Statistical Inference for Population Genetics

David Balding
Professor of Statistical Genetics
Department of Epidemiology and Public Health
Imperial College of Science, Technology, and Medicine
University of London

Lectures given at:
Institute for Mathematical Sciences
National University of Singapore

March 20 – 22, 2002

1 Introduction

1.1 Outline of problem

Suppose that we have DNA sequence data from each of a sample of individuals drawn from an interbreeding population and we wish to draw inferences about quantities such as:

- mutation rates
- population size and growth rates, and time since start of growth
- the time since the most recent common ancestor of the sample
- the age of a mutation.

To keep matters simple in this short lecture course, we restrict attention here to certain special situations. Many of these restrictions could be weakened now, given more time. Other extensions are topics of current research.

- Many species of interest are predominantly diploid: there are males and females, and each individual has two versions of each chromosome, one inherited from their father and one from their mother. Our methods will apply

directly to haploid species, for which each individual has only one version of each chromosome. They also apply to the haploid parts of the human genome: the Y chromosome and mitochondrial DNA (mtDNA).

- Our methods can also be applied to diploid genomes provided that for each individual we have *haplotype data* (i.e. from one chromosome only) and also that the effects of recombination can be ignored. (Recombination is the crossing over from paternal to maternal chromosomes in the process of creating a new chromosome for transmission to an offspring.) Because recombination rates are highly variable in the human genome, some regions of only a few hundred bp seem to be strongly affected by recombination, whereas in other parts of the genome regions of tens of Kb seem to be unaffected.

- We assume *neutrality*. In the case of Y chromosome data for example, this assumption implies that the reproductive success of each man does not depend on his DNA sequence at the loci studied.

- We ignore here the effects of any population structure: the division of the population into subpopulations such that mating occurs preferentially within a subpopulation.

Although these are severe restrictions, the remaining problems are very challenging. Even in this simplified setting, it is only within recent years that likelihood-based statistical inference has become possible for DNA sequence data.

We do not need to be restrictive about mutation: with the methods to be discussed here we can handle a wide range of data types and assumptions about mutation. The data can be the base (A, C, G, or T) at each of a sequence of contiguous nucleotides, or binary indicator indicating whether or not a sequence is the same of different from a reference sequence at a series of locations, or microsatellite loci, or combinations of the above.

1.2 Example dataset

The data are drawn from 6 individuals. For each individual, data is available from 5 loci. At each locus, the data value is 0 if the individual's DNA sequence is the common, or "wild", type at this locus, and 1 otherwise:

	Locus				
Individual	a	b	c	d	e
1	0	1	0	1	0
2	0	1	0	0	0
3	0	1	0	0	0
4	0	1	0	0	0
5	0	0	0	0	1
6	0	0	0	0	0

The data in row i of the matrix will be called the *haplotype* of the ith individual. Although we refer to "loci", these could be a sequence of 5 contiguous nucleotides.

Note that all haplotypes are the same at loci a and c; these loci are said to be *monomorphic*. In many datasets the majority of loci are monomorphic. Although monomorphic loci are uninteresting for some purposes, the number of them can be important for estimating mutation rates and inferring genealogical times.

Individuals 2, 3 and 4 have the same haplotype, which may be because they are all descended from a recent common ancestor. More generally, we can construct a matrix whose entries give the "distances" between each pair of individuals, where "distance" is the number of loci at which the two individuals differ:

Individual	2	3	4	5	6
1	1	1	1	3	2
2		0	0	2	1
3			0	2	1
4				2	1
5					1

These distances are expected to have an approximately monotonic relationship with the times since the corresponding pairs of individuals last shared a common ancestor.

