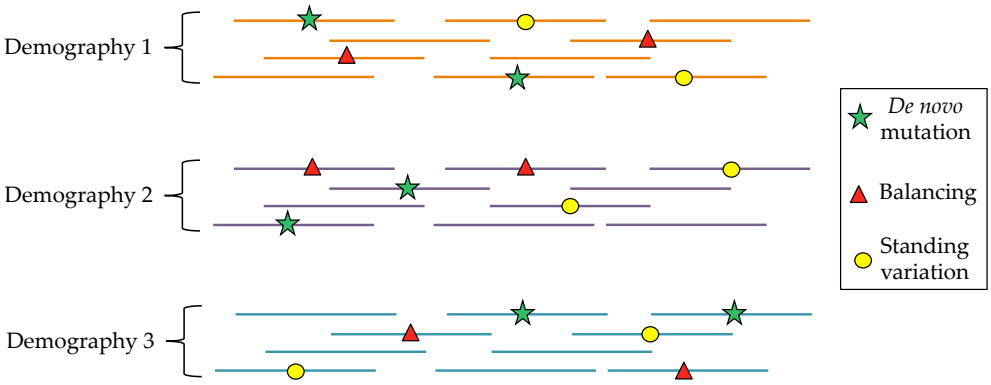


Figure 4.7: Input data for the demography and selection scenario. For each demographic history (bottleneck), we simulated many different genomic regions. Each region can either have no selection, one site with an *de novo* mutation under positive selection, one site under balancing selection, or one standing variant under positive selection.

4.4.1 Simulating a realistic dataset

To create a simulated dataset that is appropriate for our scenario of interest, we first define the parameters we would like to estimate. For simplicity, and to fit with the African *Drosophila* history in Figure 4.6, we define a demographic history to be a 3-parameter bottleneck (a recent population size, a bottleneck size, and an ancient size). And we define a region as belonging to 4 different selection classes: no selection (neutral), positive directional selection, balancing selection, or selection on standing variation. See [23] for a more complete analysis of the different types of selection in *Drosophila*. To make our data fit with the idea of demography affecting the entire genome and selection affecting a particular region, we simulate many genomic regions under the same bottleneck, but the selection class for each one is chosen independently. See Figure 4.7 for an illustration of the data.

To simulate this dataset, we used `msms` [25]. To make our simulated data as close to the real data as possible, we simulated $n = 22$ haplotpyes to match the African *Drosophila* dataset. We repeated the following procedure 1000 times. First we selected three population sizes, then simulated 150 regions with these sizes. Each region was 100kb, with one of three selection schemes, or neutral. We used a baseline effective population size $N_{\text{ref}} = 100,000$, a per base, per generation mutation rate $\mu = 8.4 \times 10^{-9}$ [39], and a per base, per generation recombination rate $r = 3.423 \times 10^{-8}$ (based off of recombination and mutation rate estimates in [10]). We used a generation time of 10 generations per year. Based on the history in [21], we used the time of the bottleneck to be $T_A = 237,227$ in years, and to last for 1000 generations. Our effective population size parameters of interest and their prior distributions are below:

1. Recent effective population size scaling factor: $\lambda_1 \sim \text{Unif}(1, 10)$

2. Bottleneck effective population size scaling factor: $\lambda_2 \sim \text{Unif}(0.05, 0.5)$

3. Ancient effective population size scaling factor: $\lambda_3 \sim \text{Unif}(1, 10)$

For the selection classes, the different types are shown below:

- **Class 0**: no selection, neutral. However, we did simulated "neutral" datasets that were somewhat "linked" to a selected site.
- **Class 1**: positive selection on a *de novo* mutation (i.e. hard sweep). The selection coefficient s was chosen uniformly from (0.05, 0.1), and the selection start time was chosen uniformly from (0.001, 0.01), in coalescent units.
- **Class 2**: balancing selection. The selection coefficient and selection start time were chosen in the same fashion as Class 1.
- **Class 3**: positive selection on standing variation (i.e. soft sweep). The selection coefficient and selection start time were chosen as before, but now the frequency of the selected allele was uniformly chosen from $(5 \times 10^{-5}, 5 \times 10^{-4})$. This follows Garud et al. [29], who found that there was low power to detect selection from standing variation with initial frequencies beyond 10^{-3}.

Using this strategy, we simulated 1000 different demographic histories, with 150 regions for each one, for a total of 150,000 datasets. To build 150 regions for each demography, we simulated 30 datasets for each of the classes 1-3, and 60 neutral datasets. We masked some of the bases for the simulated data to make it as similar as possible to the real dataset.

4.4.2 Transforming input data into summary statistics

Unlike the image data example, unfortunately we cannot plug the raw genomic data into a deep learning method. Like ABC, we need to transform the data into summary statistics that are potentially informative about the parameters of interest. Unlike ABC however, deep learning should not be negatively affected by correlated or uninformative summary statistics. So we sought to include any and all summary statistics of the data. The statistics we used are shown below, and the regions described afterwards:

1. Unfolded site frequency spectrum (SFS) for the sites where we had ancestral allele information [10] and known allele information at at least 18/22 individuals (across the entire region): 17 statistics.

2. Folded SFS for the sites where we did not have ancestral allele information, but still known allele information at 18/22 individuals (for each of three regions): $9 \cdot 3 = 27$ statistics.

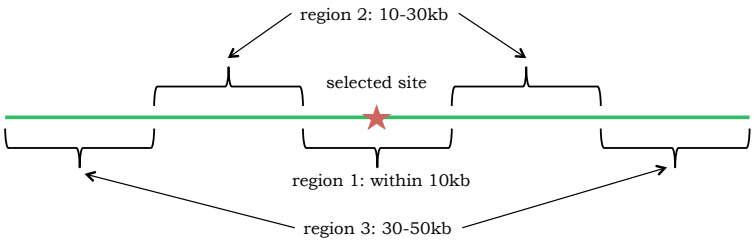


Figure 4.8: Regions used for computing the statistics, which are based off of [93].

---

3. Length distribution between segregating sites (for each of three regions): $16 \cdot 3 = 48$ statistics.

4. Identity-by-state (IBS) tract length distribution (for each of three regions): $25 \cdot 3 = 75$ statistics.

5. Linkage disequilibrium distribution between sites near the selected sites and region sites (for each of three regions): $16 \cdot 3 = 48$ statistics.

6. H1, H12, and H2 statistics, as described in [29]. These statistics help to distinguish between hard and soft sweeps: 3 statistics.

This gives us a total of 218 statistics. For statistic types 2-5, the regions we used were: close to the selected site (within 10kb on either side), mid-range from the selected site (within 10kb - 30kb on either side), and far from the selected site (from 30kb to 50kb on either side). These regions are based off of the simulation scenario in [93], and shown more explicitly in Figure 4.8.

4.4.3 Deep learning model

To modify our deep learning method to accommodate this type of inference problem, during training we have an outer-loop that changes the demography as necessary, and an inner loop that accounts for differences in selection for each region. During testing, we estimate demography and selection for each region separately, then average the demographies to get one global estimate for each genome. One final complication is that we estimate continuous parameters for the population sizes, but consider selection to be a discrete parameter. This involves a linear activation function for the population sizes and a softmax classifier for the selection parameter. A diagram of our deep learning method is shown in Figure 4.9.

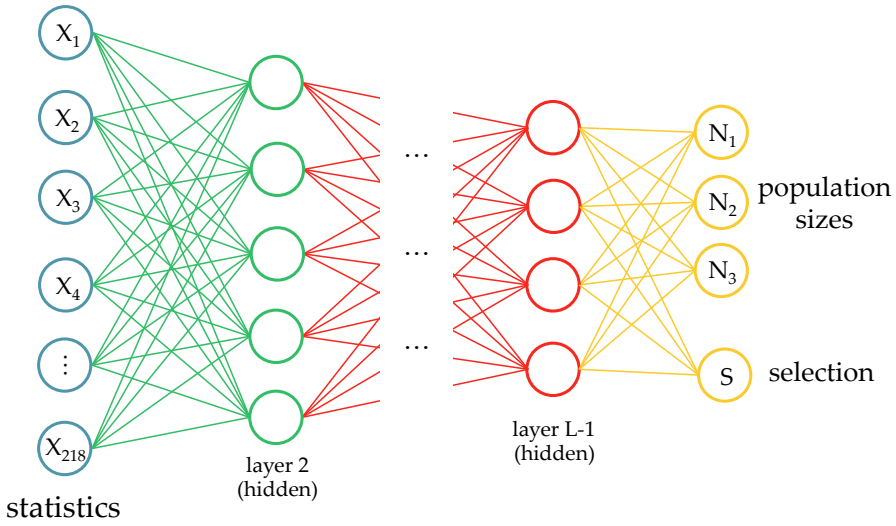


Figure 4.9: Our deep learning framework for effective population size changes and selection.

