`MIMAR` - a program to estimate parameters of the isolation-migration model from recombining loci

C. Becquet

December 17, 2010

This document describes how to use `MIMAR`, a program that estimates the demographic parameters of an "isolation-migration" model from recombining loci (see Figure 1). `MIMAR` provides estimates of five parameters: the population mutation rates per base pair (bp) for the two descendant populations, θ_1 and θ_2, and for the ancestral population, θ_A, the time since the populations split in generations, T, and the symmetrical migration rate, m (Figure 1). In its current implementation, the method requires resequencing data from two populations (or closely related species) at multiple independently-evolving loci, and an outgroup sequence.

Note: One needs an outgroup sequence for each locus in order to calculate these statistics, as they require knowledge of which allele is ancestral or derived. `MIMAR` is intended for use on highly diverged populations or closely related species and may not provide precise estimates unless data sets have either shared and fixed alleles between the two samples - although note that `MIMAR` has been used successfully on human data without any fixed polymorphism.

The method uses Markov Chain Monte Carlo (MCMC) to explore the posterior distributions of the parameters given the data. Briefly, the data are summarized for each locus by the four following statistics (related to the ones studied by Wakeley and Hey, 1997): The number of derived polymorphisms unique to the samples from populations 1 and 2 (S_1 and S_2 respectively), the number of shared derived alleles between the two samples (S_s), and the number of fixed alleles in either sample (S_f). The prior distributions for the model parameters are specified by the user. For each locus and step of the MCMC, a set of genealogies or, in the case of a recombining locus, a set of ancestral recombination graphs (ARGs, Hudson, 1983), is generated by the coalescent under the isolation-migration model with those parameters, using a modified version of `ms` (Hudson, 2002). `MIMAR` then estimates the likelihood by calculating the probability of the data summaries at all the loci given the set of genealogies and the parameters. Finally, `MIMAR` outputs a sample from the posterior distribution of the parameters given the data summaries obtained using MCMC (see Becquet and Przeworski, 2007).

After estimating the parameters of the isolation-migration model, I recommend performing a goodness of fit test on the results of `MIMAR`, to ensure that the estimated model fits the data. I provide the program `MIMARgof` to help you perform such a test.

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 describe how to run the program and in particular how to specify the prior distributions. Since `MIMAR` was written using `ms`, some of the descriptions below have been adapted or copied (with the permission from R. Hudson) from the file `"msdoc.pdf"` provided here and with `ms` (Hudson, 2002). I strongly suggest that you first read `"msdoc.pdf"` and, if possible, experiment with `ms` (specifically with the split model with two populations: switches "`-I 2` n_1 n_2 M `-ej` T `2 1`") before using `MIMAR`.

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

Becquet, C. and Przeworski, M., 2007. A new approach to estimate parameters of speciation models, with application to apes. *Genome Res.* **17**:1505−1519.

Contents

What changed since the last version?

- December 17, 2010: Minor change in mimargofdir.

- March 18, 2010: I added objects and functions in order to specify the random seeds from the command line. The files `"mimar.c"`, `"mimar.h"`, `"params.c"`, `"rand1.c"`, `"rand2.c"`, `"rand1t.c"` and `"rand2t.c"` were changed.

- February11, 2010: I added the file `"mimar_noanc.c"`. The program `MIMAR_noanc` estimates the parameters of the model from data for which no ancestral sequence is available. A section about this option was added to `"MIMARdoc.pdf"`.

- December 28, 2009: The files `"mimar.c"` and `"params.c"` of `MIMAR` were changed:

  - The locus-specific recombination rates are now calculated by $\rho_y = w_y\rho(Z_y-1)$ instead of $w_y\rho(Z_y)$.
  - I added several options to obtain the locus-specific population recombination rates.
  - I changed the instances of “calloc(1...” by “malloc(...”.

  I updated `"MIMARdoc.pdf"` accordingly and tried to clarify many points in the documentation. Note that the scaling of the recombination rate can be confusing (at least for me) so be careful when setting recombination information. Thanks to Peter Andolfatto who helped me through my confusions on getting the recombination rates right.

- November 27, 2009:

  - The file `"mimar.c"` of `MIMAR` was changed.: I fixed potential memory leaks.
  - The file `"streec.c"` of `MIMAR` was changed: I fixed a bug on memory allocation that occurred when $n_{1y} + n_{2y} \gg n_{11} + n_{21}$ for any $y \in [2, Y]$. Thanks to Yongshuai Sun who mentioned the bug to me.

- March 3, 2009:

  - The file `"params.c"` of `MIMAR` was changed: I corrected a bug on memory allocation. Thanks Susan J. Miller for mentioning this bug to me.
  - The file `"mimar.c"` of `MIMAR` was changed so that now when the prior on gene flow is “`1` a b”, a and b can both be negative and the summary output file show the estimate of the gene flow rate. Thanks to Camille Roux for highlighting this problem to me.
  - The documentation was changed. I added a figure to help calculate the summary statistics.

- November 5, 2008: The files `"mimar.h"` and `"params.c"` of `MIMAR` were changed: I added the switch “`-R`” and the Perl script `"perlRELAUNCHmimar"`. These allow one to relaunch `MIMAR` from an interrupted run. I updated `"MIMARdoc"` accordingly.

- May 29, 2008: The file `"params.c"` of `MIMAR` was changed: I removed several check flags to allow greater freedom to the user. Thanks Armando Geraldes for mentioning the problem to me.

- January 20, 2008: I wrote some additional explanation on the calculation of the summary statistics in `"MIMARdoc"`. The program was not changed.

- September 18, 2007: The files "params.c" and "mimar.h" of MIMAR were changed. In the previous version, the locus names could not start with a number. Now the locus names can be any string of up to 50 characters.

Downloading and compiling

All relevant files are included in the tar file "mimar.tar" available at http://przeworski.uchicago.edu/cbecquet/download.html. Download this tar file to your machine, then extract the files from the archive with: "tar -xvf mimar.tar". After extracting, type "cd mimardir/" and compile the program by typing:

```
gcc -o mimar mimar.c params.c streec.c randX.c -lm
```

(X is either 1, 2, 1t or 2t) or alternatively, by typing make, which contains this compilation line with optimization and rand1.c.

The choice of compilation depends on which pseudo-random number generator the user has available. "rand1.c" and "rand2.c" call drand48() and drand(), respectively. These pseudo-random number generators could also be replaced by another generator, such as one of those described in **Numerical Recipes in C**. With "rand1.c" and "rand2.c", MIMAR first looks for the file "seedmimar" to find the seed values for initializing the random number generator. If no "seedmimar" file is found, the generator is seeded with a default value. In either case, the seed is printed on the second line of the summary output file, so that the exact same analysis can be generated again if desired. When the estimation procedure is finished, the state of the random number generator is output to "seedmimar". In this way, each time MIMAR is invoked, a new analysis is produced. If you want to produce the same analyses, "seedmimar" can be edited to contain the value(s) indicated on the second line of the summary output file. (The program can also be compiled with "rand1t.c" and "rand2t.c", which use the system clock for seeding the generators and does not use the file "seedmimar" at all.)

Estimating parameters of the Isolation-Migration model

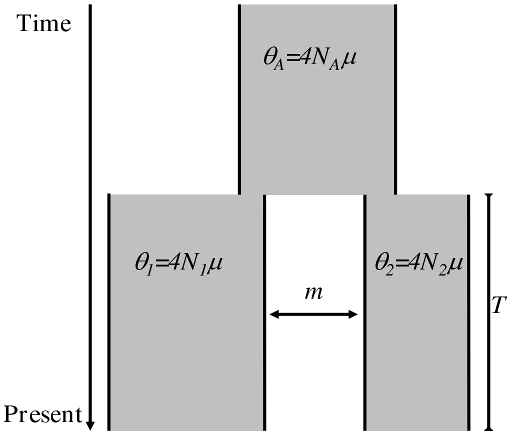


Figure 1: **The "isolation-migration" model**, in which two populations diverged T generations ago from a common ancestral population. The parameters θ_1, θ_2 and θ_A, are the population mutation rates for populations 1, 2 and the ancestral population, respectively. μ is the mutation rate per bp and N_1, N_2 and N_A are the diploid effective sizes of the first, second and ancestral populations, respectively. The split time in generations is T, m is the symmetrical migration rate between populations per generation such that, $M = 4N_1m$ is the expected number of individuals in population 2 replaced by migrants from population 1 each generation.

