`ms` - a program for generating samples under neutral models

Richard R. Hudson

May 29, 2007

This document describes how to use `ms`, a program to generate samples under a variety of neutral models. The purpose of this program is to allow one to investigate the statistical properties of such samples, to evaluate estimators or statistical tests, and generally to aid in the interpretation of polymorphism data sets. The typical data set is obtained in a resequencing study in which the same homologous segment is sequenced in several individuals sampled from a population. The classic example of such a data set is the Adh study of Kreitman(1983) in which 11 copies of the Adh gene of *Drosophila melanogaster* were sequenced. In this case, the copies were isolated from 11 different strains of *D. melanogaster* collected from scattered locations around the globe.

The program `ms` can be used to generate many independent replicate samples under a variety of assumptions about migration, recombination rate and population size to aid in the interpretation of such polymorphism studies. The samples are generated using the now standard coalescent approach in which the random genealogy of the sample is first generated and then mutations are randomly place on the genealogy (Kingman, 1982; Hudson, 1990; Nordborg, 2001). The usual small sample approximations of the coalescent are used. An infinite-sites model of mutation is assumed, and thus multiple-hits and back mutations do not occur. However, when used in conjunction with other programs, finite-site mutation models or micro-satellite models can be studied. For example, the gene trees themselves can be output, and these gene trees can be used as input to other programs which will evolve the sequences under a variety of finite-site models. These are described later.

The program is intended to run on Unix, or Unix-like operating systems, such as Linux or MacOsX. The next section describes how to download and compile the program. The subsequent sections described how to run the program and in particular how to specify the parameter values for the simulations.

If you use `ms` for published research, the appropriate **citation** is:

Hudson, R. R. (2002) Generating samples under a Wright-Fisher neutral model of genetic variation. **Bioinformatics 18**: 337-338.

Downloading and compilation

All relevant files including this pdf file are included in the tar file ms.tar available at `http://home.uchicago.edu/~rhudson1`. Download this tar file to your machine then extract the files from the archive with: `tar -xvf ms.tar` . After extracting, type `cd msdir`, then compile the programs by typing:

```
gcc -o ms ms.c streec.c rand1.c -lm
```

or

```
gcc -o ms ms.c streec.c rand2.c -lm
```

or alternatively, typing `./clms`, which contains this compilation line with `rand1.c`. The choice of these compilations depends on which random number generator one has available. `rand1.c` and `rand2.c` call `drand48()` and, `rand()`, respectively. I make no claims concerning the properties of these generators, and it would be straightforward to replace these by another generator, such as one of those described in **Numerical Recipes in C** (Press et al., 1992).

With `rand1.c` and `rand2.c`, one can specify the three seeds on the command line, with `-seeds x1 x2 x3` where the x's indicate the three integer seed values. If no `-seeds` switch appears on the command line, `ms` looks for the file "`seedms`" to find the seed values for initializing the random number generator. If no `seedms` file is found the generator is seeded with a default value. When the simulation is finished, the state of the random number generator is output to `seedms`. In this way, each time `ms` is invoked a new set of results is produced. If one wishes to produce the same set of samples, `seedms` can be edited to contain the value(s) indicated on the second line of the output file.

In all cases, the seeds are printed on the second line of the output file, so that the same set of samples can be generated again if desired.(The program can also be compiled with `rand1t.c` or `rand2t.c`, which use the system clock for seeding the generators and don't utilize the file `seedms` at all.)

The basic command line:

`ms` nsam nreps `-t` θ

The above line shows the simplest usage of `ms` which generates samples under the basic neutral model, with constant population size, no recombination, panmixis, and an infinite-sites model. In this case there are three arguments to `ms`: nsam, nreps, and, following the switch "`-t`", the parameter θ. The two arguments, nsam and nreps are required and must appear in this order.(Although there are exceptions, most of the switches can appear in any order.) `nsam` is the number of copies of the locus in each sample, and `nreps` is the number of independent samples to generate. The third parameter here is the mutation parameter, $\theta = (4N_0\mu)$, where N_0 is the diploid population size and where μ is the neutral mutation rate for the entire locus. (Note: As described later,when the -I switch

is used, `ms` can generate samples under models with population structure. In those cases N_0 is the number of diploids in each subpopulation.) At least one of the options, `-t -s` , `-T` or `-L` must be used. The latter three options are described later. After `nsam` and `nreps`, any or all other switches with their parameters can be read from a file using `-f filename`.

Here is an example command line:

```
ms 4 2 -t 5.0
```

In this case, the program will output 2 samples, each consisting of 4 chromosomes, generated assuming that $\theta = 5.0$.

The output

The output from the example command in the previous section would look like this (the exact output will depend on the random number generator) :

```
ms 4 2 -t 5.0
27473 36154 10290


//
segsites: 4
positions: 0.0110 0.0765 0.6557 0.7571
0010
0100
0000
1001


//
segsites: 5
positions: 0.0491 0.2443 0.2923 0.5984 0.8312
00001
00000
00010
11110
```

The first line of the output is the command line. The second line shows the random number seeds(described in the "Downloading and compilation" section.) Following these two lines are a set of lines for each sample. Each sample is preceded by a line with just "//" on it. (With tbs arguments some other numbers appear on this line following the "//" .) That line is followed by "`segsites:`" and the number of polymorphic sites in the sample. Following that line is a line

which begins with "`positions:`" which is followed by the positions of each polymorphic site, on a scale of (0,1). The positions are randomly and independently assigned from a uniform distribution. (With recombination, the distribution is somewhat more complex.) Following the positions, the haplotypes of each of the sampled chromosomes are given, each as a string of zeros and ones. The ancestral state is coded with a zero, and the mutant, or derived state, indicated with a one. When a sample has no polymorphic sites, the line with positions and the sample haplotypes are omitted.

When the option `-L` is used the output includes for each sample, a line giving the time of the most recent common ancestor of the sample and the sum of the lengths of the branches of the gene tree, each in units of $4N_0$ generations. This line appears right after the "//" and begins with "`time:`".