4.5 Results

4.5.1 Results for demography only

One of the first things we tested was the impact of unsupervised pre-training using autoencoders. Figure 4.10 shows that the pre-training is very effective. Initializing the weights of the entire network randomly, then trying to finding a global minimum for the cost function is ineffective. In this scenario we simulated data with variable demography only, no selection.

We also tested the impact on ABC of adding more summary statistics. For this we used ABCtoolbox [115], using the same numbers of training and testing datasets as we used for deep learning. Although ideally adding more summary statistics should reduce the estimation error, adding summary statistics to ABC often increases the error. This is typically because uninformative summary statistics can bias the distance metric (i.e. increase the distance) between two datasets with similar parameters. In Figure 4.11, we can see that the absolute squared error increases for population sizes 1 and 3 when we add more statistics.

Then we compared deep learning to ABC, using 5,000 datasets, with 90% for training and 10% for testing. The results are shown in Figure 4.12, and we can see that deep learning significantly outperforms ABCtoolbox in this scenario.

We also wanted to investigate which statistics were the most informative. There could be many ways of interpreting informative statistics from the weights learned by the deep learning training procedure. We first compute the sum of all the weights on all the paths between the summary statistics and the parameters. Then we select the statistics that correspond to the paths with the highest weights. In Figure 4.13, we plot the values of the statistics for a particular dataset, then highlight in green the statistics that were determined

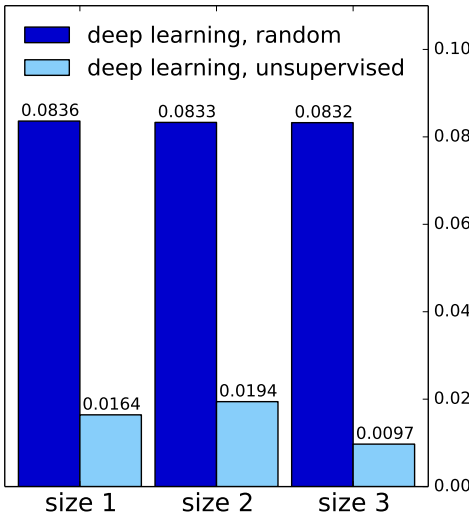


Figure 4.10: Absolute squared error (y-axis) on the test dataset, for a deep network with 6 hidden layers. The impact of unsupervised pre-training is clear. Initializing all the weights of the full network randomly produces the dark blue errors, whereas initializing the weights using an autoencoder produces the light blue errors. This scenario is for demography only.

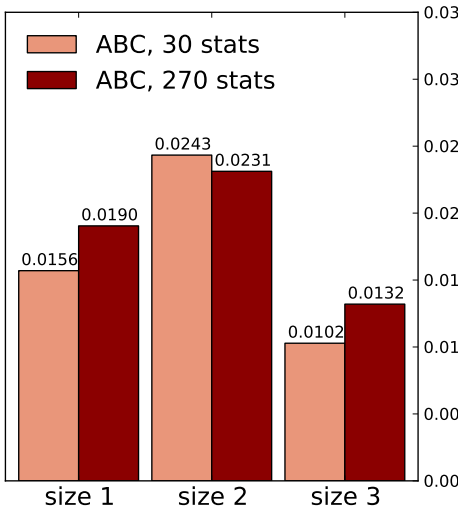


Figure 4.11: Impact on the ABC absolute squared error (y-axis) of adding more summary statistics. The 30 statistics were chosen to be the statistics with the highest weights from a deep learning framework (so by some measure they should be quite informative for the population sizes of interest). If we add many more less informative statistics, the absolute squared error increases for population sizes 1 and 3. Ideally adding more statistics should decrease the error, even if the new statistics are weakly informative or even uninformative. This scenario is for demography only.

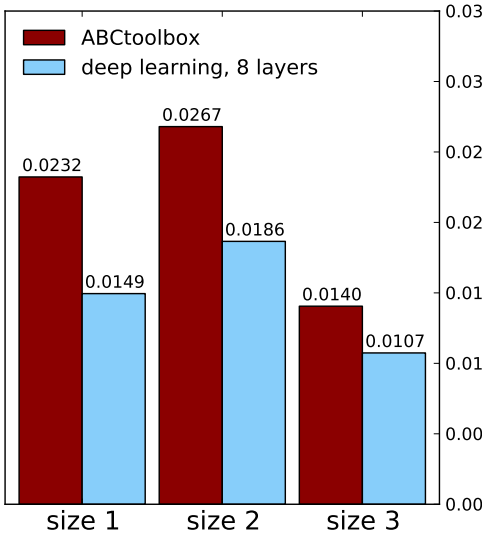


Figure 4.12: A comparison between ABCtoolbox and deep learning for a demography only scenario. With 8 hidden layers and 5,000 simulated datasets (90% training data and 10% test data), deep learning outperforms ABCtoolbox, as measured by the absolute squared error (y-axis).

---

to be most informative using this procedure.

One hyper-parameter that should be investigated more closely is λ, the weight decay parameter. If λ is set to be too high, large weights will be penalized too much, and interesting features of the data cannot be learned well. But if λ is set too low, the weights tend to display runaway behavior. Due to this balance, a validation procedure is typically used to find the right λ. In our case, the additional runtime of simulating more data and performing more training would be too computationally expensive, but we do provide a small validation study in Figure 4.14.

4.5.2 Results for demography and selection

Moving to the demography and selection scenario, we used the dataset and summary statistics described in Sections 4.4.1 and 4.4.2, with 150,000 datasets. We used 700 demographies for training and 300 for testing. In Figure 4.15, we show the result for a network with 3 hidden layers of sizes 25,25,10. Encouragingly, the best performance was found when we collected the regions classifies as neutral (not the true neutral regions), and then used these for demographic inference.

To analyze the selection results, we calculated a confusion matrix in Table 4.2 to show which datasets of each class were classified correctly, or classified as belonging to a different class. Our most frequent errors were classifying datasets under selection as neutral (first column of the matrix). This could be because the selection occurred anciently and quickly,

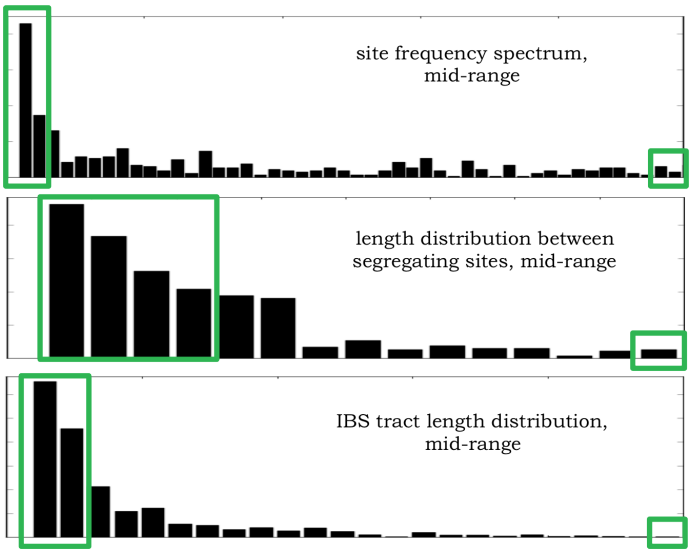


Figure 4.13: One of the advantages of deep learning is that the learned weights of the network can be used to find the most informative summary statistics for the parameters of interest. Here is a bar chart of the values of the summary statistics for a particular dataset, with the most informative statistics highlighted in green. The tails of the distributions generally seem to be the most informative. This scenario is for demography only.