We consider a neutral model in which an ancestral population suddenly splits into two populations, which either diverge in isolation or continue to exchange migrants (Fig. 1). We further assume that n_1 and

n_2 chromosomes have been sampled from two populations and fully resequenced at Y randomly chosen, independently evolving loci.

The population model, often called "isolation-migration", is described by the population split time in generations, T, and three population mutation rates per bp, $\theta_1 = 4N_1\mu$, $\theta_2 = 4N_2\mu$ and $\theta_A = 4N_A\mu$ (Fig. 1). Throughout, the subscripts 1, 2 and A refer to parameters that describe populations 1, 2 and the ancestral population, respectively. Following the program `IM` (Hey and Nielsen, 2004), we assume that there is an independent estimate of the average per generation mutation rate per bp across loci (e.g. estimated from divergence), $\widehat{\mu}$, which can be used to estimate the effective population sizes from the population mutation rates (e.g., as $N_1 = \frac{\theta_1}{4\hat{\mu}}$; $\hat{\mu}$ is required and provided by the user with the switch "`-u` $\hat{\mu}$"). In addition, there is a symmetric migration rate, m, which corresponds to the fraction of a population that is replaced by migrants from the other population each generation. The migration rate is specified in terms of the expected *number* of individuals in population 2 replaced by migrants from population 1 each generation (in forward direction), $M = 4N_1m$. `MIMAR` estimates the parameters of the isolation-migration model illustrated in Figure 1. To do so, it estimates the posterior distribution $\pi(\mathbf{\Theta}|\mathbf{D}) \propto p(\mathbf{D}|\mathbf{\Theta})p(\mathbf{\Theta})$, where $\mathbf{\Theta} = (\theta_1, \theta_2, \theta_A, T, M, \mathbf{P})$, $\mathbf{D}$ is the data and $p(\mathbf{\Theta})$ denotes the prior distributions for the estimated parameters of the isolation-migration model, as well as on the set of recombination rates described below, $\mathbf{P}$. Note that any of the parameters in $\mathbf{\Theta}$ may be fixed. If you want to estimate a parameter, a prior distribution needs to be provided. Specifically, unless they are fixed, the parameters θ_1, θ_2 and θ_A, with the time in generations, T, are chosen from uniform distributions. In turn, unless M is fixed, $\ln(M)$ is chosen from a uniform distribution. In the following sections, examples are provided assuming some parameters are fixed to their known values, while other parameters are estimated with values chosen from prior distributions.

Briefly, `MIMAR`'s estimation relies on the information contained in the four statistics derived from those studied by Wakeley and Hey (1997), calculated for multiple loci (see section "The summary statistics"). For each locus, `MIMAR` generates a set of genealogies under a model with those parameters. By default, `ngen=10` genealogies or ARGs are generated per locus for each step of the MCMC (this can be custom changed using the switch "`-x ngen`"). `MIMAR` then estimates the likelihood by calculating the probability of the data summaries at all the loci given the set of genealogies. Finally, `MIMAR` outputs a sample from the posterior distribution of the parameters given the data summaries using MCMC (Becquet and Przeworski, 2007).

In addition to the five demographic parameters, there are a number of locus-specific parameters. We assume that each locus follows the infinite sites mutation model (Kimura, 1969), then define an inheritance scalar x, which is equal to 1 for autosomal, $\frac{3}{4}$ for X-linked and $\frac{1}{4}$ for Y- and mtDNA-linked loci, reflecting copy number differences. To allow for mutation rate variation among loci with the same inheritance pattern, we introduce an additional scalar v for each locus. Given this parametrization, the locus-specific mutation rate in population 1 is given by $xvZ\theta_1$, where Z is the length of the locus in bp after filtering (see section "The input file"), and the locus-specific population mutation rates for other populations are defined analogously.

Moreover, the set of locus-specific population recombination rates, $(\rho_1, \cdots, \rho_Y)$, is a nuisance parameter referred as $\mathbf{P}$. The population recombination rate per bp is defined as $\rho = 4N_1c$, where c is the per bp per generation recombination rate. We ignore gene conversion, treating all recombination as crossovers alone. We also define an inheritance and rate variation scalar for recombination, w, which is usually set to ω, the ratio of the locus-specific population recombination rate per bp over ρ (i.e., $\omega = 0$ for the mtDNA- and Y-, $\frac{1}{2}$ for X-linked loci and 1 ($\frac{1}{2}$) for autosomal loci in mammals (in *Drosophila*)). If sites were filtered (i.e., $Z_r > Z$, where Z_r was the initial locus length, see Figure 3), w should be set to $\omega\frac{Z_r-1}{Z-1}$. There are various options to specify the population recombination rate (see section "Crossing over").

- Either ρ is fixed across loci (with the switch "`-r` ρ"), such that `MIMAR` obtains the locus-specific population recombination rate with $w\rho(Z-1) = \omega 4N_1c(Z_r-1)$ (Note that in all cases, `MIMAR` obtains the same locus-specific population recombination rate, i.e., $\omega 4N_1c(Z_r-1)$).

- Alternatively, if an estimate of the locus-specific population recombination rate per bp is available for each locus from linkage disequilibrium analysis, $\widehat{\rho_\circ} = 4\widehat{N_1c}$, the scalar w can be set to $\omega\widehat{\rho_\circ}\frac{Z_r-1}{Z-1}$ to incorporate this knowledge in the estimation (with the switch "`-r` 1"). Note that the product $\omega\widehat{\rho_\circ}$ is the scaled sex-averaged locus-specific population recombination rate per bp, i.e., for an X-linked locus, c is the female recombination rate and $\omega = \frac{1}{2}$ so that $\omega\widehat{\rho_\circ} = 2\widehat{N_1 c}$. In this case, MIMAR obtains the locus-specific population recombination rate with $w(Z-1)$.

- Similarly, if an estimate of the recombination rate per bp is available for each locus from pedigree analysis, $\widehat{c}$, the scalar w can be set to $\omega\widehat{c}\frac{Z_r-1}{Z-1}$ to incorporate this knowledge in the estimation (with the switch “-r 2”). Note that the product $\omega\widehat{c}$ is the scaled sex-averaged locus-specific recombination rate per bp, i.e., for an X-linked locus, $\widehat{c}$ is the estimate of the female recombination rate so the scaled sex-averaged recombination rate is $\frac{1}{2}\widehat{c} = \omega\widehat{c}$. In this case, MIMAR obtains the locus-specific population recombination rate with $w\frac{\theta_1}{\mu}(Z-1)$.

- Finally, the recombination rate for each locus can be a nuisance parameter, in which case, the ratio $r = \frac{c}{\mu}$ can be drawn from either an exponential distribution prior with mean $\frac{1}{\lambda}$ (with the switch “-r e λ”) or a normal distribution prior with mean υ and standard deviation σ (with the switch “-r n ν σ”). In these cases, MIMAR obtains the locus-specific population recombination rate with $rw\theta_1(Z-1)$.