Other useful ways to summarise the data include the numbers of:

- distinct haplotypes: 4
- haplotypes that occur once in the sample: 3
- segregating loci (i.e. not monomorphic): 3
- singleton loci (i.e. only one individual differs from the others): 2

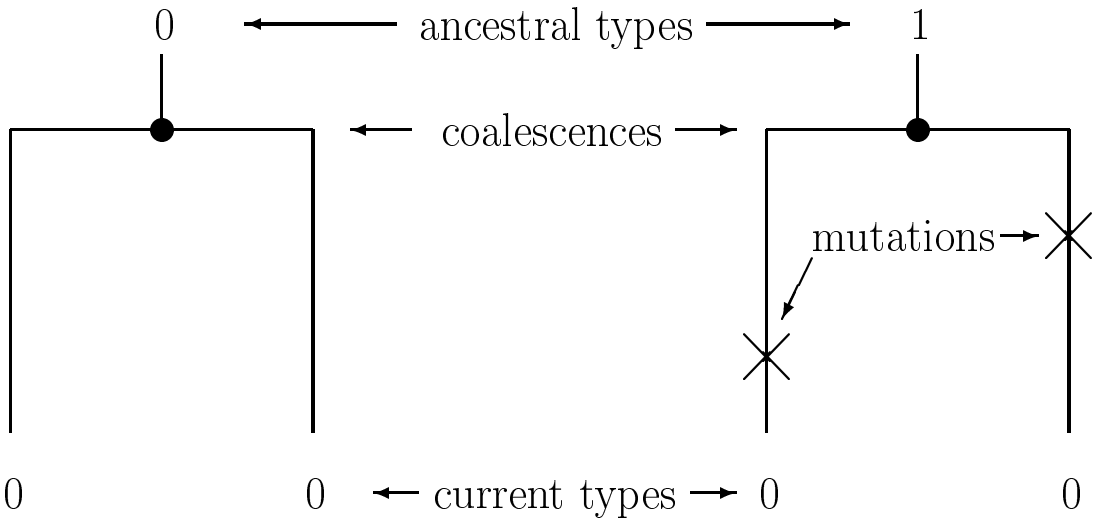
Direct interpretation of the full data or any of these summary statistics is complicated by the fundamental problem in analysing population genetics data: the complex pattern of dependencies. These dependencies arise because of the shared

inheritance: for example two or more haplotypes may have a recent common ancestor and hence share many mutations, while another group of haplotypes may have no recent ancestry in common.

To tackle this problem, we model the genealogical history of the observed sample. In the absence of recombination, there is a single genealogical tree underlying the sample, with one leaf for each haplotype. In the absence of selection, mutations occur independently on the genealogical tree.

1.3 Modelling assumptions

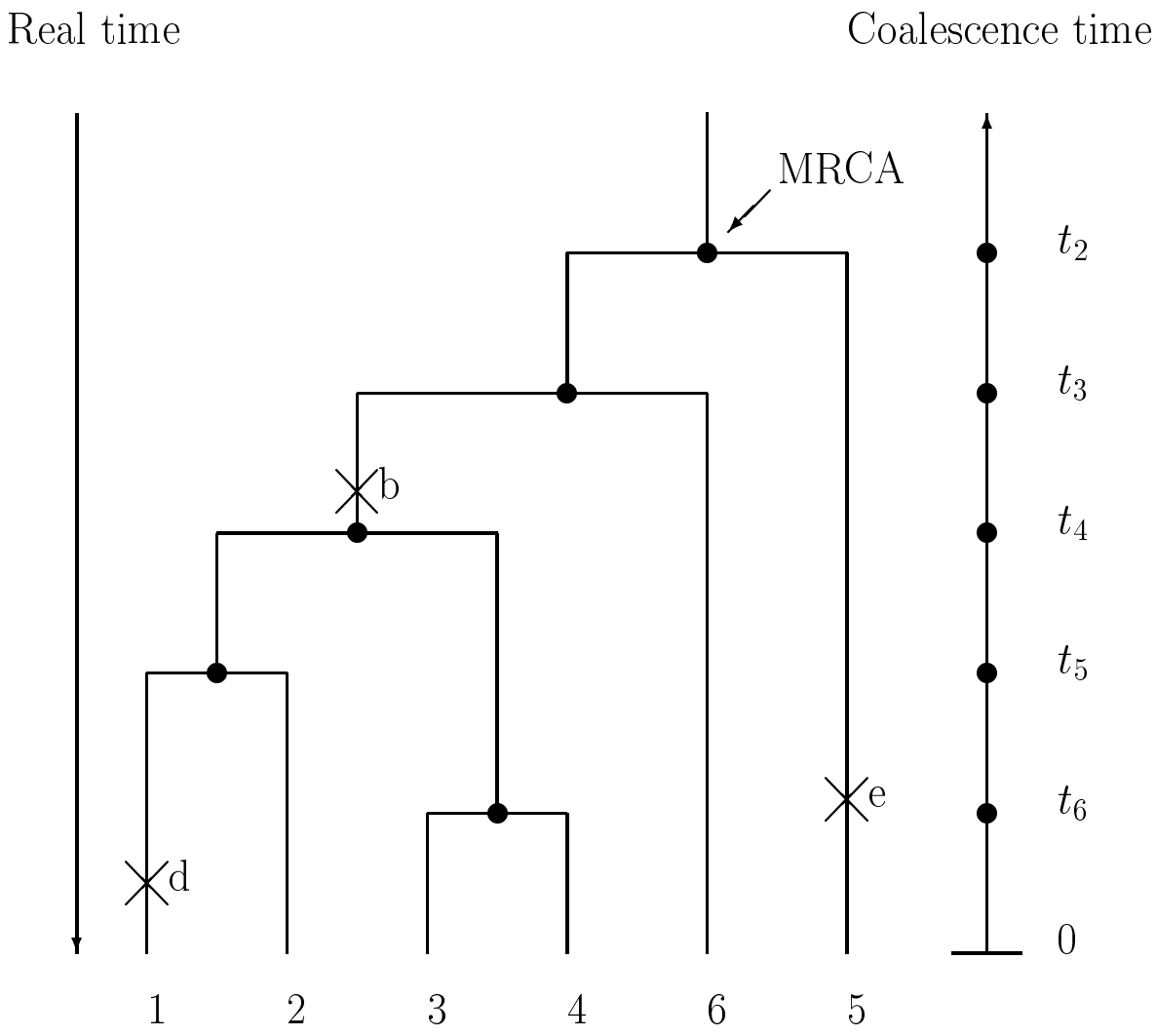
One specific difficulty with interpreting the example dataset is that two individuals may have the same allelic type at a locus because they both inherited this type from a recent common ancestor, or they may have arrived at the same state via independent mutation events. The no-mutation and a possible two-mutation scenario at a single locus are represented below:



Another two-mutation scenario has ancestral state 0 and two opposite mutations on the same branch. If mutation is rare, the two-mutation scenarios are less likely than the no-mutation scenario. One common assumption is that the two-mutation scenarios *never* occur: at each locus, there is at most one mutation event underlying the observed variation. This is the so-called "infinite sites" assumption. Although unlikely to be strictly true, it is approximately true for many human datasets because mutation rates are typically about 10^{-8} per bp per generation, whereas it is typically only a few thousand generations since the MRCA (most recent common ancestor) of a sample. Under the infinite-sites assumption, the number of mutations which have occurred in the history of the sample since its MRCA is exactly the number of segregating loci.

A related problem is that the ancestral haplotype, that is the haplotype of the MRCA, may be unknown. A common assumption is that the most frequent type in the sample is the ancestral type. Alternatively, it may be possible to infer the ancestral haplotype by obtaining the haplotype of an individual in a closely related species (such as chimpanzee in the case of humans).

If the ancestral state is assumed known, and the infinite-sites assumption is made, then many genealogical histories can be excluded as inconsistent with the data. For the example dataset and assuming the ancestral haplotype to be 00000, the four individuals having the mutant type (i.e. 1) at locus b must form a cluster in the tree. Also the mutation at b must preceded the mutation at d. One possible genealogy consistent with these constraints is:



1.4 Coalescent models: simulation

We will draw on coalescent theory to provide models for the unknown genealogical history of the sample. Recall that time in the standard coalescent is measured backwards, and in units of N generations, where N is the effective population size. The time to the next coalescent event when there are currently j lineages has the $\exp(j(j-1)/2)$ distribution. Mutations occur at points of a Poisson process at rate $\theta/2$, where $\theta = 2N\mu$ and μ is the mutation rate per locus per generation.

The standard coalescent arises as an approximation to the genealogy of a sample drawn from a large, constant-size, random-mating population.

Since very little statistical inference can be performed exactly under the standard coalescent model, it is necessary to be able to simulate random samples from it in order to carry out approximate methods of inference. There are two main approaches to simulation, backwards in time and forwards in time.