Output gene trees:

`ms` nsam nreps `-T`

When the option `-T` is used the trees representing the history of the sampled chromosomes are output. For example, the command line `ms 5 2 -T` results in the following output:

```
ms 5 2 -T
3579 27011 59243


//
((2:0.074,5:0.074):0.296,(1:0.311,(3:0.123,4:0.123):0.187):0.060);


//
(2:1.766,(4:0.505,(3:0.222,(1:0.163,5:0.163):0.059):0.283):1.261);
```

This output represents the trees for two samples. The tree format is the Newick format utilized by Phylip and a number of other applications. The branch lengths are in units of $4N_0$ generations. The sampled chromosomes are labelled `1, 2 ...` corresponding to ordered sampled chromosomes. This labelling is irrelevant with unstructured models, but with island models described later, the labelling can be important. This command can be used together with the `-t` or `-s` commands.

With recombination a tree is output for each segment within which no recombination has occurred in the history of the sample.

The gene trees output by `ms` can be used as input to other programs. To illustrate, we use `seq-gen`, the sequence evolution program of Rambaut and Grassly(1997) which can simulate the evolution of sequences under a variety of mutation models. The following lines generate 2 samples of sequences 40 base pairs long, according to a simple finite-sites model with θ per base pair of 0.2.

```
ms 5 2 -T | tail +4 | grep -v // >treefile
seq-gen -mHKY -l 40 -s .2 <treefile >seqfile
cat seqfile
 5 40
1        GGCGTCCGGCCAAAGGTTCTTGACACACGATACCTTAGTT
2        GGCGTCCGGCCAAAGGTTCTTGACACACGATACCTTAGTT
3        GGCGACCTGACAAAGGGTCTTGACACACGATACGTTAGTT
4        GGCGACCGGACAAAGGGTCTTGACACACGATACGTTAGTT
5        GGCGACCGGACAAAGGGTCTTGACACACGATACGTTAGTT
 5 40
3        ACCAGGGCTCGGAACTCTCTGGGTTTGCCTAAGTGCCCTT
2        ACCAGGGCTCGGATCTCTCTGGGTTTGCCTAAGTGCCCTT
5        ACCAGGGCTCGGATCTCTCTGGGTTTGCCTAAGTGCCCTT
1        ACCAGAGCTCGCTACTATCCGGGTGTGCCTAAGATCCCAC
4        ACCAGAGCTCGCTACTTTACGGGTGTGCCTTAGTGGCGCA
```

The `tail +4` command is used to strip off the first four lines output by ms. The `grep -v //` command removes the lines with double slashes from the `ms` output. The `-mHKY` command for `seq-gen` specifies (in this case) the Jukes-Cantor mutation model. The `-s` command for `seq-gen` scales the branch lengths so that θ per site is 0.2, in this case. This can also work with recombination but `seq-gen` requires that one specify the maximum number of recombinational segments. One cannot know this ahead of time but one can guess that the number of segments will be less than say $10\rho\sum_{i=1}^{n-1}\frac{1}{i}$ Here is an example, for generating 2 samples of sequences 1000 base pairs long with $\rho = 3.0$:

```
ms 5 2 -T -r 3. 1000 | tail +4 | grep -v // >treefile
seq-gen -mHKY -l 1000 -s .2 -p 50  <treefile >seqfile
```

The `-p` option for `seq-gen` specifies the maximum number of recombinational segments. The argument of the `-l` option of `seq-gen` must match the number of sites of the `-r` option of `ms`. The `-r` option of `ms` is described in a later section.

It may be of interest to know the time of the most recent common ancestor of each sample. The command line switch, `-L`, will generate an additional line of output for each sample, containing the word, "time: " followed by the time back to the most recent common ancestor of the sample in units of $4N_0$ generations. With recombination, the time printed is the time of the most recent common ancestor for the site in the exact center of the segment simulated. Also printed on this line is the total tree length, that is, the sum of the lengths of the branches of the gene tree, also in units of $4N_0$ generations. With recombination, the total tree length printed is a weighted average across sites of total tree length, such that the product of the printed quantity and θ is the expected number of segregating sites, conditional on the coalescent history of the sample.

Fixed number of segregating sites:

`ms` nsam nreps `-s` segsites

As described in Hudson(1993) , one may in some cases wish to generate samples in such a way that every sample has the same number of segregating sites (regardless of the size of the genealogy generated.) This is done by using the `-s` option on the command line. For example,

```
ms 10 3 -s 7
```

will output 3 samples, each consisting of 10 chromosomes. Each sample will have exactly 7 polymorphic sites. It should be emphasized that this method of generating samples is not equivalent to specifying a value of θ, and then conditioning on some fixed number of polymorphic sites. Rather, these samples are produced by randomly placing a fixed number of mutations on each genealogy generated.

If both the `-s` option and the `-t` option are used, there will be one additional line of output for each sample. This line, which appears after the segsites line, begins with `prob:` which is followed by the probability of the specified number of segregating sites given the genealogical history of the sample and the value of θ specified with the `-t` option. For example, `ms 5 2 -t 3.0 -s 5` produces the following output:

```
ms 5 2 -t 3.0 -s 5
3579 27011 59243


//
segsites: 5
prob: 0.145142
positions: 0.0803 0.1516 0.2276 0.8501 0.8636
01110
10000
00001
00001
10000


//
segsites: 5
prob: 0.0709507
positions: 0.0856 0.1591 0.2881 0.2901 0.6601
11000
11010
01001
01001
00100
```

Crossing over:

`ms` nsam nreps `-t` θ `-r` ρ nsites

`ms` will generate samples under a finite-sites uniform recombination model (Hudson, 1983). In this model, the number of sites between which recombination can occur is finite and specified by the user (with the parameter nsites). Despite the finite number of sites between which recombination can occur, the mutation process is still assumed to occur according to the "infinite-sites" model, in that no recurrent mutation occurs, and the positions of the mutations are specified on a continuous scale from zero to one, as in the no-recombination case. To include crossing-over (recombination) in the model, one uses the `-r` switch and specifies the cross-over rate parameter, ρ which is $4N_0r$, where r is the probability of cross-over per generation between the ends of the locus being simulated. In addition, the user specifies the numbers of sites, nsites, between which recombination can occur. One should think of nsites as the number of base pairs in the locus. For example, to simulate samples of size 15 for a locus which is 2,501 base pairs long with cross-over probability between adjacent base pairs of 10^{-8} per generation, and assuming $N_0 = 10^6$ one enters:

```
ms 15 1000 -t 10.04 -r 100.0 2501
```

In this case, $\rho = 100$, since the cross-over probability between the two ends of the locus is $10^{-8}(2501-1) = 2.5\times10^{-5}$, and thus $\rho = 4\times10^6\times2.5\times10^{-5} = 100$. (In addition we have assumed that the neutral mutation rate is 10^{-9} per site to obtain θ.) Once again, although we obtained θ by multiplying the number of sites by the mutation rate per site, the program assumes an infinite-sites model in which no recurrent mutation occurs.