The summary statistics

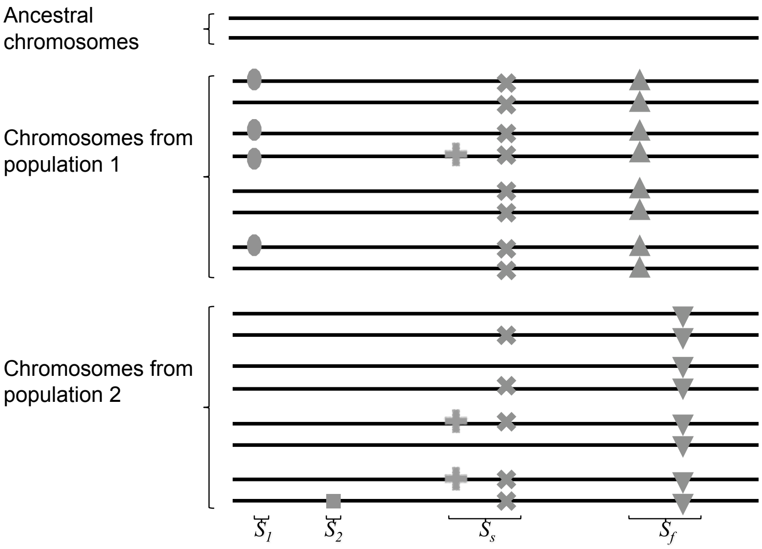


Figure 2: Examples of polymorphisms for each statistics. The black lines represent the ancestral alleles along autosomal chromosomes found in four individuals in two populations and an outgroup individual. The grey symbols represent derived alleles found in the sample: the polymorphism with the oval mutation will count in S_1 ($f_1 = 0.5$ and $f_2 = 0$), the site with the square mutation will count in S_2 ($f_1 = 0$ and $f_2 = 0.125$), the sites with the plus and cross mutations will count in S_s ($f_1 = 0.125$ and $f_2 = 0.25$ (plus) and $f_1 = 1$ and $f_2 = 0.625$ (cross)) and the sites with the triangle mutations will count in S_f ($f_1 = 1$ and $f_2 = 0$ (upward triangle) and $f_1 = 0$ and $f_2 = 1$ (downward triangle)).

MIMAR requires summaries of the polymorphism data at each locus as input. To summarize the data, we use statistics derived from those of Wakeley and Hey (1997) for this type of inference problem assuming known ancestral alleles: for each locus, we consider the number of derived polymorphisms unique to the samples from populations 1 and 2 (S_1 and S_2 respectively), the number of shared derived alleles between the two samples (S_s), and the number of fixed polymorphisms in either sample (S_f). Therefore, the data used for estimation are defined by $\mathbf{D} = (\mathbf{D_1}, \cdots, \mathbf{D_Y})$, the set of statistics for the Y loci, in which $\mathbf{D_y}$ is the vector of summaries for locus y, S_k, $k \in \{1,\ 2,\ s,\ f\}$.

The statistics are calculated as follows: First, we assume that an outgroup sequence is available and can be used to determine which allele is derived without error. In practice, it may be advisable to use two outgroup sequences to minimize error in inferring the ancestral state. Each polymorphic site is assigned to

one of the statistics depending of its frequency of the *derived* allele in the population i, f_i. Again, to calculate the derived allele frequencies, the ancestral allele needs to be known.
Specifically, if $0 < f_i \leq 1$ in each population sample, the derived allele is shared, if $f_i = 0$, $f_j = 1$, $i \neq j$, the allele is fixed in the sample j, and if $f_i = 0$ and $f_j < 1$, $i \neq j$, the derived allele is polymorphic specifically in sample j. For example, a polymorphic site is unique to sample 1 if the frequencies of the derived allele are $f_1 = 0.5$ and $f_2 = 0$. Similarly, a polymorphic site is shared if the frequencies of the derived allele are $f_1 = 0.125$ and $f_2 = 0.25$ or $f_1 = 1$ and $f_2 = 0.625$ (See Figure 2).

The input file

The user needs to provide the following information for each locus. For locus y:

- The locus name, `Name`$_y$. Any string of up to 50 characters.
- The length of the locus in base pairs, Z_y.
  - Z_y corresponds to the number base pairs in the locus after filtering out indels, sites with missing data, ambiguous ancestral alleles and/or more that two alleles. For recombining loci, one needs to incorporate the ratio of possibly recombining sites (i.e., $Z_{r_y} - 1$, where Z_{r_y} is the initial length of the locus) over $Z_y - 1$ into the recombination scalar w_y to avoid biases.
  - Note that it is possible to incorporate the information from sites with more than one alleles (as long as the ancestral allele is unambiguous) by considering each derived allele independently without changing the length Z_y (see Figure 3).

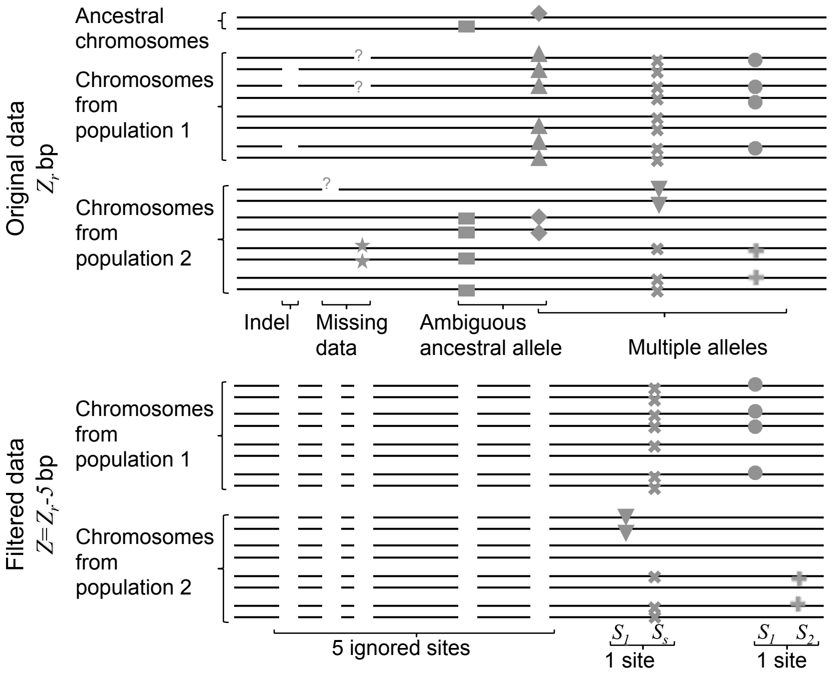


Figure 3: Example of filtering before calculating the summary statistics of polymorphism. See legend of Figure 2. The upper panel shows the data of a locus of size Z_r bp before filtering with one indel, two sites with missing data, two sites with ambiguous ancestral alleles (one has three alleles) and two sites with more than two alleles in the sample. The lower panel show the locus after filtering out the indel and the sites with missing data and ambiguous ancestral alleles so that the number of possibly mutating sites is $Z = Z_r - 5$ bp. The two remaining sites with more than two alleles are separated into two polymorphic sites, each with one allele considered derived and the other as ancestral, although they count as a single site (i.e., no change to Z). These sites are assigned to the different summary statistics as shown in Figure 2.

- The inheritance scalar, x_y (i.e., 1 for autosomal loci, 0.75 for X- and 0.25 for Y- and mtDNA-linked loci).

- The mutation rate variation scalar, v_y (which can be estimated e.g., from divergence data: i.e., if the mutation rates across loci are not homogeneous, set v_y to the ratio of observed to expected divergence at each locus).

- The recombination inheritance and rate variation scalar, w_y.

  - w_y is usually set to ω_y, the ratio of the locus-specific population recombination rate per bp over ρ (i.e., $\omega_y = 0$ for the mtDNA and Y, 0.5 for X and 1 (0.5) for autosomes in mammals (in *Drosophila*)).
  - If sites were filtered out (i.e., $Z_{r_y} > Z_y$, see Figure 3) w_y should be set to $\omega_y \frac{Z_{r_y}-1}{Z_y-1}$.
  - If an estimate of the locus-specific population recombination rate per bp is available for each locus from linkage disequilibrium analysis, $\widehat{\rho_{\text{o}_y}} = \widehat{4N_1c}_y$, the scalar w_y can be set to $\omega_y\widehat{\rho_{\text{o}_y}}\frac{Z_{r_y}-1}{Z_y-1}$ to incorporate this knowledge in the estimation (with the switch "`-r` 1", see section "Crossing over" for details). Note that the product $\omega_y\widehat{\rho_{\text{o}_y}}$ is the scaled sex-averaged locus-specific population recombination rate per bp, i.e., for an X-linked locus c is the female recombination rate and $\omega_y = \frac{1}{2}$ so that $\omega_y\widehat{\rho_{\text{o}_y}} = 2\widehat{N_1c}$.
  - If an estimate of the locus-specific recombination rate per bp is available for each locus from pedigree analysis, $\widehat{c_y}$, the scalar w_y can be set to $\omega_y\widehat{c}\frac{Z_{r_y}-1}{Z_y-1}$ to incorporate this knowledge in the estimation (with the switch "`-r` 2", see section "Crossing over" for details). Note that the product $\omega_y\widehat{c_y}$ is the scaled sex-averaged locus-specific recombination rate per bp, i.e., for an X-linked locus, $\widehat{c_y}$ is the estimated female recombination rate so the scaled sex-averaged recombination rate is $\frac{1}{2}\widehat{c_y} = \omega_y\widehat{c_y}$.