Algorithm 1: Backwards in time

First simulate the vector $\mathbf{w} = (w_1, w_2, \ldots, w_{n-1})$ of times between coalescence events, ordered backwards in time, where w_i has the $\exp((n-i+1)(n-i)/2)$ distribution. Then the ith coalescence event (backward in time) occurs at time

$$t_i = \sum_{j=1}^{i} w_j$$

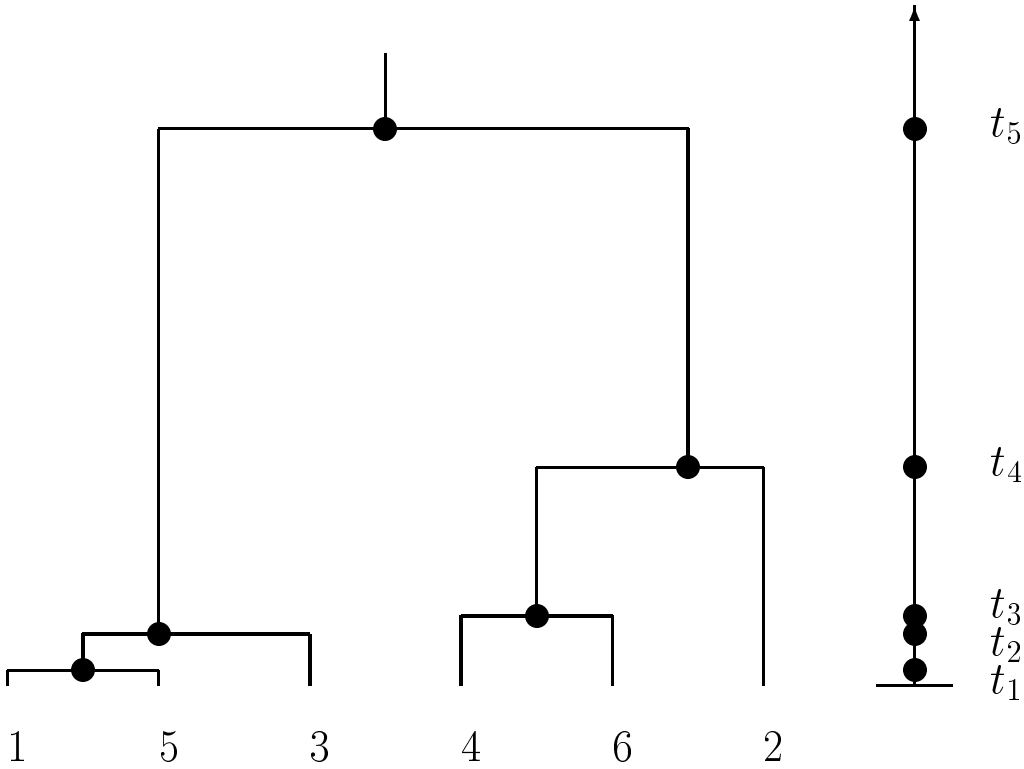
For example, with $n = 6$ we may simulate (to 2 decimal places):

$$\mathbf{w} = (0{\cdot}05, 0{\cdot}12, 0{\cdot}06, 0{\cdot}49, 1{\cdot}12).$$

(Recall that the expected value of $\mathbf{w}$ is (0·07,0·10,0·17,0·33,1·00).) The coalescence time vector corresponding to the above simulated value of $\mathbf{w}$ is then approximately (0·05,0·17,0·23,0·72,1·84).

Next, for $i = 1, 2, \ldots n-2$, simulate a pair of integers chosen uniformly without replacement in $\{1, 2, \ldots, n-i+1\}$. For example, with $n = 6$ we may choose ((1,5),(1,3),(3,4),(2,3)). The ith pair represents the two lineages which coalesce at time t_i. At the last coalescence event (backwards in time), i.e. when $i = n-1$, the pair must be (1,2). In order to interpret these pairs we need a system for relabelling lineages following the ith coalescence event, so that the remaining lineages are labelled from 1 to $n-i$. Here, if the coalescing pair are (j, m) with $j < m$, we label the new lineage j and subtract one from the label of all lineages with current label $> m$. For example, if there are currently 5 lineages and lineages 2 and 4 coalesce, the combined lineage is labelled 2 and lineage 5 is relabelled 4.