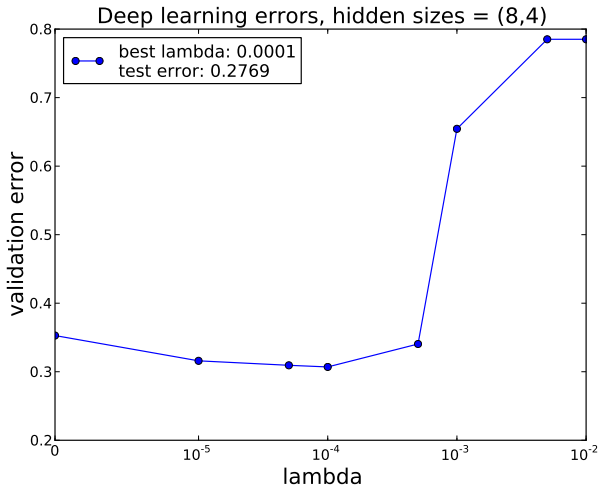


Figure 4.14: Validation procedure for a network with two hidden layers of size 8 and 4. The x-axis shows increasing values of λ, and the y-axis shows the error on the validation dataset. The curve shows a characteristic shape with low and high λ producing poorer results than an intermediate value. For these hidden layers sizes and this dataset, $\hat{\lambda} = 0.0001$ was optimal.

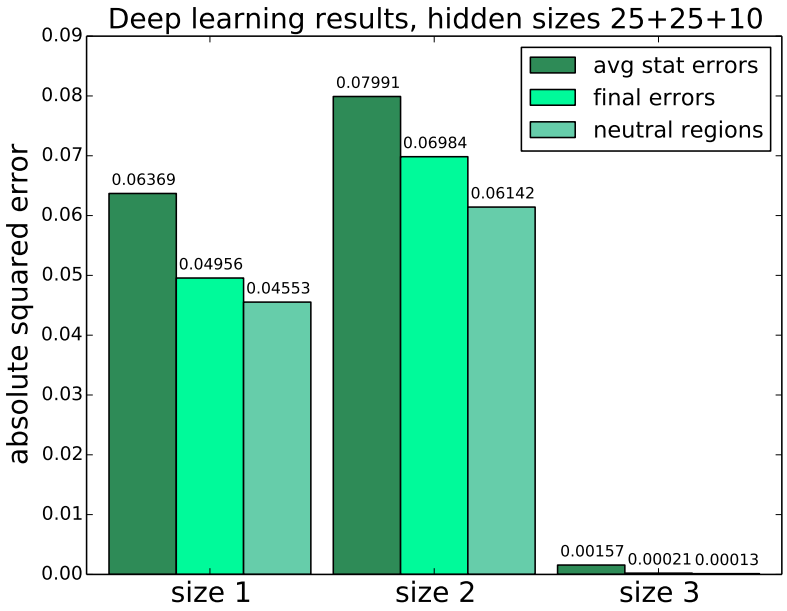


Figure 4.15: Here are population size results for the scenario with demography and selection. We evaluate the results in three ways. First, for each test demography, we average all the statistics together, and then run these values through the training network. Second, for each test demography, we run the datasets through the network one by one, then average the predictions. Finally, we perform the second procedure, but with only the regions we classified as neutral. This final method achieves the optimal performance. We note that the most ancient size (N_3) is always the most accurately estimated.

---

so at the present time the selected site looks like any other fixed site in the genome. This type of false negative error is preferable to finding many regions under selection that are truly neutral.

We then ran the real *Drosophila* data through this trained network just like any other test dataset. For the demography, the bottleneck results are shown in Table 4.3. The last row (neutral regions) is our best estimate. We can see that using all the regions tends to downwardly bias the sizes, which makes sense given that selection often reduces diversity.

Finally, our estimates for the number of regions under different types of selection are shown in Table 4.4.

In terms of runtime, the vast majority is spent simulating the data. During training, most of the runtime is spend fine-tuning the deep network, which requires computing the cost function and derivatives many times. To speed up this computation, our deep learning implementation is parallelized across datasets, since each dataset adds to the cost function independently. This significantly improved the training time for deep learning, which can be run overnight on this dataset with a modest number of hidden layers. Once the training is completed, an arbitrary number of datasets can be tested more or less instantaneously. In

	fraction predicted for each class			
neutral	**0.9399**	0.0122	0.0066	0.0412
hard sweep	0.1863	**0.7391**	0.0184	0.0561
balancing selection	0.1170	0.0166	**0.8268**	0.0394
soft sweep	0.1232	0.0926	0.0306	**0.7534**

Table 4.2: Confusion matrix for the selection predictions, in the demography and selection scenario. Each row represents the datasets that truly belong to each selection class. Each column represents the datasets that were actually classified as each selection class. So ideally we would like all 1's down the diagonal, and 0's in the off-diagonal entries. We can see that neutral datasets are the easiest to classify, and often regions under selection look neutral as well (first column). The overall percentage of misclassified datasets was 16%.

prediction	N_1 (recent)	N_2 (bottleneck)	N_3 (ancestral)
Duchen et al. [21]	5,224,100	620	4,975,360
Average statistics	524,587	40,616	243,761
Average predictions	803,505	46,795	240,955
Neutral regions	836,781	48,144	261,052

Table 4.3: Population size results for African *Drosophila*. The last row (predictions based on the regions we classified as neutral) represents our best estimate of the population sizes.

selection class	neutral	hard sweep	balancing selection	soft sweep
number of regions	3236	34	544	603

Table 4.4: Selection results for African *Drosophila*. We find many regions under balancing selection and selection from standing variation. We find relatively few hard sweeps.

contrast, each of the "training" datasets for ABC must be examined for *each* test dataset. This takes several days for a dataset of this size, although it could be parallelized across the test datasets.

4.6 Discussion

Using deep learning for population genetics is still in its infancy, and there are many directions of future work. Deep learning provides a way to distinguish informative summary statistics from informative ones. Exactly how to define informative is an open question, but learning more about how statistics relate to parameters could be very useful for population genetics going forward.

The prospect of using deep learning to classify regions as neutral or selected is very appealing for subsequent demographic inference. There are other machine learning methods that perform such classification, but they are generally limited to two classes (selected or neutral). Such methods are often based on SVMs, which have been shown to be less robust than deep learning.

We would also like to apply deep learning to a wider variety of scenarios in populations genetics. Population structure and splits would be an example, although this would most likely require a different set of summary statistics. We could use statistics based on PSMC, or rare variant statistics such as f2, as described in [73].

Finally, machine learning methods have been criticized for their "black-box" nature. In some sense they throw away a lot of the coalescent modeling that we know to be realistic, although this is included somewhat in the expert summary statistics of the data. It would be advantageous to somehow combine the strengths of coalescent theory and the strengths of machine learning to create a robust method for population genetic inference. At the same time, it is appealing from a computer science perspective to have the method learn everything by itself. It should be possible to plug raw genetic data into a machine learning method like is usually done for image data. It would be fascinating to see if a machine learning method could automatically learn informative summary statistics such as the site frequency spectrum. These somewhat orthogonal deep learning goals could potentially both yield interesting theoretical and practical results.

Chapter 5

Conclusions

In this thesis, we have described two new methods for performing population genetic inference on modern sequencing data. The first algorithm, diCal, estimates a piecewise-constant effective population size history for a single population. The main contribution of diCal over previous methods is its ability to handle an arbitrary number of genomes in a coalescent HMM framework. However, even though diCal is theoretically scalable, the original version is still computationally intensive, both in terms of time and memory. diCal is parallelized, which can provide a significant decrease in the runtime, but increases the memory requirement.