- The sample size for the locus in population 1 and 2, n_{1_y} and n_{2_y}, respectively.

- The summary statistics of the polymorphism (see section "The summary statistics"): S_{1_y} S_{2_y} S_{s_y} S_{f_y}.

To provide this information, use the switch "`-lf input`", to specify the input file name containing the information as described below for Y loci:

$$\begin{array}{lllllllllll}
\texttt{Name}_1 & Z_1 & x_1 & v_1 & w_1 & n_{1_1} & n_{2_1} & S_{1_1} & S_{2_1} & S_{s_1} & S_{f_1} \\
\texttt{Name}_2 & Z_2 & x_2 & v_2 & w_2 & n_{1_2} & n_{2_2} & S_{1_2} & S_{2_2} & S_{s_2} & S_{f_2} \\
\ldots & & & & & & & & & & \\
\texttt{Name}_Y & Z_Y & x_Y & v_Y & w_Y & n_{1_Y} & n_{2_Y} & S_{1_Y} & S_{2_Y} & S_{s_Y} & S_{f_Y}
\end{array}$$

Information for a locus can be separated by a tab or space. Information for each locus needs to be in a single line. Before the loci information, any other information can be included, but it must end by "`//`" to specify that the program should ignore this text.

Below is an example for a four locus data set, in which `locus1_autosom` is autosomal and `locus2_Xlinked`, `locus3_Ylinked` and `locus4_mtDNA` are X- Y- or mtDNA-linked, respectively. The recombination scalar was set accordingly and I assumed that the mutation rate on the mtDNA was 2×10^{-7} (see the file "`inputmimar`"):

```
Name           length x_y  v_y w_y n_1 n_2 S_1 S_2 S_s S_f //
locus1_autosom 1000   1    1   1   10  10  14  1   11  0
locus2_Xlinked 1000   0.75 1   0.5 10  10  2   5   2   0
locus3_Ylinked 1000   0.25 1   0   10  10  2   2   0   0
locus4_mtDNA   1000   0.25 10  0   10  10  9   20  14  0
```

The basic command line

```
mimar nsteps bsteps Y -lf input -u μ -t θ1 -ej T -o soutput [options]
```

This line shows the simplest usage of MIMAR. There are three arguments followed by the parameters (introduced by switches, such as "-t"). The three arguments, nsteps, bsteps and Y, must appear in this order, while the switches can appear in any order.

nsteps The total number of steps (or minutes if the switch "-y t" is used) until the end of the MCMC run, i.e., it is the sum of the burning steps and the following MCMC steps.

bsteps The number of burnin steps.

Y The number of loci considered.

μ The generational mutation rate per bp.

$\theta_1 = 4N_1\mu$ The population mutation rate per bp for the first population. When the other population mutation rates are not specified, they are equal to θ_1.

T The split time in generations, at which, backward in time, all lineages in population 2 are moved to population 1.

input The input file name containing the information on the loci with their S statistics (see section "The input file").

output The name of the summary output file, which contains a sample from the posterior distribution of the parameters represented in 1000 bins (see section "The summary output file").

[options] A list of any options/switches described in the next sections.

The migration rate is zero by default. If the user provides a value for a parameter, it is fixed; otherwise, if a prior range is provided, the parameter is estimated (see section "Setting the range of the prior distributions"). The user needs to specify θ_1 and T because these two parameters are the minimum information required to built the simplest isolation-migration model, in which the two populations split T generations ago without subsequent gene flow, and in which the ancestral and descendant populations have the same population mutation rate, θ_1.

In the following basic command line MIMAR will analyze the data in the file "inputmimar" (see example in section "The input file"). It will print 10,000 MCMC steps after 1,000 steps of burnin in the standard output and the marginal posterior distributions in the summary output file named soutput (see section "The outputs of MIMAR's analysis"). Note that since in this example θ_1 and T are fixed, the 10,000 MCMC steps will all have the same parameter values: $\theta_1 = \theta_2 = \theta_A = 0.005$, $T = 10,000$ generations and $M = 0$:

```
mimar 11000 1000 4 -lf inputmimar -u 2e-8 -t .005 -ej 1e4 -o soutput
```

Providing the other parameters of the model

To provide information about the other parameters of the isolation-migration model, the user needs to use either of the following switches:

```
-n θ2 -N θA -M M
```

$\theta_2 = 4N_2\mu$ The population mutation rate per bp for the second population.

$\theta_A = 4N_A\mu$ The ancestral population mutation rate per bp.

$M = 4N_1m$ The expected number of migrants between the two populations each generation, where m is the symmetrical migration rate between the two populations. Note that M is defined in term of N_1 (see section "Spatial structure and migration:" in "`msdoc.pdf`" for further details). The estimate of the posterior distributions for the symmetrical rates of gene flow m can be obtained by calculating $m = M\frac{\mu}{\theta_1}$ for each recorded steps in the standard output file.

In the following example, `MIMAR` will proceed as before, but all the MCMC steps will have the parameter values: $\theta_1 = 0.005$, $\theta_2 = 0.003$, $\theta_A = 0.005$, $T = 10,000$ generations and $M = 0.7$. Note that, like in the previous example, the 10,000 lines in the standard output will have the same parameters values since all the parameters are fixed.

```
mimar 11000 1000 4 -lf inputmimar -u 2e-8 -t .005 -n .003 -ej 1e4 -N .005 -M .7 -o soutput
```

Setting the range of the prior distributions

Any or all of the parameters in $\mathbf{\Theta} = (\theta_1, \theta_2, \theta_A, T, M, \mathbf{P})$ can be estimated (with the exception of the recombination rates in $\mathbf{P}$. Because $\mathbf{P}$ is a nuisance parameter not estimated, its values are either fixed (when ρ is fixed), or drawn from the distribution described above, see section "Crossing over"). For each parameter to be estimated, the user needs to provide the bounded support of the uniform prior distribution (to estimate M, the uniform prior is on $\ln(M)$). The ranges are given as follows:

`-t  u` a b : draws θ_1 from Uniform$[a, b]$.

`-n  u` a b : draws θ_2 from Uniform$[a, b]$.

`-ej u` a b : draws T from Uniform$[a, b]$.

`-N  u` a b : draws θ_A from Uniform$[a, b]$.

`-M  l` a b : draws $\ln(M)$ from Uniform$[a, b]$.

In the following example, the parameters θ_1, T and M will be estimated (see the files "`exsoutput`" and "`outputmimar`"). The prior range for θ_1 is Uniform$[0.001, 0.01]$, the prior range for T is Uniform$[0, 100000]$ generations, the expected number of migrants, M, will range between $[0.135, 7.389]$. Since θ_2 is not provided it will be equal to θ_1 at every step, while θ_A is fixed to 0.005.

```
 mimar 11000 1000 4 -lf inputmimar -u 2e-8 -t u .001 .01 -ej u 0 1e5 -N .005 -M l -2 2 -o
                                     exsoutput >outputmimar
```

My advice: To allow better mixing, I typically set the variance of the order of $\frac{1}{40}$, $\frac{1}{50}$ of the width of the prior for each estimated parameter using the switch "`-v`" (see section "Change the variances for the normal kernel distributions").