The example values given above of $\mathbf{w}$ and the coalescing pairs lead to the following tree:



Now for the mutation events. For each of the $2(n-1)$ branches of the tree, simulate k independent Poisson$(\theta l/2)$ random variables, where k is the number of loci and l is the length of the branch. The jth value is the number of mutations occurring along that branch at the jth locus. Typically $\theta l/2 \ll 1$ and so realised numbers of mutations will usually be zero, with occasional ones. However multiple mutations at a locus can occur on a single branch, or on distinct branches. If it is desired to create a dataset satisfying the infinite-sites assumption, for any locus at which multiple mutations arise in the simulation, erase all but one of them, the one retained being chosen uniformly at random. Alternatively, instead of the Poisson, simulate a Bernoulli$(\theta L/2)$ random variable which indicates whether or not a mutation arises at that locus, where L is the total branch length of the tree, and if a mutation arises choose its branch in proportion to the branch lengths.

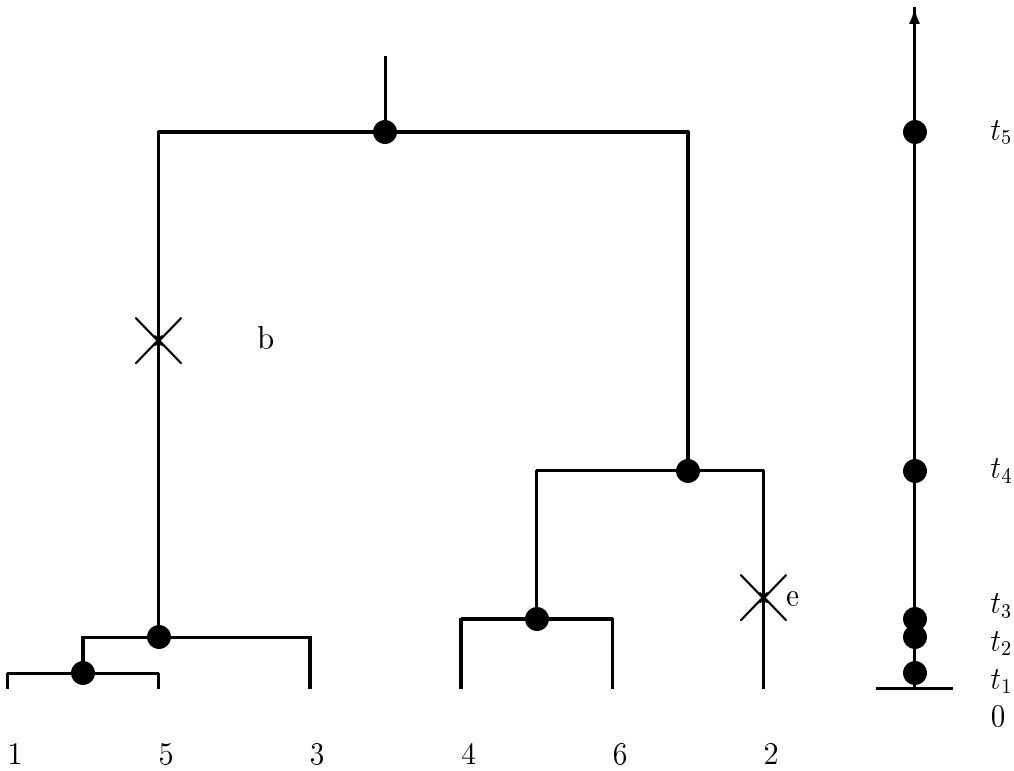
For our example tree illustrated above,

$$L = \sum_{i=1}^{5} (7-i) w_i = 4{\cdot}85,$$

just above its expected value:

$$\mathrm{E}[L] = \sum_{j=1}^{5} \frac{2}{j} \approx 4{\cdot}57.$$

If the scaled mutation rate per locus is $\theta = 0{\cdot}2$, then the expected number of mutations at each locus is $0{\cdot}485$. Suppose that we have five loci, labelled a, b, c, d, and e, and the realised numbers of mutations at each locus is 0, 1, 0, 0, and 1, the mutations occurring on the branches indicated below:



Finally, to generate a dataset, start at the root of the tree, the MRCA, and assign its haplotype to be either a fixed value such as 00000, or chosen randomly according to a given distribution. Follow the lineages forward in time altering the haplotype according to the mutation events occurring along each branch, if any. At each branching point the current haplotype is duplicated with one copy following each branch.