Crossing-over and gene conversion:

`ms` nsam nreps `-t` θ `-r` ρ nsites `-c` f λ

In addition to crossing-over, gene conversion (intra-locus non-cross-over exchange) can be included in the simulations. We assume a model like that of Wiuf and Hein(2000). Gene conversion is assumed to initiate between a specified pair of sites in a given chromosome with probability g. We let f denote the ratio, g/r, where r is the probability per generation of crossing-over between adjacent sites.(The parameters g and r are the same parameters that appear in Wiuf and Hein.) The parameter f is input to the program, as well as the mean conversion tract length, λ, after the `-c` switch. (The mean track length is equal to $1/q$ in the notation of Wiuf and Hein.) The conversion tract is assumed to extend, say to the right, for a random number of sites (denoted L), which is geometrically distributed:

$$Prob[L = j] = p^{j-1}(1-p), \quad j = 1, 2, \ldots \ .$$

where

$$p = \frac{\lambda - 1}{\lambda}.$$

To carry out simulations with gene conversion but no crossing-over, one uses the `-r` option with ρ equal to zero, and the `-c` option. In this case value following the `-c` option is the value of $4N_0g$, rather than the ratio g/r.

Exponentially growing or shrinking population size:

`ms` nsam nreps `-t` θ `-G` α

The switch `-G` is used to specify that the population has been growing (or shrinking) exponentially. That is, the population size is given by: $N(t) = N_0 \exp^{-\alpha t}$, where t is the time before the present, measured in units of $4N_0$ generations. (A negative value of α means that the population was larger in the past than at present.) Negative values of α are allowed, however, other past demographic events must also be specified to prevent a situation where no coalescent event ever occurs. These additional demographic events are specified with the `-e` options described below.

It is important to realize that the parameters θ, ρ and $4N_0m$ (described later) are defined in terms of the present population size, N_0.

Under island models, specified with the `-I` option described below, the `-G` option specifies that **all** subpopulations grow at the same rate. (See switch `-g` for specifying different growth rates for different subpopulations.)

Spatial structure and migration:

`ms` nsam nreps `-t` θ `-I` npop n_1 n_2 ... $[4N_0m]$

The program, `ms`, can produce samples under island models with arbitrary migration rates, and different population sizes. To produce samples under an island model, one adds the switch `-I` followed by the number of subpopulations, npop, and the sample configuration. The sample configuration is a list of npop integers (n_1 n_2 ...) indicating the number of chromosomes sampled from each subpopulation. (Note: The subpopulations are numbered from 1 to npop.) By default all subpopulations are assumed to consist of the same number of diploids, denoted N_0. For a symmetric island model, one can enter the migration parameter $4N_0m$ after the sample configuration as illustrated in the template above. In this case, each subpopulation receives migrants at the same rate from each of the other subpopulations. (m is the fraction of each subpopulation made up of new migrants each generation.) The migration rate parameter is optional. If this parameter is not provided, the migration rate is set to zero. Thus the command line:

```
ms 15 1000 -t 2.0 -I 3 10 4 1 5.0
```

will generate 1000 samples of size 15, where the first 10 chromosomes in each sample are from subpopulation 1, the next 4 chromosomes are sampled from subpopulation 2, and one chromosome is sampled from subpopulation 3. (If, in this example, the `-T` option were used, the chromosomes from the first subpopulation would be labelled `1` to `10`, the chromosomes from the second subpopulation would be labelled `11` to `14`, etc.) In this example all three subpopulations are assumed to be the same size, and receive migrants at the same rate from each of the other subpopulations. The migration parameter, $4N_0m$ is 5.0 .

When the `-I` switch is employed, the mutation parameter and the recombination parameters, entered with the `-t` and `-r` options, are defined in terms of the subpopulation size rather than the total population size. That is, θ is defined as $4N_0\mu$, with N_0 being the subpopulation size, and similarly ρ is $4N_0r$, where N_0 is the subpopulation size. Again, in the output, the first n_1 haplotypes represent the sampled chromosomes from subpopulation 1, and the following n_2 haplotypes are from subpopulation 2, etc.

`ms` will also produces samples under models with arbitrary backward migration matrices. This is done by first using the `-I` option and its arguments (with or without the migration rate parameter.) This establishes the number of subpopulations, the sample configuration, and assigns preliminary values to the elements of the migration matrix. The elements of the migration matrix are $M_{ij} = 4N_0m_{ij},\ i,j = 1,\ldots,$`npop` , where m_{ij} is the fraction of subpopulation i which is made up of migrants from subpopulation j each generation. The `-I` option sets all elements of the migration matrix, $M_{ij}, i \neq j$, equal to $4N_0m/$(npop-1) (or to the default value zero.) To modify any non-diagonal elements of this matrix, one can use the `-m` or `-ma` options as described next.