The outputs of `MIMAR`'s analysis

The standard output

The first twelve lines of an example of the standard output are reported below (see the file "`outputmimar`"):

```
Analysis of file inputmimar

Prior distributions ranges/parameter values (-1 means that no information was
provided. If both values are the same, the parameter was fixed by the user)
theta1 0.001 0.01
theta2 0.001 0.01
Tcoal 0 2
Tgen 0 100000
thetaA 0.005 0.005
M12 -2 2
M21 0 0

Step # theta1 theta2 Tcoal Tgen thetaA M12 M21 L
1001 0.00512989 0.00512989 0.0279893 7179.1 0.005 5.96188 5.96188 8.52405e-14
...
```

The first twelve lines of `MIMAR`'s standard output report the name of input file analyzed, the list of parameter values or ranges of the prior distributions and the header for the values reported for each accepted MCMC step (here, only the first line of $10,000$ sets of parameters after the burnin is shown). Each line after the header records an accepted set 0f parameters in the MCMC (the switch “`-i int`” defines how often sets of parameters are recorded, by default `int`=1). In order:

1. The step number followed by the accepted set of parameters for this MCMC step:
2. θ_1
3. θ_2
4. $t = \frac{T}{4N_1}$ in coalescent unit
5. T in generations
6. θ_A
7. $M_{12} = 4N_1m_{12}$
8. $M_{21} = 4N_1m_{21}$
9. $p(\mathbf{D}|\mathbf{\Theta})$, the likelihood of the data summaries for all loci given the parameters and the set of ARGs generated for this step.

`MIMAR` provides t in coalescent units so that the user can estimate $T = \theta_1 \frac{t}{\hat{\mu}}$ using a different estimate of μ. When one estimates a symmetrical migration rate as in this case, M, the same value is indicated in both M_{12} and M_{21} columns.

I provide the script `"MIMARplot.R"`, which reads the results from the standard output file (by default named "outputmimar") and allows the user to display the marginal posterior distributions and summaries of these distributions (i.e., modes and 90^{th} percentiles) for different numbers of burnin steps and bins in `R`.

The summary output file

The first eight lines of an example of summary output file is reported below (see the file `"exsoutput"`):

```
./mimar 11000 1000 4 -lf inputmimar -u 2e-8 -t u .001 .01 -ej u 0 1e5 -N .005 -M l
-2 2 -o exsoutput
3579 27011 59243

# generated steps. Accepted H>1. Accepted H<1. Acceptance rate. Rejected H<1.
Rejected L==0. Rejected outside prior. cpumin. cpusec.
11000 313 319 0.0574545 3873 0 6495 0 5.4308

theta1 theta2 Tcoal Tgen thetaA M12 M21
0.001004504504505 0 0.001004504504505 0 0.001001001001001 0 50.050050050050054 0
0.002000202204206 0 0.135606495866729 0
...
```

The first line is the command line. The second line lists the random number seeds (see section "Downloading and compiling"). The next two informative lines report nine values providing information about the analysis to help you ensure that the burnin was large enough and that the MCMC is mixing well and converged (see section "How long to run the program and how to improve the MCMC?" for details on how to use this information):

`Value1` The total number of steps generated (i.e., `value1` + `value2` + `value5` + `value6` + `value7`). If the is the final summary output file the number is also equal to `nsteps` + `bsteps`.

`Value2` The number of steps accepted with the Hasting ratio $h \geq 1$.

`Value3` The number of steps accepted with the Hasting ratio $h < 1$.

`Value4` The acceptance rate (i.e., $\frac{\texttt{Value2+Value3}}{\texttt{Value1}} = \frac{\texttt{Value1}-(\texttt{Value5+vValue6+Value7})}{\texttt{Value1}}$).

`Value5` The number of steps rejected with the Hasting ratio $h < 1$.

`Value6` The number of steps rejected because the estimate of the likelihood was 0 for at least one of the locus.

`Value7` The number of steps rejected because one of the parameter value was outside the prior range.

`Value8` The minutes of running time.

`Value9` The remaining seconds of running time.

Then come the samples from the marginal posterior distributions of the parameters summarized into 1000 bin histograms (only the first line is shown in the example here). For each bin, the mid-value of the bin and the posterior density is indicated (for this example, all the first bins of the histograms have zero posterior likelihood). The estimate of the marginal posterior distribution of the symmetrical migration rate, M, is shown under M_{12}, (M_{21} is empty). For each bin the values correspond to:

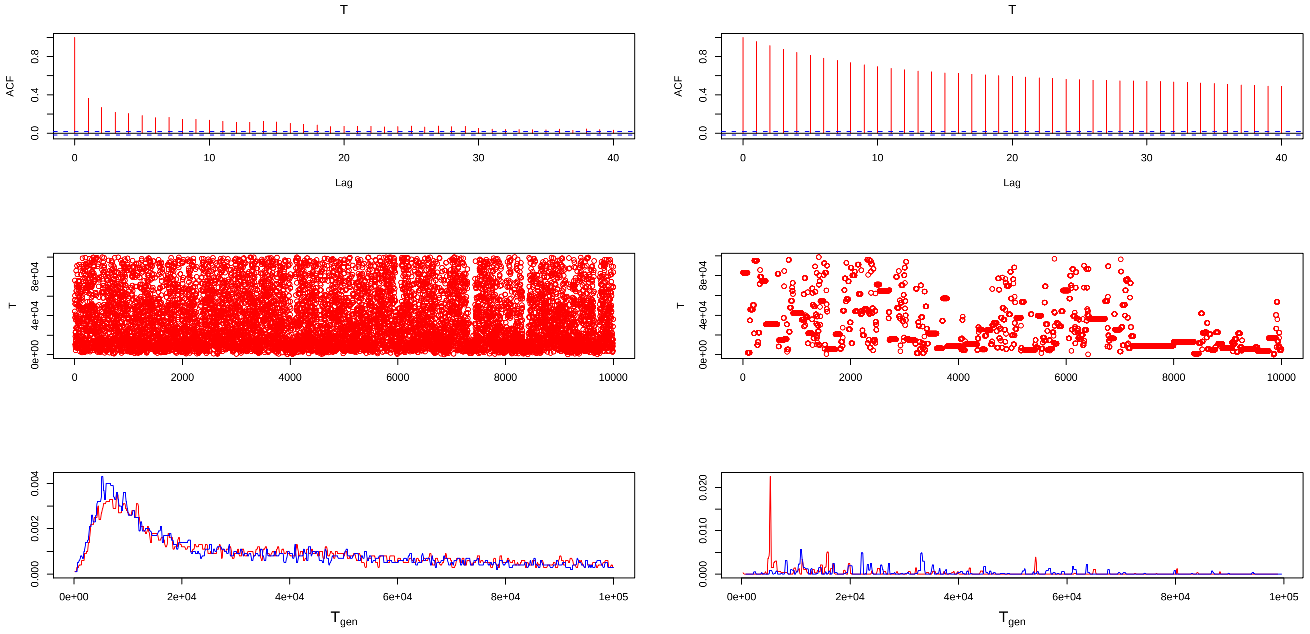
1. θ_1 2. $p(\theta_1|\mathbf{D})$ 3. θ_2 4. $p(\theta_2|\mathbf{D})$ 5. t 6. $p(t|\mathbf{D})$ 7. T 8. $p(T|\mathbf{D})$
9. θ_A 10. $p(\theta_A|\mathbf{D})$ 11. M_{12} 12. $p(M_{12}|\mathbf{D})$ 13. M_{21} 14. $p(M_{21}|\mathbf{D})$

Where $p(\mathbf{D}|\mathbf{\Theta_i})$ is the posterior probability of the parameter values in the bin given the data summaries for all loci, $\Theta_i \in \{\theta_1, \theta_2, \theta_A, T, M\}$. The eight last lines of the summary output file provide the point estimates (means, modes, medians), variances and 90% central credible intervals of the marginal posterior distributions for the parameters of the model:

```
...
parameters mean mode var per.05 perc.5 perc.95
Theta1 0.00243583 0.00148198 1.23143e-06 0.00119369 0.0019955 0.00507658
Theta2 0.00243583 0.00148198 1.23143e-06 0.00119369 0.0019955 0.00507658
Tcoal=Tgen/4N1 0.427456 0.113113 0.131365 0.023023 0.293293 1.28629
Tgen 40770.1 7957.96 7.62065e+08 3553.55 34584.6 96146.1
ThetaA 0.005 0.00500025 0 0.00500025 0.00500025 0.00500025
M12 2.44724 0.878097 3.27728 0.470482 1.81122 6.67255
M21
```

How long to run the program and how to improve the MCMC?