To improve the computational complexity of diCal, we also discuss an algorithm that decreases the runtime from quadratic to linear in the number of time discretization intervals. This algorithm is provided in the context of diCal, but is applicable to coalescent HMMs more generally. It would be interesting to see this runtime improvement implemented in other methods such as PSMC, CoalHMM, or MSMC. In addition, we provide a binning algorithm that groups adjacent sites, which also improves diCal's runtime. Unfortunately the binning algorithm seems to negatively impact diCal's performance in the recent past, but future work on binning with multiple sequences will hopefully solve this issue. Further, we describe an algorithm for using diCal to estimate local trees along the genome, which can then be used to estimate the number of remaining lineages as a function of time. This function can be used as an alternative method for estimating population sizes, or perhaps other types of demographic events.

The choice of time discretization for diCal (or any other coalescent HMM) is delicate. In this thesis we discuss an adaptive procedure for choosing the best discretization possible within some constraints. A good discretization will be fine (as opposed to coarse) during times of interesting demographic changes, but not so fine that a lack of coalescent events causes runaway behavior. Improving and automating discretization choice is an interesting and promising area of future work. Right now it is difficult to see how coalescent HMMs could move away from a discrete state space (and thus often a piecewise-constant population size history), but a continuous-time hidden state could potentially alleviate many of the issues that arise with a discrete-time model. Additionally, as sample sizes increase, discrete-time

models like the Wright-Fisher model could be used in a more explicit fashion for population size inference.

The runtime of a coalescent HMM generally depends on three parameters: the sample size n, the sequence length L, and the number of time discretization intervals d. When many genomes are available, it requires very little user time to vary these parameters, and often yields unexpected results. An important avenue of future work will be tuning n, L, and d to improve inference in humans and other organisms.

The second algorithm presented in this thesis is a tailored deep learning method for jointly inferring effective population size changes and natural selection. There has been little previous work in this area, as likelihood-based methods would be extremely computationally challenging for this problem. Machine learning methods have proven very effective at learning arbitrary functions from input data to parameters of interest. One challenge when applying deep learning to genomic data is that the raw data cannot be directly plugged into the methods. Here we provide one strategy for decomposing raw sequence data into summary statistics that can then be used for deep learning. A strength of our method is that it can handle hundreds of summary statistics, even those that are strongly correlated or weakly informative.

We apply this deep learning method to African *Drosophila melanogaster* data to infer a bottleneck history while jointly classifying regions of the genome into different selection classes. Each region can either be classified as neutral, a hard selective sweep, a soft selective sweep, or balancing selection. Using continuous parameters combined with discrete parameters is very unusual for either population genetics or deep learning, but provides us with flexibility in our inference strategy.

Using machine learning for population genetics is a young area, with many directions to explore. One interesting line of research will hopefully combine the biologically accurate modeling of coalescent theory with machine learning methods. In a completely different direction, it would also be interesting to explore using raw sequence data directly for machine learning methods, bypassing the need for summary statistics. It is unclear which type of method would achieve superior performance, but both directions will ideally help us better understand how demographic events shape genetic data.

Bibliography

[1] Simon Aeschbacher, Mark A. Beaumont, and Andreas Futschik. "A novel approach for choosing summary statistics in approximate Bayesian computation". In: *Genetics* 192.3 (2012), pp. 1027–1047.

[2] Philip Awadalla et al. "Direct measure of the de novo mutation rate in autism and schizophrenia cohorts". In: *American Journal of Human Genetics* 87.3 (2010), pp. 316–324.

[3] Francisco J. Ayala. "The myth of Eve: molecular biology and human origins". In: *Science* 270.5244 (1995), pp. 1930–1936.

[4] Mark A. Beaumont, Wenyang Zhang, and David J. Balding. "Approximate Bayesian Computation in Population Genetics". In: *Genetics* 162.4 (2002), pp. 2025–2035.

[5] Alan O. Bergland et al. "Genomic evidence of rapid and stable adaptive Oscillations over seasonal time scales in Drosophila". In: *PLoS Genetics* 11.10 (2014), e1004775.

[6] Anand Bhaskar and Yun S. Song. "Descartes' rule of signs and the identifiability of population demographic models from genomic variation data". In: *Annals of Statistics* 42.6 (2014), pp. 2469–2493.

[7] Michael G.B. Blum and Olivier François. "Non-linear regression models for Approximate Bayesian Computation". In: *Statistics and Computing* 20.1 (2010), pp. 63–73.

[8] Michael G.B. Blum et al. "A comparative review of dimension reduction methods in approximate Bayesian computation". In: *Statistical Science* 28.2 (2013), pp. 189–208.

[9] James A. Cahill et al. "Genomic evidence for island population conversion resolves conflicting theories of polar bear evolution". In: *PLoS Genetics* 9.3 (2013), e1003345.

[10] Andrew H. Chan, Paul A. Jenkins, and Yun S. Song. "Genome-wide fine-scale recombination rate variation in *Drosophila melanogaster*". In: *PLoS Genetics* 8.12 (2012), e1003090.

[11] Brian Charlesworth. "Effective population size and patterns of molecular evolution and variation". In: *Nature Reviews Genetics* 10 (2009), pp. 195–205.

[12] Dan Claudiu Cireşan, Ueli Meier, and Jürgen Schmidhuber. "Multi-column deep neural networks for image classification". In: *IEEE Conference on Computer Vision and Pattern Recognition* (2012), pp. 3642–3649.

[13] Dan Claudiu Cireşan et al. "Deep Big Simple Neural Nets Excel on Handwritten Digit Recognition". In: *Neural Computation* 22.12 (2010).

[14] The 1000 Genomes Project Consortium. "A map of human genome variation from population-scale sequencing". In: *Nature* 467 (2010), pp. 1061–1073.

[15] Alex Coventry et al. "Deep resequencing reveals excess rare recent variants consistent with explosive population growth". In: *Nature Communications* 1.131 (2010), pp. 1–6.

[16] Dana C. Crawford et al. "Evidence for substantial fine-scale variation in recombination rates across the human genome". In: *Nature Genetics* 36 (2004), pp. 700–706.

[17] George Cybenko. "Approximations by superpositions of sigmoidal functions". In: *Mathematics of Control, Signals, and Systems* 2.4 (1989), pp. 303–314.

[18] George E. Dahl, Tara N. Sainath, and Geoffrey E. Hinton. "Improving deep neural networks for LVCSR using rectified linear units and dropout". In: *IEEE International Conference on Acoustic Speech and Signal Processing (ICASSP)* (2013), pp. 8609–8613.

[19] Dennis DeCoste and Bernhard Schölkopf. "Training Invariant Support Vector Machines". In: *Machine Learning* 46 (2002), pp. 161–190.

[20] Robert Dodier. *LBFGS optimization routine, Java translation.* `http://riso.sourceforge.net/`. 1999.

[21] Pablo Duchen et al. "Demographic inference reveals African and European admixture in the North American *Drosophila melanogaster* population". In: *Genetics* 193.1 (2013), pp. 291–301.

[22] Julien Y. Dutheil et al. "Ancestral population genomics: the coalescent hidden Markov model approach". In: *Genetics* 183.1 (2009), pp. 259–274.

[23] Eyal Elyashiv et al. "A genomic map of the effects of linked selection in Drosophila". In: (2014). `http://arxiv.org/abs/1408.5461`.

[24] Henry A. Erlich et al. "HLA Sequence Polymorphism and the Origin of Humans". In: *Science* 274 (1996), pp. 1552–1554.

[25] Gregory Ewing and Joachim Hermisson. "MSMS: A Coalescent Simulation Program Including Recombination, Demographic Structure, and Selection at a Single Locus". In: *Bioinformatics* 26 (16 2010), pp. 2064–2065.

[26] Paul Fearnhead and Dennis Prangle. "Constructing summary statistics for approximate Bayesian computation: semi-automatic ABC". In: (2011). `http://arxiv.org/abs/1004.1112`.

[27] Paul Fearnhead and Nick G. C. Smith. "A novel method with improved power to detect recombination hotspots from polymorphism data reveals multiple hotspots in human genes". In: *American Journal of Human Genetics* 77.5 (2005), pp. 781–794.

[28] Nicolas Galtier, Frantz Depaulis, and Nicholas H. Barton. "Detecting bottlenecks and selective sweeps from DNA sequence polymorphism". In: *Genetics* 155 (2000), pp. 981–987.