For our example simulation, assuming ancestral haplotype 00000 we can represent the resulting dataset as:

	Locus				
Individual	a	b	c	d	e
1	0	1	0	0	0
2	0	0	0	0	1
3	0	1	0	0	0
4	0	0	0	0	0
5	0	1	0	0	0
6	0	0	0	0	0

where 1 in row i and column j indicates that a mutation has arisen in the history of individual i at locus j since the MRCA of the sample. More generally, the mutation process may have a further random element, such as the direction (up or down) of a single-step microsatellite mutation.

Most of the complexity in turning the above algorithm into computer code rests in tracking the branch on which the mutation events occurred. For some purposes this may not be of interest, and it may suffice to record only the total number of mutations in the tree at each locus. In this case the above algorithm can be greatly simplified. It is necessary only to generate that w_i, calculate L and generate a Poisson($L\theta/2$) random variable for each locus. S-plus code for a function performing the above simulation is as follows:

```
qcoal <- function(niter=10000, nsamp=6, nloc=5, theta=0.2)
{
  ns1 <- nsamp-1
  rate <- (ns1:1)*(nsamp:2)/2
  w <- matrix(rexp(ns1*niter,rate),ns1,niter)
  L <- apply((nsamp:2)*w,2,sum)
  nmut <- matrix(rpois(nloc*niter,rep(L,nloc)*theta/2),niter,nloc)
  nmut
}
```

A run of the above algorithm with default parameters produced an average of 2·27 mutations per simulation, close to its expectation of 2·28. The sample variance was 3·67 (expectation 3·75). Recall that the variance of a Poisson random variable equals its mean; the additional variance arises here because the value of L is common across loci at a simulation: if L is small there will tend to be few mutations at every locus, and vice-versa.

Similarly, at 92% of simulated loci there was at most one mutation, and so the probability that none of the 5 loci has more than one mutation would be $(0{\cdot}92)^5$ or about 65% if each locus were independent. In fact, about 68% of simulations satisfied this criterion, a statistically significant difference because of the large number of simulations (10 000).

Algorithm 2: Forwards in time

First, choose the ancestral haplotype as above. This is assigned to two haplotypes immediately after they branched from their MRCA (a branching event is the same as a coalescence, but now viewed forward in time). Starting with $j = 2$, simulate an $\exp(j(j+\theta-1)/2)$ random variable, the time to the next event. Next simulate a type for this event: it is a mutation with probability $k\theta/(k\theta+j-1)$, otherwise it is a branching event. At a branching event, one of the existing haplotypes is replicated,

and j is increased by one. At a mutation event, the haplotype and locus at which the mutation occurs are chosen uniformly at random, and any other random feature of the mutation process simulated from its appropriate distribution. The process is repeated until $j = n+1$, when the haplotype most recently created is discarded.

In the S-plus code below, only the resulting haplotype data is output, not the times of events or any property of the genealogical tree. Mutations are labelled by powers of 2, so that output values that are not a power of two result from more than one mutation.

```
forcoal <- function(nsamp=6, nloc=5, theta=0.2)
{
  ns1 <- nsamp-1
  hap <- matrix(0,nsamp+1,nloc)
  nmu <- rep(0,nloc)
  j <- 2
  while(j<=nsamp)
    if(runif(1) < nloc*theta/(j-1+nloc*theta))
    {
      mut <- c(sample(j,1),sample(nloc,1))
      hap[mut[1],mut[2]] <- hap[mut[1],mut[2]]+2^nmu[mut[2]]
      nmu[mut[2]] <- nmu[mut[2]]+1
    }
    else
    {
      hap[j+1,] <- hap[sample(j,1),]
      j <- j+1
    }
  hap[1:nsamp,]
}
> forcoal()
     [,1] [,2] [,3] [,4] [,5]
[1,]    0    0    0    1    0
[2,]    0    0    0    0    0
[3,]    0    0    0    0    0
[4,]    0    0    0    0    0
[5,]    0    0    0    1    0
[6,]    0    0    0    0    1
> forcoal()
     [,1] [,2] [,3] [,4] [,5]
[1,]    1    0    0    6    1
[2,]    0    1    0    1    2
[3,]    0    1    0    1    2
```

```
[4,]    0    1    0    1    2
[5,]    0    1    0    1    2
[6,]    0    1    0    1    2
```

In the first simulation output shown above, exactly two mutations arose, at loci 4 and 5. In the second simulation, one mutation arose at each of loci 1 and 2, three arose at locus 4 and two at locus 5. At locus 4, the oldest mutation is shared by haplotypes 2 through 6, while the two more recent mutations arise in the history of haplotype 1 only.