The command line:

```
ms 15 1000 -t 2.0 -I 3 10 4 1 5.0 -m 1 2 10.0 -m 2 1 9.0
```

will generate samples as described in the last example but with a slightly modified migration matrix. In this case all elements of the matrix are 2.5 (= 5.0/(3-1)), except elements M_{12} and M_{21} which have been set to 10.0 and 9.0, respectively. Any number of `-m` commands can be specified. The `-ma` option is used to specify an entire migration matrix. The option is followed by a list of all the elements of the migration matrix: $M_{11}, M_{12}, M_{13}, M_{21}, M_{22}, \ldots$. As indicated, entries for the diagonal elements, $M_{i,i}$, must be included on the command line, but are meaningless and ignored by the program. Here is an example:

```
ms 15 1000 -t 10.0 -I 3 10 4 1 -ma x 1.0 2.0 3.0 x 4.0 5.0
6.0 x
```

where the migration parameters are $M_{12} = 1.0, M_{13} = 2.0, M_{21} = 3.0, M_{23} = 4.0, M_{31} = 5.0$ and $M_{32} = 6.0$. It is handy to indicate the diagonal elements with x's, or any symbol one chooses to make the matrix more readable.

It is the user's responsibility to provide sample configurations, migration matrices and past demographic events for which the sampled chromosomes will

eventually coalesce. If the user specifies a sample configuration from subpopulations that do not communicate, the program will, in some cases, happily continue tracing the lineages back in time, forever.

Another suboption of `-I` is the `-n` option which allows the user to specify that a particular subpopulation has a different size relative to N_0. For example, following a `-I` option with its required arguments, the option `-n 3 .5` sets the population size of the third subpopulation to $0.5 * N_0$. (Note that this command does not change N_0 and will not change any of the parameters, m_{ij}, or any of the elements of the migration matrix, M_{ij}, which are defined in terms of N_0 and the m_{ij}. However the number of migrant copies arriving each generation from a particular population, or from all other subpopulations, to population 3 is half as large as a consequence of this command. This is because the fraction of migrants is the same, but the total number of individuals is half what it would have been without the `-n` command.) To set other subpopulations to other sizes, additional `-n i size` commands are appended.

The `-G` option sets the growth rate of all subpopulations to one value (as indicated before.) To set individual subpopulations to have different growth rates, the `-g i` α_i command is used to set the growth rate of subpopulation i to α_i.

It is important to realize that `-m`, `-ma`, `-n` and `-g` are suboptions of -I and must appear only after the `-I` option has been used to establish the number of subpopulations (npop) and the sample configuration. `-G` α commands can appear before or after `-I` commands, and will in either case set the growth parameters for all subpopulations to the same value. Note however, that a `-g i` α_i command that precedes a `-G` command on the command line will be overridden by the `-G` command and thus will have no effect. Thus `-g` options should only appear after any `-G` options.

Past demographic events:

To specify that demographic parameters change at specific times in the past, the `-e` switches are used. These switches are: `-eG`, `-eg`, `-eN`, `-en`, `-em`, `-ema`, `-es` and `-ej`. In each case the first parameter following the switch is the time when the demographic change occurred, measured from the present in units of $4N_0$ generations. In all cases, the parameter values specified apply to the time interval immediately farther in the past from the time point specified. (Henceforth, the term "past-ward" will be used to indicated farther in the past.) The arguments which follow the time parameter specify subpopulations and other relevant parameters, as indicated in the following list:

`-eG` t α	Set all growth rates to α at time t.
`-eg` t i α_i	Set growth rate of subpop i to α_i at time t.
`-eN` t x	Set all subpop's to size $x * N_0$ and growth rates to zero.

`-en` t i x Set subpop i size to $x * N_0$ at time t and growth rate to zero.

`-eM` t x Set all elements of the migration matrix, M_{ij}, to $x/(\text{npop}-1)$ at time t.

`-em` t i j x Set $M_{ij}(= 4N_0 m_{ij})$ to x at time t. m_{ij} is the fraction of subpopulation i made up of migrants each generation from subpopulation j.

`-ema` t npop M_{11} M_{12} M_{13} M_{21} ... Assign new values to all elements of the migration matrix.

`-es` t i p Split subpopulation i into subpopulation i and a new subpopulation, labeled $npop + 1$. Each ancestral lineage in subpopulation i is randomly assigned to subpopulation i with probability p and subpopulation $npop + 1$ with probability $1 - p$. The size of subpopulation $npop + 1$ is set to N_0. Migration rates to and from the new subpopulation are assumed to be zero and the growth rate of the new subpopulation is set to zero. Subpopulation i retains the same growth rate and migration rates as before the event. In the forward direction this corresponds to population admixture. The size, growth rates and migration parameters for the new subpopulation can be immediately modified by following the `-es` command with appropriate additional `-e` commands. Remember, that if changed population size and growth rates are desired at the same time point, that one must put the size change command first followed by the growth rate change command. This is because the size change command changes the growth rate to zero.

`-ej` t i j Move all lineages in subpopulation i to subpopulation j at time t. Migration rates from subpopulation i ($M_{ki}, k \neq i$) are set to zero. (Prior to May 19, 2007, ms left migration rates unchanged .) In the forward direction this corresponds to population splitting. Population growth rates are unchanged.

It is important to note that `ms` generates genealogical histories by working back from the present, and that each of these `-e` commands changes the parameters for the period immediately past-ward (farther back in the past from) the time point specified. For example, `ms 10 30 -t 4.0 -eN 0.2 .02` specifies that the population size was constant at size N_0 from the present back to time $0.2 * 4N_0$, and farther back in time the population size was $0.02 * 4N_0$. The population size change was instantaneous and occurred at time $0.2 * 4N_0$ generations before the present.

Also, note that the `-eN` and `-en` commands change the population size(s), but also have the side-effect of changing the growth rate(s) to zero. If one desires the growth rate to be non-zero in the time period immediately past-ward of a population size change, one needs to add a `-eG` or `-eg` command right after the `-eN t x` or `-en t i x` command. For example,

```
ms 15 1000 -t 6.4 -eN 0.3 0.5 -eG .3 7.0
```

In this case, the order of switches on the command line is crucial. The following is INCORRECT:

```
ms 15 1000 -t 6.4 -eG .3 7.0 -eN 0.3 0.5
```

since the the effect of the change-in-growth-rate command, `-eG .3 7.0` would be canceled by the following `-eN 0.3 0.5` command which has the side-effect of changing the growth rate to zero. Moving backward in time, `ms` invokes simultaneous demographic events in the order that the are specified on the command line. (The only other command with a side-effect is the `-ef` switch which sets some of the migration rates to zero.)