[29] Nandita R. Garud et al. "Recent selective sweeps in North American *Drosophila melanogaster* show signatures of soft sweeps". In: *PLoS Genetics* 11.2 (2015), e1005004.

[30] John H. Gillespie. *Population Genetics: A Concise Guide*. Second. Johns Hopkins University Press, 2004.

[31] Toni I. Gossmann, Megan Woolfit, and Adam Eyre-Walker. "Quantifying the variation in the effective population size within a genome". In: *Genetics* 189 (2011), pp. 1389–1402.

[32] Simon Gravel et al. "Demographic history and rare allele sharing among human populations". In: *PNAS* 108.29 (2011), pp. 11983–11988.

[33] Alex Graves, Abdel-rahman Mohamed, and Geoffrey Hinton. "Speech recognition with deep recurrent neural networks". In: *IEEE International Conference on Acoustic Speech and Signal Processing (ICASSP)* (2013), pp. 6645–6649.

[34] Robert C. Griffiths, Paul A. Jenkins, and Yun S. Song. "Importance sampling and the two-locus model with subdivided population structure". In: *Advances in Applied Probability* 40.2 (2008), pp. 473–500.

[35] Robert C. Griffiths and Simon Tavaré. "Sampling theory for neutral alleles in a varying environment". In: *Philosophical Transactions of the Royal Society London B: Biological Sciences* 344.1310 (1994), pp. 403–410.

[36] Martien A.M. Groenen et al. "Analyses of pig genomes provide insight into porcine demography and evolution". In: *Nature* 491 (2012), pp. 393–398.

[37] Ilan Gronau et al. "Bayesian inference of ancient human demography from individual genome sequences". In: *Nature Genetics* 43 (2011), pp. 1031–1034.

[38] Ryan N. Gutenkunst et al. "Inferring the joint demographic history of multiple populations from multidimensional SNP frequency data". In: *PLoS Genetics* 5.10 (2009), e1000695.

[39] Cathy Haag-Liautard et al. "Direct estimation of per nucleotide and genomic deleterious mutation rates in *Drosophila*". In: *Nature* 445 (2007), pp. 82–85.

[40] Penelope R. Haddrill et al. "Multi-locus patterns of nucleotide variability and the demographic and selection history of *Drosophila melanogaster* populations". In: *Genome Research* 15 (2005), pp. 790–799.

[41] Frank Hailer et al. "Nuclear genomic sequences reveal that polar bears are an old and distinct bear lineage". In: *Science* 336.6079 (2012), pp. 344–347.

[42] Michael F. Hammer. "A recent common ancestry for human Y chromosomes". In: *Nature* 378 (1995), pp. 376–378.

[43] Kelley Harris and Rasmus Nielsen. "Inferring demographic history from a spectrum of shared haplotype lengths". In: *PLoS Genetics* 9.6 (2013), e1003521.

[44] Kelley Harris et al. "Decoding coalescent hidden Markov models in linear time". In: *Proc. 18th Annual Intl. Conf. on Research in Computational Molecular Biology (RECOMB)* 8394 (2014), pp. 100–114.

[45] Geoffrey E. Hinton and Ruslan R. Salakhutdinov. "Reducing the dimensionality of data with neural networks". In: *Science* 313.5786 (2006), pp. 504–507.

[46] Geoffrey E. Hinton et al. "Improving neural networks by preventing co-adaptation of feature detectors". In: (2012). `http://arxiv.org/abs/1207.0580`.

[47] Asger Hobolth et al. "Genomic relationships and speciation times of human, chimpanzee, and gorilla Inferred from a coalescent hidden Markov model". In: *PLoS Genetics* 3.2 (2007), e7.

[48] John J. Hopfield. "Neural networks and physical systems with emergent collective computational abilities". In: *PNAS* 79.8 (1982), pp. 2554–2558.

[49] Kurt Hornik. "Approximation Capabilities of Multilayer Feedforward Networks". In: *Neural Networks* 4.2 (1991), pp. 251–257.

[50] Richard R. Hudson. "Generating samples under a Wright-Fisher neutral model". In: *Bioinformatics* 18.2 (2002), pp. 337–338.

[51] Richard R. Hudson. "Properties of the neutral allele model with intergenic recombination". In: *Theoretical Population Biology* 23.2 (1983), pp. 183–201.

[52] Maria De Iorio and Robert C. Griffiths. "Importance sampling on coalescent histories. I". In: *Advances in Applied Probability* 36.2 (2004), pp. 417–433.

[53] Maria De Iorio and Robert C. Griffiths. "Importance sampling on coalescent histories. II: Subdivided population models". In: *Advances in Applied Probability* 36.2 (2004), pp. 434–454.

[54] Nicola Jones. "The Learning Machines". In: *Nature* 505 (2014), pp. 146–148.

[55] Paul Joyce and Paul Marjoram. "Approximately sufficient statistics and Bayesian computation". In: *Statistical Applications in Genetics and Molecular Biology* 7.1 (2008), p. 26.

[56] Balázs Kégl and Róbert Busa-Fekete. "Boosting products of base classifiers". In: *Proceedings of the 26th Annual International Conference on Machine Learning* (2009), pp. 497–504.

[57] Alon Keinan and Andrew G. Clark. "Recent explosive human population growth has resulted in an excess of rare genetic variants". In: *Science* 36.6082 (2012), pp. 740–743.

[58] Daniel Keysers et al. "Deformation models for image recognition". In: *EEE Transactions on Pattern Analysis and Machine Intelligence* 29.8 (2007), pp. 1422–1435.

[59] J.F.C. Kingman. "On the Genealogy of Large Populations". In: *Journal of Applied Probability* 19A (1982), pp. 27–43.

[60] Augustine Kong et al. "Rate of *de novo* mutations and the importance of father's age to disease risk". In: *Nature* 488 (2012), pp. 471–475.

[61] Alex Krizhevsky, Ilya Sutskever, and Geoffrey E. Hinton. "ImageNet classification with deep convolutional neural networks". In: *Advances in Neural Information Processing Systems* (2012), pp. 1097–1105.

[62] Yann LeCun et al. "Gradient-Based Learning Applied to Document Recognition". In: *Proceedings of the IEEE* 86.11 (1998), pp. 2278–2324.

[63] Michael K. K. Leung et al. "Deep learning of the tissue-regulated splicing code". In: *Bioinformatics* 30.12 (2014), pp. i121–i129.

[64] Heng Li and Rich Durbin. "Inference of human population history from individual whole-genome sequences". In: *Nature* 475 (2011), pp. 493–496.

[65] Junrui Li et al. "Joint analysis of demography and selection in population genetics: where do we stand and where could we go?" In: *Molecular Ecology* 21.1 (2012), pp. 28–44.

[66] Na Li and Matthew Stephens. "Modelling Linkage Disequilibrium, and identifying recombination hotspots using single-nucleotide polymorphism data". In: *Genetics* 165.4 (2003), pp. 2213–2233.

[67] Kao Lin, Andreas Futschik, and Haipeng Li. "A fast estimate for the population recombination rate based on regression". In: *Genetics* 194 (2013), pp. 473–484.

[68] Kao Lin et al. "Distinguishing positive selection from neutral evolution: boosting the performance of summary statistics". In: *Genetics* 187 (2011), pp. 229–244.

[69] Shiping Liu et al. "Population Genomics Reveal Recent Speciation and Rapid Evolutionary Adaptation in Polar Bears". In: 157 (4 2014), pp. 785–794.

[70] Thomas Mailund et al. "A new isolation with migration model along complete genomes infers very different divergence processes among closely related great ape species". In: *PLoS Genetics* 8.12 (2012), e1003125.

[71] Thomas Mailund et al. "Estimating divergence time and ancestral effective population size of Bornean and Sumatran orangutan subspecies using a coalescent hidden Markov model". In: *PLoS Genetics* 7.3 (2011), e1001319.

[72] Paul Marjoram and Jeff D. Wall. "Fast "coalescent" simulation". In: *BMC Genetics* 7 (2006), p. 16.