To have confidence in MIMAR's estimation, the user needs to make sure that the burnin is long enough, and that the Markov chain converged to the "right" posterior distribution. This is important, because if the burnin is too small and/or if the chain did not reach convergence, the plots of the posterior distributions for the estimated parameters may look nice, but they could be completely wrong.



(a)

(b)

Figure 4: Example of properly mixing (a, results from the standard output files "outputmimar1" and "outputmimar2") and badly mixing MCMC (b, results from the standard output files "outputmimar" and "outputmimar0"). The top panels are graphs using the acf() function, the middle panels plot the parameter value along the run and the bottom panels plot the posterior distributions of T for two different seeds and 1000 bins. Note that graph a) can be improved (see text).

- **Burnin.** As for all MCMC approaches, the first steps needs to be ignored because they are dependent on the starting values. In the original paper, we typically used burnin of the order of 1×10^5 steps, but for small data sets, the burnin could be shorter.
- **Convergence.** To assess convergence, I typically ran MIMAR for the same input file with two different "seedmimar" files, and checked whether the two analyses yielded the same posterior distributions. I provided the file "MIMARchart.xls" to help the user output the posterior distributions from the summary output files of two different seeds. Unfortunately MIMAR tends to converge quite slowly, and at least two runs and long run times are required. Part of the reason is that there are many parameters and the state space is large. Another reason is that ngen genealogies (and in case with recombination,

ngen ARGs corresponds even more genealogies) need to be generated for each locus at each MCMC step in order to obtain a good estimate of the likelihood given the summary statistics; thus, the larger ngen and the data sets analyzed, the slower the analysis.

- **Mixing.** Because the speed of the program, and how well a chain converges, depend on how well the Markov chain explores the space of parameter values (a process called "mixing"), another approach to asses proper convergence is to monitor the mixing over the course of a run.

  This can be done by using `acf()`, the autocorrelation function of `R`, or by plotting the parameter values over time from the columns in the standard output (see Figure 4. I provided the file `"MIMARconv.R"` to generate those graphs). If the parameter value plots show trends over long portions of the run, or if autocorrelations persist for a large number of steps (see Figure 4b), this means that the state space is being explored slowly, in which case longer runs are required. The mixing can also be monitored using the values in the fifth line of the summary output file (Noted `Valuex` below, $\mathtt{x} \in [4, 7]$, cf. previous section). A good indicator of poor mixing is when the acceptance rate is low (i.e., value number 4. in the fifth line $\ll .01$). There are many ways to increase the acceptance rate and improve the mixing:

  - Increase the burnin (`bsteps`).
  - If a majority of sets of parameters are rejected because a parameter value was outside its prior range (`Value7`), this indicates that the variances may be too large and/or that the prior distributions for one or more estimated parameters may be too narrow:
    - If the marginal posterior distribution of a parameter show a high likelihood for values on the edge of the prior (i.e., large tail) this may indicates that the prior distribution for this parameters is too narrow. If so, widened the prior distribution for this parameter.
    - You can decrease the variances of the kernel distribution of the estimated parameters (using the switch "`-v`" - see my advice in section "Change the variances for the normal kernel distributions").
  - If a majority of sets of parameters are rejected because the estimated likelihood is 0 for at least one of the loci (`Value6`), it indicates that `MIMAR` may not generate enough genealogies are generated per locus (i.e., the likelihood is not estimated reliably) and/or that the prior distributions for one or more estimated parameters may be too width.
    - If the marginal posterior distribution of a parameter fits well within its prior range, you can reduce the range of the prior distribution for this parameter.
    - You can increase the number of genealogies used by `MIMAR` to estimate the likelihood at each locus using the switch "`-x ngen`". However, the trade-off of increasing `ngen` is that it slows down the analysis, since the process that generates genealogy is time consuming (see section "Change the number of genealogies generated per locus").
  - When the MCMC is not mixing well as indicated by highly correlated consecutive sets of parameters from the graphs output by `"MIMARconv.R"` and/or by the majority of rejected sets of parameters having a Hasting ratio $h < 1$ (`Value5`) and/or by spiky marginal posterior distributions (Figure 4b):
    - This indicates that the variances of the kernel distribution of the estimated parameters may be too small, which can be changed using the switch "`-v`".
    - If the standard output contains $\gg 10000$ recorded steps you may choose to reduce it size, with the advantage of reducing the autocorrelation between steps by using the switch "`-i int`" to specify a larger interval between accepted parameters recorded in the standard output (see section "Reduce the standard output size and the correlation between sets of parameters").

My advice: Getting a good mixing might require experimenting and may not be perfect (i.e., the acceptance rate may still be small).

- To monitor the analysis and its mixing and convergence over time (and interrupt `MIMAR` if things are not looking good), use the file `"mimarrun"` and/or the switch "`-L osteps`" to output a summary output file every `osteps` steps (or minutes if the switch "`-y t`" is used, see section "Monitor mixing and convergence during `MIMAR`'s run"). Consider the values in the fifth line and use the file `MIMARchart.xls` on those intermediate results.

- To assess convergence during the run (or at the end of the run and to estimate parameters), I also provided the script `"MIMARplot.R"` (provided). This `R` script reads the results from the standard output files from two different seeds (by default named `"outputmimar1"` and `"outputmimar2"`) and allows the user to print the marginal posterior distributions as well as a table with the modes and 90% confidence intervals for the five parameters of the model with different numbers of burnin steps and bins (see Figure 4, lower panel). I typically used this kind of script to generate posterior distributions, assess convergence and obtain the point estimates for my publications (Becquet and Przeworski, 2007, 2009). Note that if you want to use this script during the run, copy the standard output files for tow different seeds in a different directory (i.e., not where `MIMAR` is printing it's outputs), edit these new files and remove the last incomplete line, save, then use `"MIMARplot.R"`.

Change the variances for the normal kernel distributions

`mimar nsteps bsteps` Y `-lf input -u` μ `-t` θ_1 `-ej` T *`-v`* $\boldsymbol{V_{\theta_1}}$ $\boldsymbol{V_{\theta_2} V_T}$ $\boldsymbol{V_{\theta_A}}$ $\boldsymbol{V_{M_{12}}}$ $\boldsymbol{V_{M_{21}}}$ `-o soutput`

After the initial step, `MIMAR` proposes the new value of a parameter from a normal distribution with mean the previous value and the variance specific to the parameter. Note the variance is for the normal distribution to choose $\ln(M)$. After exploratory simulations, I chose the variances that maximized the acceptance rate for the data I used (Gilks et al., 1996; Becquet and Przeworski, 2007). By default, $V_{\theta_1} = V_{\theta_2} = V_{\theta_a} = 2 \times 10^{-4}$, $V_T = 8 \times 10^4$ and $V_{M_{12}} = V_{M_{12}} = .25$. For certain data sets, these value may not be ideal, resulting in poor mixing (i.e., a small acceptance rate (`Value4`), and specifically large `Value5` or `Value7`, cf. above). The user can change the values of the variances using the "`-v`" switch, followed by the list of six variances for each parameters (the value can be 0 for the parameters that are not estimated, but six values need to be specified). It is the user's responsibility to provide a reasonable variance for a parameter; but as a guide, it should be smaller (possibly by orders of magnitude) than the width of the prior distribution. As an example, the following command line sets the variances to: $V_{\theta_1} = 2.5 \times 10^{-4}$, $V_T = 2 \times 10^4$, $V_{M_{12}} = V_{M_{12}} = 0.1$ and $V_{\theta_2} = V_{\theta_a} = 0$:

```
mimar 11000 1000 4 -lf inputmimar -u 2e-8 -t u .001 .01 -ej u 0 1e5 -N .005 -M l -2 2 -v
2.5e-4 0 2e4 0 .1 0 -o soutput
```

My advice: I observed that I had a better mixing when I set the variances of the order of $\frac{1}{40}$ -$\frac{1}{50}$ the width of the prior for each estimated parameter.