And finally, note that changes in subpopulation sizes do not change m, the migration fraction for any subpopulation. Thus, if one changes a subpopulation size, it implicitly changes the number of migrants that arrive per generation into a subpopulation. For example, consider the command: `ms 10 100 -t 2.0 -I 2 5 5 6.0 -en 0.1 2 .25` . The phrase, `-I 2 5 5 6.0`, specifies that there are two subpopulations, each of size N_0, with migration rate $4N_0m = 6.0$. If we denote the subpopulation sizes by N_1 and N_2, these sizes in the most recent interval of time are $N_1 = N_2 = N_0$. And each subpopulation receives $2N_0m = 3.0$ migrant copies of the gene per generation. (Recall that m is the fraction of the population which are new migrants each generation, and since the subpopulation size is N_0 diploids, the number of migrant copies is $2N_0m$.) The phrase `-en 0.1 2 .25` specifies that past-ward of this event, the size of subpopulation 2, call it N_2, is one quarter of its size in recent times. But m is not changed so the fraction of subpopulation 2 which are new migrants each generation is still m. So the number of migrants arriving in subpopulation 2, $2N_2 * m$, which is only 0.75 per generation, past-ward of time point 0.1. If one wishes the number of migrant copies to be maintained at 3 per generation, one must increase the migration parameter, `-em 0.1 2 1 24.0`. With or without the later phrase, subpopulation 1 still receives three migrant copies per generation.

Using "tbs" arguments

`ms` nsam nreps `-t tbs -r tbs nsites`

In some cases, it may be useful to permit parameter values to change from one sample to the next. For example, one may want to have the parameter values for each sample, drawn from a probability distribution. One could do this by using a separate program to generate many shell commands, each running `ms` to generate a single sample, with the random parameter values being specified. This would be inefficient and slow. `ms` can do this more efficiently using what I call "tbs" arguments. This is done as follows. On the command line, instead of specifying a single value for a parameter, one simply types "tbs", without the quotes. The string "tbs" indicates that the parameter value is "**t**o **b**e **s**pecified"

later. For each sample to be generated the program will read in the parameter values from `stdin`. This can be done for any number of numerical parameters on the command line (but not for nsam, nreps, the number of segregating sites specified with the `-s` switch, or the number of populations, specified after a `-I` switch.) In the example above this paragraph, both theta and rho values are to be read in for each sample. For example, if file `thetarho` contains the two lines:

```
3.0 5.0
3.5 8.5
```

then the following command:

```
ms 5 2 -t tbs -r tbs 1000 <thetarho
```

results in the following output:

```
ms 5 2 -t tbs -r tbs 1000
32503 42481 21397

//      3.0     5.0
segsites: 5

positions: 0.2400 0.2434 0.8538 0.8825 0.9456
00010
00010
11000
11100
11001

//      3.5     8.5
segsites: 6

positions: 0.0643 0.1013 0.6817 0.7189 0.9118 0.9282
000010
000010
100100
001011
010000
```

In this example, the first sample was produced with $\theta = 3.0$ and $\rho = 5.0$, while the second sample was produced with $\theta = 3.5$ and $\rho = 8.5$. The parameter values read in for each sample, are output after the "//". As noted above, any numeric parameter value on the command line, except nsam, nreps, the number of segregating sites, and the number of populations, can be replaced with "tbs". The numbers read in from `stdin` are sequentially used to replace the tbs arguments in command line. `ms` will stop producing samples when it runs out of parameter values for replacing the "tbs" arguments or when it

has produced `nreps` samples, whichever happens first. Examples using "tbs" arguments are included in the `ms.tar` file. The `-f` option cannot be used with tbs arguments.

Some examples:

Instantaneous population size changes

Here is an example of a command line using a `-e` command :

```
ms 15 1000 -t 2.0 -eN 1.0 .1 -eN 2.0 4.0
```

which specifies that the population size was one-tenth its current size between $4N_0$ and $8N_0$ generations ago, and prior to that the population size was 4 times the current population size.

Generating an outgroup sequence

Here is an example showing how an outgroup sequence can be produced:

```
ms 11 3 -t 2.0 -I 2 1 10 -ej 6.0 1 2
```

This command establishes that there are two subpopulations with zero migration between them. One sampled chromosome (the outgroup sequence) from subpopulation one, and 10 chromosomes from subpopulation two. The two subpopulations can be considered separate species. The `-ej` command moves all lineages from subpopulation 1 to subpopulation 2 at time $24N_0$ generations before present. In the output the outgroup sequence appears first on the list of each sample. The ten ingroup sequences follow.

Instantaneous size change followed by exponential growth

In this example, we have one panmictic population that underwent a size reduction, followed by a period of constant size, followed by population expansion. This demographic history is shown in figure 1. To be concrete let us suppose that the population sizes are $N_1 = 10,000, N_2 = 5,000$, and $N_3 = 20,000$. Also suppose that the neutral mutation rate is 10^{-8} per site per generation and that we are considering a segment 8,000 base pairs long. In this case, if we take N_0 to be 20,000, we have $\theta = 4 * 20,000 * 10^{-8} * 8,000 = 6.40$. Suppose that T_1 is 16,000 generations, or 16,000/(4*20,000) = 0.2 in units of $4N_0$ generations. Similarly, suppose that T_2 is 24,000 generations or 0.3 in units of $4N_0$ generations. To specify the exponential growth on the command line we need to calculate the growth parameter α. To obtain α we solve the equation, $5,000 = 20,000 * \exp^{-\alpha 0.2}$, which is $\alpha = -(1/0.2) * \log(5,000/20,000) = 6.93$. Thus the command line to generate 1000 samples of size 15 is:

```
ms 15 1000 -t 6.4 -G 6.93 -eG 0.2 0.0 -eN 0.3 0.5 .
```