[73] Iain Mathieson and Gil McVean. "Demography and the age of rare variants". In: *PLoS Genetics* 10.8 (2014), e1004528.

[74] Gilean A.T. McVean and Niall J. Cardin. "Approximating the coalescent with recombination". In: *Philosophical Transactions of the Royal Society London B: Biological Sciences* 360.1459 (2005), pp. 1387–93.

[75] Matthias Meyer et al. "A high-coverage genome sequence from an archaic Denisovan individual". In: *Science* 338.6104 (2012), pp. 222–226.

[76] Webb Miller et al. "Polar and brown bear genomes reveal ancient admixture and demographic footprints of past climate change". In: *PNAS* 109.36 (2012), pp. 2382–2390.

[77] *MNIST database*. `http://en.wikipedia.org/wiki/MNIST_database`. Accessed: 2015-04-07.

[78] Simon Myers, Charles Fefferman, and Nick Patterson. "Can one learn history from the allelic spectrum?" In: *Theoretical Population Biology* 73.3 (2008), pp. 342–348.

[79] Masatoshi Nei and Dan Graur. "Extent of protein polymosphism and the neutral mutation theory". In: *Evolutionary Biology* 17 (1984), pp. 73–118.

[80] Masatoshi Nei and A. K. Roychoudhury. "Genic Variation Within and Between the Three Major Races of Man, Caucasoids, Negroids, and Mongoloids". In: *American Journal of Human Genetics* 26 (1974), pp. 421–443.

[81] John A. Nelder and Roger Mead. "A simplex method for function minimization". In: *The Computer Journal* 7.4 (1965), pp. 308–313.

[82] Matthew R Nelson et al. "An abundance of rare functional variants in 202 drug target genes sequenced in 14,002 people". In: *Science* 337.6090 (2012), pp. 100–104.

[83] Andrew Ng et al. *Unsupervised Feature Learning and Deep Learning Tutorial*. `http://ufldl.stanford.edu/wiki/index.php/UFLDL_Tutorial`. 2013.

[84] John Novembre et al. "Genes mirror geography within Europe". In: *Nature* 456 (2008), pp. 98–101.

[85] Matthew A. Nunes and David J. Balding. "On optimal selection of summary statistics for approximate Bayesian computation". In: *Statistical Applications in Genetics and Molecular Biology* 9.1 (2010), p. 34.

[86] Ludovic Orlando et al. "Recalibrating *Equus* evolution using the genome sequence of an early Middle Pleistocene horse". In: *Nature* 499 (2013), pp. 74–78.

[87] Pier Francesco Palamara et al. "Length distributions of identity by descent reveal fine-scale demographic history". In: *American Journal of Human Genetics* 91.5 (2012), pp. 809–822.

[88] Joshua S. Paul and Yun S. Song. "A principled approach to deriving approximate conditional sampling distributions in population genetics models with recombination". In: *Genetics* 186.1 (2010), pp. 321–338.

[89] Joshua S. Paul and Yun S. Song. "Blockwise HMM computation for large-scale population genomic inference". In: *Bioinformatics* 28.15 (2012), pp. 2008–2015.

[90] Joshua S. Paul, Matthias Steinrücken, and Yun S. Song. "An accurate sequentially Markov conditional sampling distribution for the coalescent with recombination". In: *Genetics* 187.4 (2011), pp. 1115–1128.

[91] Pavlos Pavlidis, Jeffrey D. Jensen, and Wolfgang Stephan. "Searching for footprints of positive selection in whole-genome SNP data from nonequilibrium populations". In: *Genetics* 185.3 (2010), pp. 907–922.

[92] Pavlos Pavlidis et al. "SweeD: likelihood-based detection of selective sweeps in thousands of genomes". In: *Molecular Biology and Evolution* (2013). doi:10.1093/molbev/mst112.

[93] Benjamin M. Peter, Emilia Huerta-Sanchez, and Rasmus Nielsen. "Distinguishing between selective sweeps from standing variation and from a *d*e novo mutation". In: *PLoS Genetics* 8.10 (2012), e1003011. DOI: 10.1371/journal.pgen.1003011.

[94] John E. Pool et al. "Population genomics of Sub-Saharan *D*rosophila melanogaster: African diversity and non-African admixture". In: *PLoS Genetics* 8.12 (2012), e1003080.

[95] Jonathan K. Pritchard. "Whole-genome sequencing data offer insights into human demography". In: *Nature Genetics* 43.10 (2011), pp. 923–925.

[96] Jonathan K. Pritchard et al. "Population growth of human Y chromosomes: a study of Y chromosome microsatellites". In: *Molecular Biology and Evolution* 16.12 (1999), pp. 1791–1798.

[97] Bruce Rannala and Ziheng Yang. "Bayes estimation of species divergence times and ancestral population sizes using DNA sequences from multiple loci". In: *Genetics* 164 (2003), pp. 1645–1656.

[98] Matthew D. Rasmussen et al. "Genome-wide inference of ancestral recombination graphs". In: *PLoS Genetics* 10.5 (2014), e1004342.

[99] Jared C. Roach et al. "Analysis of genetic inheritance in a family quartet by whole-genome sequencing". In: *Science* 328.5978 (2010), pp. 636–639.

[100] Roy Ronen et al. "Learning natural selection from the site frequency spectrum". In: *Genetics* 195 (2013), pp. 181–193.

[101] Aylwyn Scally and Richard Durbin. "Revising the human mutation rate: implications for understanding human evolution". In: *Nature Reviews Genetics* 10 (2012), pp. 745–753.

[102] Stephan Schiffels and Richard Durbin. "Inferring human population size and separation history from multiple genome sequences". In: *Nature Genetics* 46 (2014), pp. 919–925.

[103] Sara Sheehan, Kelley Harris, and Yun S. Song. "Estimating variable effective population sizes from multiple genomes: A sequentially Markov conditional sampling distribution approach". In: *Genetics* 194.3 (2013), pp. 647–662.

[104] Stephen T. Sherry et al. "Alu Evolution in Human Populations: Using the Coalescent to Estimate Effective Population Size". In: *Genetics* 147 (1997), pp. 1977–1982.

[105] Matthias Steinrücken, Joshua S. Paul, and Yun S. Song. "A sequentially Markov conditional sampling distribution for structured populations with migration and recombination". In: *Theoretical Population Biology* 87 (2013), pp. 51–61.

[106] Naoyuki Takahata. "Allelic genealogy and human evolution". In: *Molecular Biology and Evolution* 10.1 (1993), pp. 2–22.

[107] Paula Tataru, Jasmine A. Nirody, and Yun S. Song. "diCal-IBD: demography-aware inference of identity-by-descent tracts in unrelated individuals". In: *Bioinformatics* 30.23 (2014), pp. 3430–3431.

[108] Simon Tavaré. "Lines of descent and genealogical processes, and their application in population genetics models". In: *Theoretical Population Biology* 26.2 (1984), pp. 119–164.

[109] Simon Tavaré et al. "Inferring Coalescence Times From DNA Sequence Data". In: *Genetics* 145.2 (1997), pp. 505–518.

[110] Albert Tenesa et al. "Recent human effective population size estimated from linkage disequilibrium". In: *Genome Research* 17.4 (2007), pp. 520–526.

[111] Kevin Thornton and Peter Andolfatto. "Approximate Bayesian inference reveals evidence for a recent, severe bottleneck in a Netherlands population of *Drosophila melanogaster*". In: *Genetics* 172.3 (2006), pp. 1607–1619.

[112] John Wakeley. *Coalescent Theory: An Introduction*. Greenwood Village, Colorado: Roberts & Company Publishers, 2008.

[113] Qiu-Hong Wan et al. "Genome analysis and signature discovery for diving and sensory properties of the endangered Chinese alligator". In: *Cell Research* 23.9 (2013), pp. 1091–1105.

[114] Yong Wang and Jody Hey. "Estimating divergence parameters with small samples from a large number of loci". In: *Genetics* 184.2 (2010), pp. 363–379.

[115] Daniel Wegmann et al. "ABCtoolbox: a versatile toolkit for approximate Bayesian computations". In: *BMC Bioinformatics* 11 (2010), p. 116.