The forward-in-time approach to simulating from the standard coalescent is harder to extend to more general versions of the coalescent model. However, for the standard coalescent it is much easier to program than the backward approach, if the goal is to simulate full datasets. The resulting code may be less efficient because the number of steps in the algorithm is random, making it harder to develop efficient S-plus code using matrices rather than `for` loops. Ten thousand iterations of the above algorithm using the S-plus commands:

```
> ss <- 1:10000
> for(i in 1:10000) ss[i] <- sum(apply(forcoal(),2,sum)>0)
```

required a few minutes on a laptop computer, compared with a few seconds for a `qcoal` simulation of the same size. The results showed an average of 1·76 segregating loci per simulation. In 16·74% of simulations there was no variation at any of the 5 loci, compared with 16·95% for the `qcoal` simulations and a theoretical expectation of 16·67%.

2 Methods of Statistical Inference

2.1 Classical hypothesis testing

One approach to statistical inference is to compare the observed dataset with datasets that would have arisen under a model of interest given an assignment of values to the unknown parameters. The model is regarded as implausible if the observed dataset does not resemble the hypothetical replicates. This requires that datasets be ordered according to their similarity with the observed dataset, which is usually achieved by replacing datasets with the corresponding values of a summary statistic.

For example, consider our example dataset introduced above in section 1.2, and let us compare datasets using the summary statistic S, equal to the number of segregating loci. Under the standard coalescent model with infinite-sites mutation, there is only one free parameter: the scaled mutation rate θ.

To evaluate a specific hypothesis about θ, for example $H_0 : \theta = 1$, we need to obtain the sampling distribution of S under the model, as a function of θ. We can accurately approximate this distribution via simulation. For example, using the `qcoal` function in S-plus we obtain

```
> table(apply(qcoal(,,,1)!=0,1,sum))/10000
      0      1      2      3      4      5
 0.0039 0.0189 0.0604 0.1386 0.2825 0.4957
```

The value 0·1386, for example, is a binomial proportion, and so we can assess the standard deviation due to sampling to be about

$$\mathrm{SD} \approx \frac{1}{100}\sqrt{0{\cdot}1386 \times 0{\cdot}8614} \approx 0{\cdot}0035.$$

We see that when $\theta = 1$ the most likely value of S under our model is 5, but $\mathrm{P}(S = 3) \approx 14\%$ and so we cannot reasonably dismiss $\theta = 1$ on the basis of our observation. Repeating with $\theta = 2$ we obtain

```
> table(apply(qcoal(,,,2)!=0,1,sum))/10000
      0      1      2      3      4      5
 0.0002 0.0019 0.0084 0.0327 0.1293 0.8275
```

Now, $\mathrm{P}(S{\leq}3)$ is estimated to be 4·3%, with SD $\approx$ 0·22 %, and so we can reject $H_0 : \theta = 2$ at the 5% significance level. Note that all tests must be one-sided here because of the restricted range of S: values of S large enough to reject H_0 in favour of $H_1 : \theta > 2$ cannot arise.

```
> table(apply(qcoal(,,,0.05)!=0,1,sum))/10000
      0      1      2      3      4      5
 0.5811 0.3144 0.0868 0.0165 0.0011 0.0001
```

indicating that we can reject $H_0 : \theta = 0{\cdot}05$ with p-value $< 2\%$.

Classical hypothesis testing is relatively easy to carry out, and is still the only feasible approach in some settings. However, a major limitation is that the full dataset cannot be exploited, and there is usually no sufficient statistic available for the parameter of interest. There is thus always some loss of information in going from full data to test statistic. This may be substantial and it cannot be measured. The classical hypothesis testing approach is difficult to extend to simultaneous tests of many parameters of interest and to handle multiple nuisance parameters: these situations often arise in population genetics problems.