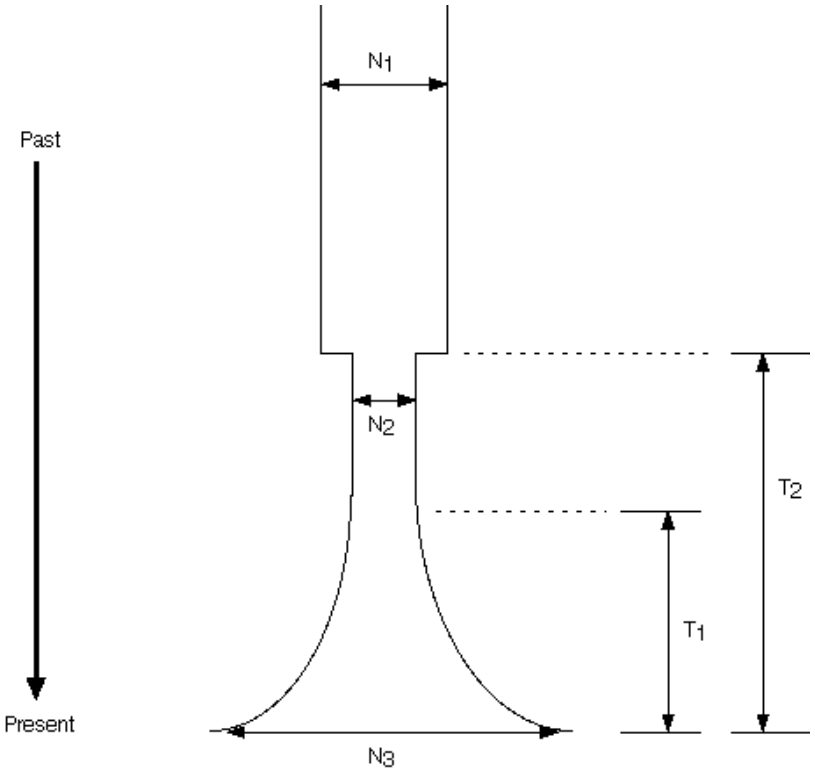

Figure 1: An example demographic history with population size changes. In this example, the population had an ancestral population size of N_1 which at time T_2 instantaneously shrank to size N_2. (All times are measured from the present.) It remained constant at size N_2 until T_1 time units before present, at which point it began expanding exponentially until the present at which point the population size is N_3. The command line to simulate under this model is described in the text.

The phrase `-G 6.93` specifies that the population decreases as we go back in time according to the equation, $N(t) = N_0 \exp^{-6.93t}$. The phrase `-eG 0.2 0.0` specifies that the growth rate changes to zero at time 0.2, and the population size preceding this time will be $N_0 \exp^{-6.93*0.2} = N_0 * 0.25$. The phrase `-eN 0.3 0.5` specifies that the population size before 0.3 times units was half of N_0.

Two species with population size differences

In this example, we consider a population with two subpopulations which were derived from a single ancestral population. The sample consists of three copies drawn from one subpopulation and 12 copies drawn from a distinct isolated subpopulation. The demographic history is shown in figure 2. We suppose that

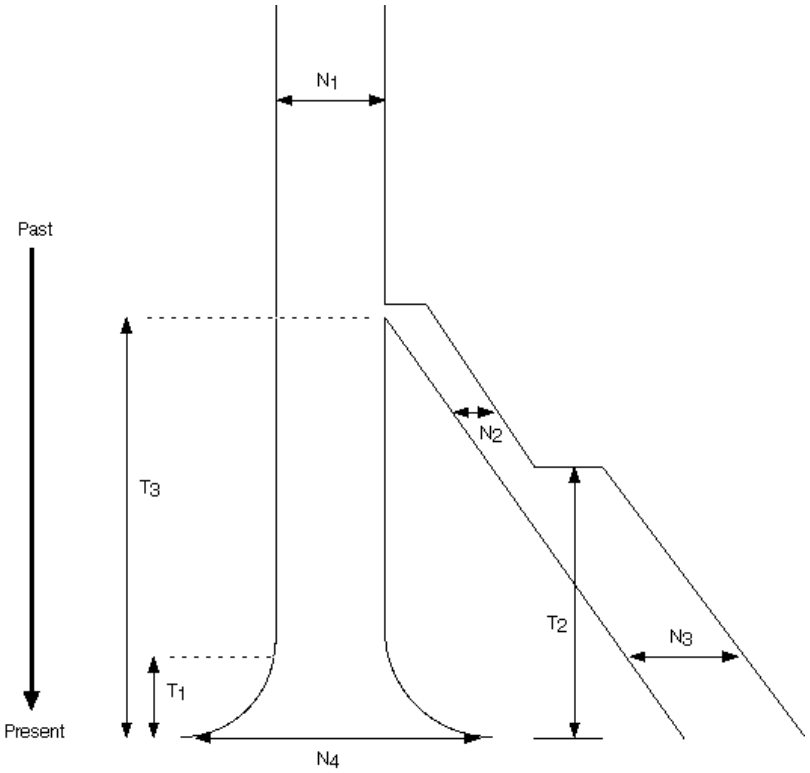

Figure 2: A demographic history in which a subpopulation splits off from an ancestral population. In this example, a population of ancestral size N_1, began growing exponentially T_1 time units in the past and reached a present size of N_4. At time T_3, it gave rise to an instantaneously isolated subpopulation which initially was of size N_2, and which at time T_2, instantaneously expanded to size N_3, and remained at that size until the present. The command line for simulating samples under this model is described in the text.

$N_1 = 10,000, N_4 = 40,000, T_1 = 5,000$ generations, $T_3 = 15,000$ generations, $T_2 = 10,000$ generations, $N_2 = 2,000$ and $N_3 = 5,000$. We also suppose that the mutation per site per generation is 10^{-8}, and that we are interested in a segment 7,000 base pairs long. Let us take N_0 to be 40,000. (As will be discuss below, we can assign N_0 to be any value we choose if we make appropriate adjustments in command line parameters.) In this case, $\theta = 10^{-8} * 4 * 40,000 * 7,000 = 11.2$. We need to convert all times into units of $4N_0$ generations, which means simply dividing the times by 4*40,000 = 160,000. So the times are $T_1 = 5,000/160,000 = 0.03125, T_2 = 10/160 = 0.0625,$ and $T_3 = 5/160 = 0.09375$. Since the first population decreases from 40,000 to 10,000 in .03125 time units, we find the growth rate by solving $10,000 = 40,000 * \exp^{-\alpha * 0.03125}$. Thus, $\alpha = -\ln(10,000/40,000)/0.03125 = 44.36$. Thus, here is the command line:

```
ms 15 100 -t 11.2 -I 2 3 12 -g 1 44.36 -n 2 0.125 -eg 0.03125 1
0.0 -en 0.0625 2 0.05 -ej 0.09375 2 1
```

Here are explanations for each item on the command line:

`ms 15 100` Generate 100 samples of size 15.