[116] Carsten Wiuf and Jotun Hein. "Recombination as a point process along sequences". In: *Theoretical Population Biology* 55.3 (1999), pp. 248–259.

[117] Sewall Wright. "Evolution in Mendelian populations". In: *Genetics* 16.2 (1931), pp. 97–159.

[118] Sewall Wright. "Size of population and breeding structure in relation to evolution". In: *Science* 87.2263 (1938), pp. 430–431.

[119] Hui Y. Xiong et al. "The human splicing code reveals new insights into the genetic determinants of disease". In: *Science* 347.6218 (2015).

[120] Shancen Zhao et al. "Whole-genome sequencing of giant pandas provides insights into demographic history and local adaptation". In: *Nature Genetics* 45.1 (2013), pp. 67–71.

Appendix A

diCal formulas and implementation details

A.1 HMM formulas

The expression $R(i,t;j,t')$ in Equation 2.10 is defined as

$$R(i,t;j,t') = \begin{cases} \left(R^{(i)}(t) + \sum_{k=0}^{i-1} R^{(k)}\right), & \text{if } i < j, \\ \left(R^{(j)}(t') + \sum_{k=0}^{j-1} R^{(k)}\right), & \text{if } i > j, \\ \left(R^{(i)}(t \wedge t') + \sum_{k=0}^{i-1} R^{(k)}\right), & \text{if } i = j, \end{cases}$$

where $\wedge$ denotes the min operator and, for $u \in [t_{k-1}, t_k)$,

$$\begin{aligned} R^{(k)}(u) &:= \frac{\rho_b \lambda_k}{n - \rho_b \lambda_k} \left(e^{-\rho_b u + n(u - t_{k-1})/\lambda_k} - e^{-\rho_b t_{k-1}}\right) \prod_{m=1}^{k-1} e^{n(t_m - t_{m-1})/\lambda_m}, \\ R^{(k)} &:= \frac{\rho_b \lambda_k}{n - \rho_b \lambda_k} \left(e^{-\rho_b t_k + n(t_k - t_{k-1})/\lambda_k} - e^{-\rho_b t_{k-1}}\right) \prod_{m=1}^{k-1} e^{n(t_m - t_{m-1})/\lambda_m}. \end{aligned}$$

After the state space has been discretized, we compute the transition probabilities using $y^{(i)}$ (the probability no recombination occurs), and $z^{(i,j)}$ (the probability recombination does occur):

$$\begin{aligned} y^{(i)} &= \frac{1}{\hat{\zeta}^{(\lambda)}(D_i, h)} \int_{t_{i-1}}^{t_i} \zeta^{(\lambda)}(t,h) e^{-\rho_b t} dt \\ &= \frac{1}{w^{(i)}} \frac{n}{n + \rho_b \lambda_i} \prod_{k=1}^{i-1} e^{-n(t_k - t_{k-1})/\lambda_k} \left(e^{-\rho_b t_{i-1}} - e^{-\rho_b t_i - n(t_i - t_{i-1})/\lambda_i}\right) \text{ and} \end{aligned}$$

$$\begin{aligned} z^{(i,j)} &= \frac{n}{w^{(i)} n_{h_{\ell-1}}} \int_{t_{j-1}}^{t_j} \int_{t_{i-1}}^{t_i} \int_0^{t_{\ell-1} \wedge t_\ell} \rho_b e^{-\rho_b t_r} \frac{\zeta^{(\lambda)}(t_\ell, h_\ell)}{\int_{t_r}^{\infty} \zeta^{(\lambda)}(\tau) d\tau} \zeta^{(\lambda)}(t_{\ell-1}, h_{\ell-1}) dt_r dt_{\ell-1} dt_\ell \\ &:= Z^{(i,j)} + w^{(j)} \sum_{k=1}^{i \wedge j - 1} R^{(k)}, \end{aligned}$$

where $Z^{(i,j)}$ corresponds to the case when the recombination event occurs during the time interval $D_{i \wedge j}$ (i.e. the latest it could) and $R^{(k)}$ corresponds to a recombination event in the time interval D_k. $R^{(k)}$ is defined as before, and $Z^{(i,j)}$ is

$$Z^{(i,j)} = \frac{n}{w^{(i)} n_{h_{\ell-1}}} \int_{t_{j-1}}^{t_j} \int_{t_{i-1}}^{t_i} \int_{t_{(i \wedge j)-1}}^{t_{\ell-1} \wedge t_\ell} \rho_b e^{-\rho_b t_r} \frac{\zeta^{(\lambda)}(t_\ell, h_\ell)}{\int_{t_r}^{\infty} \zeta^{(\lambda)}(\tau) d\tau} \zeta^{(\lambda)}(t_{\ell-1}, h_{\ell-1}) dt_r dt_{\ell-1} dt_\ell.$$

To evaluate $Z^{(i,j)}$, we must separate the computation into the cases $i < j$, $i > j$, and $i = j$,

$$Z^{(i,j)} = \begin{cases} \dfrac{w^{(j)}}{w^{(i)}} f^{(i)}, & \text{if } i < j \\ f^{(j)}, & \text{if } i > j \\ \dfrac{1}{w^{(i)}} \left(\dfrac{\rho_b \lambda_i}{n + \rho_b \lambda_i} e^{-\rho_b t_{i-1}} - 2 e^{-n(t_i - t_{i-1})/\lambda_i - \rho_b t_{i-1}} - \dfrac{\rho_b \lambda_i}{n - \lambda_i \rho_b} e^{-\rho_b t_{i-1} - 2n(t_i - t_{i-1})/\lambda_i} \right. \\ \left. + \dfrac{2n^2}{(n - \lambda_i \rho_b)(n + \lambda_i \rho_b)} e^{-\rho_b t_i - n(t_i - t_{i-1})/\lambda_i} \right) \prod_{m=1}^{i-1} e^{-n(t_m - t_{m-1})/\lambda_m}, & \text{if } i = j, \end{cases}$$

where we define

$$f^{(i)} := e^{-\rho_b t_{i-1}} + \frac{\lambda_i \rho_b}{n - \lambda_i \rho_b} e^{-n(t_i - t_{i-1})/\lambda_i - \rho_b t_{i-1}} - \frac{n}{n - \lambda_i \rho_b} e^{-\rho_b t_i}.$$

To compute the emission probabilities we define $v^{(i)}(k)$ below:

$$v^{(i)}(k) := \frac{n(\theta_\ell)^k}{\lambda_i w^{(i)} k!} e^{n t_{i-1}/\lambda_i} \prod_{j=1}^{i-1} e^{-n(t_j - t_{j-1})/\lambda_j} \sum_{j=0}^{k} c_i^{-(j+1)} \frac{k!}{(k-j)!} \left[e^{-c_i t_{i-1}} t_{i-1}^{k-j} - e^{-c_i t_i} t_i^{k-j} \right],$$

where

$$c_i := \theta_\ell + \frac{n}{\lambda_i}.$$

A.2 Computing the expected transition counts during the E-step

Naively, if we compute the expected number of transitions from state $s_{\ell-1} = (D_i, h_{\ell-1})$ to state $s_\ell = (D_j, h_\ell)$, then marginalize over the haplotypes, we obtain an $O(n^2)$ algorithm. To improve the runtime, we can decompose the probability a transition is used between locus $\ell - 1$ and ℓ into a part that depends on the absorption haplotype and a part that depends on the absorption time interval, and thus we can reduce the run time to $O(n)$. First we compute the posterior probability $A^{(\ell)}(s_{\ell-1}, s_\ell)$ that a particular transition is used between locus $\ell - 1$ to ℓ, in terms of the discretized forward and backward probabilities $F_\ell(\cdot)$ and $B_\ell(\cdot)$. Let the newly sampled haplotype have allele a at locus ℓ, so $\alpha[\ell] = a$. Then

$$A^{(\ell)}(s_{\ell-1}, s_\ell) = \frac{1}{\hat{\pi}(\alpha)} \cdot F_{\ell-1}(s_{\ell-1}) \cdot \hat{\phi}^{(\lambda)}(s_\ell | s_{\ell-1}) \cdot \hat{\xi}^{(\lambda)}(a|s_\ell) \cdot B_\ell(s_\ell).$$