Change the number of genealogies generated per locus

`mimar nsteps bsteps` Y `-lf input -u` μ `-t` θ_1 `-ej` T ***`-x ngen`*** `-o soutput`

By default, `MIMAR` generates `ngen=10` genealogies (or ARGs for recombining loci) for each locus and each steps of the MCMC. This provides a more reliable estimate of the likelihood of the data summaries given a set of parameters than if only one genealogy were sampled, which, in turn, leads to reasonable acceptance rates of the MCMC (see Methods in Becquet and Przeworski, 2007). If the acceptance rate is low (`Value4`), it may be an indication that the likelihood is not estimated reliably (i.e., specifically with high `Value6`, cf. above). One way to increase the acceptance rate is by generating more genealogies to estimate the likelihood. However, the trade-off of increasing `ngen` is that it slows the analysis, since the process that generates genealogy is time consuming. Using the following command line, for example, MIMAR will generate 100 genealogies per locus:

```
mimar 11000 1000 4 -lf inputmimar -u 2e-8 -t .005 -ej 1e4 -x 100 -o soutput
```

My advice: Note that when your model incorporate recombination you should reduce this number: as one recombination is equivalent to generate two gene genealogies for a non-recombining locus if ngen=10 MIMAR generates 20 genealogies which is twice as long. Typically is $\rho \sim \theta_1$ set ngen on the order of $2-5$, if $\rho > \theta_1$ set ngen$=$1 may be enough.

Reduce the standard output size and the correlation between sets of parameters

mimar nsteps bsteps Y -lf input -u μ -t θ_1 -ej T *-i int* -o soutput

If nsteps is large, the standard output may become large quickly, because by default all accepted parameters are recorded (i.e., int=1). In addition, when the MCMC is mixing poorly (i.e., low Value4 and high Value5, cf. above) and many accepted parameters are recorded, consecutive sets of parameters are likely highly correlated. To avoid these inconvenience, use the switch "-i int" to specify the interval between accepted parameters recorded in the standard output. With the following command line, MIMAR will output the accepted parameters every 10 steps (i.e., the first two lines of standard output will be for steps number 1001 and 1011).

```
mimar 11000 1000 4 -lf inputmimar -u 2e-8 -t .005 -ej 1e4 -i 10 -o soutput
```

My advice: To generate decent posterior distributions you need only $1000-10000$ sets of parameters so set int of the order of e.g., $\frac{\texttt{nsteps}-\texttt{bsteps}}{1000}$.

Monitor mixing and convergence during MIMAR's run

In order to monitor whether MIMAR is mixing well during a run, the users can generate summary output files at intervals to obtain the acceptance rate and other useful information in two ways.

Generate summary output files on demand with the file "mimarrun"

When you want to monitor MIMAR, edit the file "mimarrun" write "yes" in the first line and save/copy "mimarrun" in the running directory. MIMAR will write the summary output into mimarrun, which will now starts with "no". Repeat these steps whenever you want to monitor MIMAR again.

Generate summary output files at regular intervals

mimar nsteps bsteps Y -lf input -u μ -t θ_1 -ej T *-L osteps* -o soutput

osteps is the number of steps separating intermediate summary output files. Each file will be named "soutput-x", where x=1 when 1×osteps steps are reached, 2 when 2×osteps steps are reached... If osteps < bsteps, the files generated within the burnin period will contain empty histograms. The following command line will generate five summary output files, at 2000, 4000, 6000, 8000, and 10000 steps, named "exsoutput-1", "exsoutput-2", "exsoutput-3", "exsoutput-4" and "exsoutput-5", respectively.

```
mimar 11000 1000 4 -lf inputmimar -u 2e-8 -t .005 -ej 1e4 -L 2000 -o soutput
```

Note that if the switch -y t is used, osteps is defined in minutes. The following command line will generate 24 summary output files, one every hour:

```
mimar 1440 1000 4 -lf inputmimar -u 2e-8 -t .005 -ej 1e4 -y t -L 60 -o soutput
```

My advice: Do not set `osteps` $< \frac{\texttt{nsteps}}{50}$, as the directory may become very large indeed.

More complex models

Crossing over

Fixed $\boldsymbol{\rho}$ across loci.

```
mimar nsteps bsteps Y -lf input -u μ -t θ1 -ej T -r ρ  -o soutput
```

To include crossing-over in the model, use the switch "`-r` ρ " to specify the population cross-over rate parameter, $\rho = 4N_1c$, where c is the probability of cross-over per generation per bp. For each locus y with recombination scalar $w_y = \omega_y \frac{Z_{r_y}-1}{Z_y-1} > 0$ (specified in the input file, see section "The input file"), `MIMAR` will generate genealogies under a coalescent model with recombination (Hudson, 1983). Z_{r_y} and Z_y are the initial length and the length after filter for locus y, respectively and ω_y is the ratio of the locus-specific population recombination rate per bp over ρ (i.e., $\omega_y = 0$ for the mtDNA and Y, $\frac{1}{2}$ for X and 1 ($\frac{1}{2}$) for autosomes in mammals (in *Drosophila*)). `MIMAR` obtains the locus-specific population recombination rate with $\rho_y = \omega_y 4N_1c(Z_{r_y} - 1)$ in all the options with the switch "`-r`". For example, for the following command line, for `locus1_autosom` and `locus2_Xlinked` in `"inputmimar"` (see section "The input file" and the file `"inputmimar"`), `MIMAR` calculate the locus-specific population recombination rates with $1 \times 999 \times 0.005 \sim 5$ and $0.5 \times 999 \times 0.005 \sim 2.5$, respectively:

```
mimar 11000 1000 4 -lf inputmimar -u 2e-8 -t .005 -ej 1e4 -r .005 -o soutput
```

Fixed locus-specific $\boldsymbol{\rho}$.

The user can fix the locus-specific recombination rates if estimates are available as follows:

- "`-r 1`": If an estimate of the locus-specific population recombination rate per bp, $\widehat{\rho_{\circ_y}} = \widehat{4N_1c_y}$, is available for locus y from linkage disequilibrium analysis, set the recombination scalar in the input file to $w_y = \omega_y \widehat{\rho_{\circ_y}} \frac{Z_{r_y}-1}{Z_y-1}$. To specify that estimates of the population recombination rates are known, use the switch "`-r 1`". In this case, `MIMAR` obtains the population recombination rate at a locus with $w_y(Z_y-1)$. For example, if you know that the population recombination rate per bp for `locus1_autosom` is 0.005 and for `locus2_Xlinked` is 0.008, then the recombination scalars will be $w_1 = 1 \times 0.005$ and $w_2 = \frac{1}{2} \times 0.008 = 0.004$ (see section "The input file" and the file `"inputmimar_4Nc"`). For the following command line, `MIMAR` will calculate the locus-specific population recombination rates with $999 \times 0.005 \sim 5$ and $999 \times 0.004 \sim 4$ for the first two loci of the input file `"inputmimar_4Nc"`, respectively:

  ```
  mimar 11000 1000 4 -lf inputmimar_4Nc -u 2e-8 -t .005 -ej 1e4 -r 1 -o soutput
  ```

  ATTENTION: Make sure that the product $\omega_y \widehat{\rho_{\circ_y}}$ is the scaled sex-averaged locus-specific population recombination rate per bp, i.e., for an X-linked locus, c is the female recombination rate and $\omega_y = \frac{1}{2}$ so that $\omega_y \widehat{\rho_{\circ_y}} = 2\widehat{N_1c_y}$.