`-t 11.2` Sets θ to 11.2 as explained above.

`-I 2 3 12` Establishes that there are two isolated subpopulations, with three sampled copies from subpopulation one and 12 sampled from subpopulation two (for a total of fifteen as indicated by the first command line argument.) Migration rate is not specified, so it is set to the default value of zero.

`-g 1 44.36` Sets growth parameter of subpopulation one to 44.36, as calculated above.

`-n 2 0.125` Sets the initial size of subpopulation 2 to $0.125 * N_0$ (since N_3 is one-eighth of $N_4 = N_0$.)

`-eg 0.03125 1 0.0` Specifies that the growth rate for subpop 1 changes to zero at time 0.03125.

`-en 0.0625 2 0.05` Specifies that subpopulation 2 size changes to $0.05 * N_0$ at time 0.0625 .

`-ej 0.09375 2 1` Specifies than lineages of subpopulation 2 are moved to subpopulation 1 at time 0.09375.

Note that we can assign N_0 to have any convenient value if we make all the appropriate changes in the times and other parameters. For example, if we set $N_0 = 10,000$, then because N_0 is one quarter of what is was, theta is one fourth as large ($= 11.2/4$), all the times are four times larger, and the growth rate is one quarter as large as before. In this case, the present population size for subpopulation 1 is 4 times N_0, and the present population size of subpopulation 2 is one-half of N_0 which also must be specified on the command line. Then the command line becomes:

```
ms 15 100 -t 2.8 -I 2 3 12 -g 1 11.09 -n 1 4.0 -n 2 0.5 -eg 0.125
1 0.0 -en 0.25 2 .2 -ej 0.375 2 1
```

The statistical properties of samples produced with the alternative command line are identical to those produced by the first one.

Stepping stone model with recent barrier

In this example (illustrated in figure 3) we set up 6 subpopulations with migration between subpopulations 1 and 2, between 2 and 3, between 4 and 5, and between 5 and 6, only. At time, say T= 2.0 time units we remove the barrier between 3 and 4. The command line is :

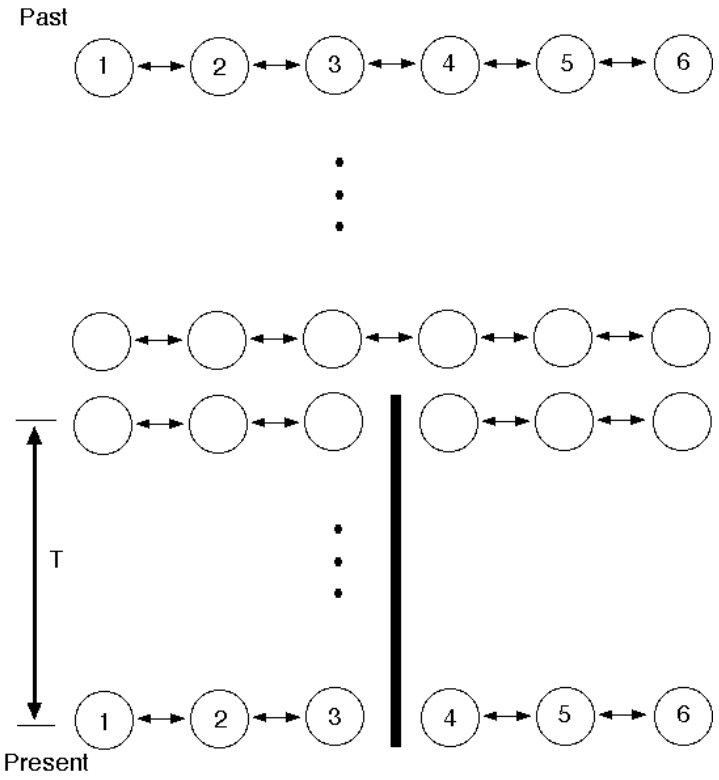

Figure 3: A demographic history of six subpopulations that exchange migrants in a stepping-stone model. At a time T time units in the past a barrier to gene flow arose, such that no further gene flow occurs between subpopulation 3 and subpopulation 4. The command line to generate this history is described in the text.

```
ms 15 100 -t 3.0 -I 6 0 7 0 0 8 0 -m 1 2 2.5 -m 2 1 2.5 -m 2 3 2.5
-m 3 2 2.5 -m 4 5 2.5 -m 5 4 2.5 -m 5 6 2.5 -m 6 5 2.5 -em 2.0 3 4
2.5 -em 2.0 4 3 2.5
```

The phrase, `-I 6 0 6 0 0 6 0`, sets up 6 subpopulations with zero migration rate between them and establishes that a sample of size 7 is taken from subpopulation 2 and a sample of size 8 is taken from subpopulation 5. (In the output the first 7 haplotypes are from subpopulation 2 and the next 8 are from subpopulation 5. The "`-m` " commands set up migration, $4Nm = 2.5$, between the neighboring subpopulations (except between subpopulation 3 and 4). The "`-em`" commands modify the migration matrix at time 2.0 in the past such that pastward of this time, migration at rate $4Nm = 2.5$ occurs also between subpopulation 3 and 4.

Summary of command line options

-f filename Read switches and parameters from "filename".

`-seeds x1 x2 x3` Specify seeds for the random number generator.

`-t` θ Set value of $4N_0u$.

`-s` j Make samples with fixed number of segregating sites, j.

`-T` Output gene trees.