Now we marginalize over the haplotypes, plugging in the transition density formula

$$\begin{aligned}
\sum_{h_{\ell-1}} \sum_{h_\ell} A^{(\ell)}(s_{\ell-1}, s_\ell) &= \frac{1}{\hat{\pi}(\alpha)} \sum_{h_{\ell-1}} \sum_{h_\ell} F_{\ell-1}(s_{\ell-1}) \cdot \hat{\phi}^{(\lambda)}(s_\ell | s_{\ell-1}) \cdot \hat{\xi}^{(\lambda)}(a|s_\ell) \cdot B_\ell(s_\ell) \\
A^{(\ell)}(D_i, D_j) &= \frac{1}{\hat{\pi}(\alpha)} \sum_{h_{\ell-1}} \sum_{h_\ell} F_{\ell-1}(s_{\ell-1}) \cdot \hat{\xi}^{(\lambda)}(a|s_\ell) \cdot B_\ell(s_\ell) \left(y^{(i)} \delta_{s_{\ell-1}, s_\ell} + z^{(i,j)} \frac{n_{h_\ell}}{n} \right) \\
&= \frac{1}{\hat{\pi}(\alpha)} \left[\delta_{i,j} y^{(i)} \left(\sum_h F_{\ell-1}(D_i, h) \hat{\xi}^{(\lambda)}(a|D_i, h) B_\ell(D_i, h) \right) \right. \\
&\quad \left. + z^{(i,j)} \left(\sum_{h_{\ell-1}} F_{\ell-1}(s_{\ell-1}) \right) \left(\sum_{h_\ell} \frac{n_{h_\ell}}{n} \hat{\xi}^{(\lambda)}(a|s_\ell) B_\ell(s_\ell) \right) \right]
\end{aligned}$$

which is linear in n since we are only ever summing over one haplotype. To get the expected transition counts, we then sum over all the breakpoints, so $A_{ij} = \sum_{\ell=2}^{L} A^{(\ell)}(D_i, D_j)$.

A.3 Discretizing time

With an ideal time discretization, coalescence events would be uniformly distributed across intervals, but inferring the distribution of coalescence times is equivalent to the problem of population size estimation. Our heuristic discretization procedure seeks to avoid poor discretization by using the observed spacing of SNPs in the data. Let $\mathcal{T}$ be the empirical distribution of absorption times for all the contiguous segments inferred by a posterior decoding of our dataset. Then, for a discretization with d intervals, our goal is to compute $t_1, \cdots, t_{d-1}$ such that we see the same number (i.e., $|\mathcal{T}|/d$) of absorption times in each interval.

We first tackle the problem of breaking up our data into segments with the same pairwise coalescence time, then compute the expectation of this time. The locations of ancestral recombination breakpoints divide up a sequence pair into segments that each have a single coalescence time, but we do not know these breakpoints. However, it will often be the case that all the base pairs between two adjacent SNPs will coalesce at the same time, or be split between just two different times on either side of a recombination breakpoint. Moreover, in many cases, the positional distribution of SNPs and that of recombination breakpoints will be correlated — in particular, both SNPs and recombination breakpoints will be spaced farthest apart in regions of recent coalescence time. With this rationale, we take the observed distances between SNPs as a proxy for the length distribution of nonrecombining segments. To be more specific, let $\mathcal{L}$ be the list of all lengths between adjacent SNPs for all pairs of haplotypes, and let the d empirical quantiles of $\mathcal{L}$ be bounded by $L_1, \cdots, L_{d-1}$.

Now we need the expected coalescence time of an l-base segment with no mutation or recombination. Conditional on m mutation events and r recombination events, the coalescence time for two lineages under a constant population size is distributed as $\Gamma(1+m+r, 1+l\theta+l\rho)$ (see for a derivation with mutation only), so the expected coalescence time for $m = r = 0$ is

$$\frac{1}{1 + l(\theta + \rho)}.$$

In our implementation, we drop the 1 in the denominator since this represents our prior under *constant* population sizes of two lineages coalescing at rate 1. We want to minimize the use of our prior, so we put more weight on the term related to the empirical length distribution. Putting this all together, we plug in the quantiles of $\mathcal{L}$ into this formula to obtain t_i:

$$t_i = \frac{1}{L_{d-i}(\theta + \rho)}.$$

If an approximate time range of interest is known (for example, in humans we might be interested in the last million years), then the user can specify an end-time $T_{\max}$. Then all times are scaled by $T_{\max}/t_{d-1}$.

A.4 Simulation details

The following `ms` commands were used to simulate data under three population size change histories.

S1: `ms 10 1 -T -r 10000 1000000 -eN 0.05 0.1 -eN 0.2 0.5 -eN 0.5 1.25`
S2: `ms 10 1 -T -r 10000 1000000 -eN 0 10 -eN 0.05 0.1 -eN 0.2 0.5 -eN 0.5 1.25`
S3: `ms 10 1 -T -r 10000 1000000 -eN 0 0.75`

Note that `ms` times are in units of $4N_{\text{ref}}$ generations, so we multiplied the raw times above by 2 to compare to PSMC and diCal. Mutation rates were not specified above, since the only

base	A	C	G	T
A	0.503	0.082	0.315	0.100
C	0.186	0.002	0.158	0.655
G	0.654	0.158	0	0.189
T	0.097	0.303	0.085	0.515

Table A.1: Mutation matrix for realistic human data. The rows represent the original base, and the columns represent the mutated base.

`ms` output used was tree at each base (`-T` flag). Mutations were then added to the trees using a finite sites model, the mutation matrix in Table A.1, and a mutation rate $\theta = 0.01 \times 1.443$. The factor of 1.443 accounts for the fact that this mutation matrix allows mutations that do not actually change the base (i.e., an A $\rightarrow$ A transition); see [10] for further explanation. This mutation matrix was also used for the real data analysis.

The following style of command was used to run PSMC. We used 20 iterations as described in the PSMC paper, and the same pattern of parameters we used for our SMCSD.

```
psmc -p 3+2+2+2+2+2+3 -t 7 -N 20 -r 1 -o output.psmc input.psmcfa
```

To run our method, the following style of command was used.

```
java -jar diCal.jar -i input.fasta -p params.txt -n 9 -t 5 -a "3 2 2 2 2 2 3"
```

The parameter file includes the number of loci in each sequence, the number of alleles (4 in our case), an estimate of the mutation rate, mutation matrix, and recombination rate, and the discretization. The `-n` flag specifies the number of haplotypes to use in the trunk, so there are $n + 1$ total. The `-t` flag specifies the number of threads to use; memory requirements scale linearly with this parameter. If `-t 1` was specified in the case, then `-Xmx5G` could be used for the memory requirement. The `-a` flag specifies the pattern of parameters, in an analogous fashion to PSMC. Note that these parameters have changed in the latest version of diCal available online.

A.5 Comparing diCal to PSMC

Although diCal and PSMC are both implementations of the sequentially Markov coalescent in a discrete-time framework, they have significant differences that must be considered when comparing results from the two programs. One difference is that PSMC scales all population sizes with respect to an inferred parameter $\theta_{\text{psmc}} = 4N_{\text{psmc}}\mu$. In contrast, diCal scales population sizes with respect to a fixed input $\theta_{\text{smcsd}} = 4N_{\text{smcsd}}\mu$. Neither θ is right or wrong,

they are just scaled with respect to a different N_{ref}. If we arbitrarily set $N_{\text{smcsd}} = 1$, then

$$N_{\text{psmc}} = \theta_{\text{psmc}}/\theta_{\text{smcsd}}$$

Thus when analyzing the results, we multiplied the PSMC sizes and times by N_{psmc}. We also multiplied the `ms` times by 2, since they are in units of $4N_{\text{ref}}$ generations.

To compare the performance of the two programs fairly, we gave both PSMC and diCal the same amount of data. Specifically, we compared the performance of diCal with a n-sequence leave-one-out scheme to the performance of PSMC with the same n sequences, but paired up sequentially (i.e. sequence 1 with 2, sequence 3 with 4, etc).