`-L` Output the time to the most recent common ancestor and the sum of the branch lengths in units of $4N_0$ generations.

`-r` ρ **nsites** Recombination: $\rho = 4N_0r$ where r is the recombination rate between the ends of the segment being simulated (not the per site rate) and nsites is the number of sites between which recombination can occur.

`-c f` λ Gene conversion: $f = g/r$, where g is the conversion rate per site. (If $r = 0, f = 4N_0g$.) λ is the mean conversion tract length. Even if $\rho = 0$, the `-r` switch must be invoked to specify nsites.

`-G` α Set growth parameter to α. If island model, set growth rate of all subpopulations to α.

`-I` npop n1 n2 ... [$4N_0m$] Assume island model with symmetric migration rate, $4N_0m$, and sample configuration n1, n2 ... .

- **-n** i x Set size of subpopulation i to $x * N_0$.
- **-g** i α_i Set growth rate of subpopulation i to α_i.
- **-m** i j M_{ij} Set (i, j) element of migration matrix to M_{ij}.
- **-ma** M_{11} M_{12} ... M_{21} M_{22} ... ... Assign values to the elements of migration matrix.

`-eG` t α Set all growth rates to α at time t.

`-eg` t i α_i Set growth rate of subpop i to α_i at time t.

`-eN` t x Set all subpop's to size $x * N_0$ and growth rates to zero.

`-en` t i x Set subpop i size to $x * N_0$ at time t and growth rate to zero.

`-eM` t x Set all elements of the migration matrix to $x/(\text{npop} - 1)$ at time t.

`-em` t i j x Set $4N_0m_{ij}$ to x at time t. (m_{ij} is the fraction of subpopulation i made up of migrants each generation from subpopulation j.)

`-ema` t npop M_{11} M_{12} M_{13} ... M_{21} ... Assign new values to the migration matrix.

`-es` t i p Split subpopulation i into subpopulation i and subpopulation npop+1. Each lineage in subpopulation i is randomly assigned to subpopulation i with probability p and subpopulation npop +1 with probability $1-p$. Migration rate to and from new subpopulation is assumed to be zero and its growth rate is set to zero. The size of the new subpopulation is set to N_0. Subpopulation i retains same growth rate and migration rates as before the event. In the forward direction this corresponds to population admixture.

`-ej` t i j Move all lineages in subpopulation i to subpopulation j at time t. Migration rates from subpopulation i ($M_{ki}, k \neq i$) are set to zero. Growth rates are unchanged. In the forward direction this corresponds to population splitting.

Using `ms` with sample_stats

Included in the `ms.tar` file is a program `sample_stats.c` which will process the output of `ms` and calculate a small set of sample statistics for each generated sample. (`sample_stats` is compiled as follows: `gcc -o sample_stats sample_stats.c tajd.c -lm`.) For example, the output of:

```
ms 30 4 -t 3.0 | sample_stats
```

is

```
pi: 7.708046    ss: 25  D:  0.788423    thetaH: 7.533333    H:  -0.174713
pi: 0.519540    ss: 6   D:  -1.874712   thetaH: 3.894253    H:  3.374713
pi: 1.429885    ss: 5   D:  0.362688    thetaH: 4.087356    H:  2.657471
pi: 3.190805    ss: 15  D:  -0.529742   thetaH: 0.809195    H:  -2.381609
```

The output consists of one line for each sample. On this line is the average number of pairwise differences between sequences (labeled pi),the number of segregating sites (labeled ss:), Tajima's D (labeled D:), and Fay and Wu's θ_H (labeled thetaH) and the difference between θ_H and `pi`. If both the `-t` and `-s` switches are used, the output contains one additional column which contains the values of "prob". If tbs arguments are used the values of the parameters are output at the end of each line.

One can further process the output of `sample_stats` to obtain mean, standard deviation and percentile estimates of any of these quantities using the program, `stats.c` also provided in `ms.tar`. For example,

```
ms 30 10000 -t 3.0 -eN .2  0.1  |  \
>   sample_stats |  cut -f 6 | stats .025 .5 .975
```

will output the estimated mean, standard deviation, 2.5th percentile, median, and the 97.5th percentile of the distribution of Tajima's D for a model in which the population has experienced a recent rapid increase in size. (In this example, the population size was one-tenth the current size until $0.2 * 4N_0$ generations

ago.)(`cut` is a Unix utility which can "cut" out a specified column of a file. Since Tajima's D is output in the 6th column of the output of `sample_stats` we use "cut -f 6" to get the D values.)

References

Hudson, R. R. (1983). Properties of a neutral allele model with intragenic recombination. *Theoretical Population Biology*, 23:183–201.

Hudson, R. R. (1990). Gene genealogies and the coalescent process. In Futuyma, D. and Antonovics, J., editors, *Oxford Surveys in Evolutionary Biology, Vol. 7*, pages 1–43. Oxford University Press, Oxford.

Hudson, R. R. (1993). The how and why of generating gene genealogies. In Takahata, N. and Clark, A. G., editors, *Mechanisms of Molecular Evolution*, pages 23–36. Sinauer Associates, Sunderland, MA.

Kingman, J. F. C. (1982). On the genealogy of large populations. *J. Appl. Probab.*, 19A:27–43.

Kreitman, M. (1983). Nucleotide polymorphism at the alcohol dehydrogenase locus of drosophila melanogaster. *Nature*, 304:412–7.

Nordborg, M. (2001). Coalescent theory. In Balding, D. J., Bishop, M. J., and Cannings, C., editors, *Handbook of Statistical Genetics*, pages 179–212. John Wiley and Sons, Chichester, UK.

Press, W. H., Teukolsky, S. A., Vetterling, W. T., and Flannery, B. P. (1992). *Numerical Recipes in C*. Cambridge University Press, Cambridge.

Rambaut, A. and Grassly, N. C. (1997). Seq-gen: An application for the monte carlo simulation of dna sequence evolution along phylogenetic trees. *Comput. Appl. Biosci.*, 13:235–238.

Wiuf, C. and Hein, J. (2000). The coalescent with gene conversion. *Genetics*, 155:451–62.