- "`-r 2`": If an estimate of the locus-specific recombination rate per bp, $\widehat{c_y}$, is available for locus y from pedigree analysis, set the recombination scalar in the input file to $w_y = \omega_y \widehat{c_y} \frac{Z_{r_y}-)}{Z_y-1}$. To specify that estimates of the recombination rates are known, use the switch "`-r 2`". In this case, `MIMAR` obtains the population recombination rate at a locus with $w_y \frac{\theta_1}{\mu}(Z_y - 1)$. For example, if you know that recombination rate per bp for `locus1_autosom` is 1×10^{-8} and for `locus2_Xlinked` is 4×10^{-8}, then the recombination scalars will be $w_1 = 1 \times 10^{-8}$ and $w_2 = \frac{1}{2}4 \times 10^{-8} = 2 \times 10^{-8}$ (see section "The input file" and the file "inputmimar_c"). For the following command line, MIMAR will calculate the locus-specific population recombination rates with $999\frac{\theta_1}{\mu} \times 10^{-8} \sim 2.5$ and $999\frac{\theta_1}{\mu}2 \times 10^{-8} \sim 5$ for the first two loci of the input file "inputmimar_4Nc", respectively:

```
mimar 11000 1000 4 -lf inputmimar_c -u 2e-8 -t .005 -ej 1e4 -r 2 -o soutput
```

ATTENTION: Make sure that the product $\omega_y\widehat{c_y}$ is the scaled sex-averaged locus-specific recombination rate per bp, i.e., for an X-linked locus, $\widehat{c_y}$ is the estimate of the female recombination rate so the scaled sex-averaged rate is $\frac{1}{2}\widehat{c_y} = \omega_y\widehat{c_y}$.

Variable locus-specific $\boldsymbol{\rho}$.

The recombination rate can be allowed to vary across loci and across steps. In this case, the user needs to use the switches

- "***-r e*** $\boldsymbol{\lambda}$": the ratio $r = \frac{c}{\mu}$ is drawn from an exponential distribution prior with mean $\frac{1}{\lambda}$ for each recombining locus. In the following example, $\mathrm{E}(r) = 0.6$:

```
mimar 11000 1000 4 -lf inputmimar -u 2e-8 -t .005 -ej 1e4 -r e 1.667 -o soutput
```

- "***-r n*** $\boldsymbol{\nu}$ $\boldsymbol{\sigma}$": the ratio $r = \frac{c}{\mu}$ is drawn from a normal distribution prior with mean υ and standard deviation σ for each recombining locus. In the following example, r is chosen from Normal$(1, 0.1)$:

```
mimar 11000 1000 4 -lf inputmimar -u 2e-8 -t .005 -ej 1e4 -r n 1 .01 -o soutput
```

Note that in these cases, the recombination rates for the loci are nuisance parameters, and are chosen independently across recombining loci and across steps. MIMAR obtains the population recombination rate at a locus with $r_y w_y \theta_1 (Z_y - 1)$.

Asymmetrical migration rates

mimar nsteps bsteps Y -lf input -u μ -t θ_1 -ej T *-m* ***i j*** $\boldsymbol{M_{ij}}$ -o soutput

In all the studies we published to date, we only considered symmetrical migration rates (Becquet and Przeworski, 2007, 2009). However, there is also an option to fix or estimate asymmetrical migration rates. Thinking forward in time, the expected number of individuals that migrate from population i into population j each generation is $M_{ij} = 4N_1 m_{ij}$, i and $j \in [1, 2]$, $i \neq j$, where m_{ij}, is the fraction of population j that is made up of migrant from population i every generation. Note that M_{ij} is defined in term of N_1 (see section "Spatial structure and migration:" in "msdoc.pdf" for further details) .

To fix M_{ij}, simply add "-m *i j* M_{ij}" to the command line. Alternatively, you can estimate M_{ij} by writing "-m *i j* 1 *a b*", in which case $\ln(M_{ij})$ is drawn from Uniform$[a, b]$. The estimate of the posterior distributions for the asymmetrical rates of gene flow m_{ij} can be obtained by calculating $m_{ij} = M_{ij}\frac{\mu}{\theta_1}$ for each recorded steps in the standard output file.

The following command line will generate genealogies for the isolation-migration model with randomly sampled number of migrants per generation from population 2 into population 1 and from population 1 into population 2 (thus estimating M_{12} and M_{21} simultaneously).

```
mimar 11000 1000 4 -lf inputmimar -u 2e-8 -t .005 -ej 1e4 -m 1 2 1 -2 2 -m 2 1 1 -2 2 -o
                                  soutput
```

Other options conserved from ms. Use at your own peril!

I have not tested these additional options so do not guarantee that they will work properly.

Crossing-over and gene conversion:

`mimar nsteps bsteps` Y `-lf input u` μ `-t` θ_1 `-ej` T ***`-r`*** $\boldsymbol{\rho}$ `-c` f $\boldsymbol{\lambda}$ `-o soutput`

See `"msdoc.pdf"` for details.

Exponentially growing or shrinking population size

`mimar nsteps bsteps` Y `-lf input -u` μ `-t` θ_1 `-ej` T ***`-G`*** $\boldsymbol{\alpha}$ `-o soutput`

See `"msdoc.pdf"` for details.

To set individual populations to have different growth rates, the "`-g` i α_i" command is used to set the growth rate of population i to α_i. See `"msdoc.pdf"` for details.

Past demographic events

It is the users responsibility to provide sample configurations, migration rates and past demographic events for which the sampled chromosomes will eventually coalesce. Note also that the program, as is, can not analyze data sets for more than two populations at a time.

To specify that demographic parameters change at specific times in the past, the "`-e`" switches are used. These switches are: "`-eG`", "`-eg`", "`-eb`" (was initially "`-eN`" in `ms`), "`-en`", "`-eM`", "`-em`". In each case, the first parameter following the switch is τ, the ratio of the time of the event divided by the split time of the isolation-migration model. Thus, the time of the event is τT in generations. The arguments following the time parameter specify populations and other relevant parameters, as indicated in the following list:

`-eG` τ α	Set all growth rates to α at time τT generations.
`-eg` τ i α_i	Set growth rate of population i to α_i at time τT generations.
`-eb` τ x	Set all population mutation rates to $x\theta_1$ at time τT generations.
`-en` τ i x	Set population i mutation rate to $x\theta_1$ at time τT generations.
`-eM` τ x	Set the symmetrical migration rate to x at time τT generations.
`-em` τ i j x	Set $4N_1m_{ij}$ to x at time τT generations.

The following example specifies the first population mutation rate to one-tenth of its current value (in this case θ_1) between $8000 < t < 10000$ generations.

```
mimar 11000 1000 4 -lf inputmimar -u 2e-8 -t .005 -ej 1e4 -en 1 1 .1 -en .8 1 1 -o soutput
```

Other MCMC options

Relaunch `MIMAR` from an interrupted run

`MIMAR` may crash for various reasons. In this case, you can relaunch `MIMAR` from the interrupted MCMC with the switch "`-R`":

`mimar nsteps` *`0`* Y *`-R output-cont last_step [original_cmd_line]`* `-o` *`soutignore`*

Note that `bsteps` should be 0 in this case.

`original_cmd_line` The list of switches with the same values specified in the original `MIMAR` command line.

`soutignore` The name of the new summary output file. The histograms and summary statistics should be ignored since it is records only the results from the second run.

`output-cont` The file name in which the end of the standard output with the continuing posterior distribution will be recorded.

`last_step` The last COMPLETE line of the interrupted posterior distribution in the standard output generated by `MIMAR` (i.e., the nine values specified for each steps, see section "The standard output"):
The step number, θ_1, θ_2, t, T, θ_A, M_{12}, M_{21}, $p(\mathbf{D}|\mathbf{\Theta})$.

To calculate estimates, one would need to consolidate the interrupted posterior and the new one. Here is a step by step example:

1. I ran then interrupted the following comment line:
   ```
   mimar 11000 1000 4 -lf inputmimar -u 2e-8 -t u .001 .01 -ej u 0 1e5 -N .005 -M l -2
   2 -o exsoutempty >outputint
   ```
   "`exsoutempty`" is the summary output file but contains only the command line and the original seeds. The last line of the standard output file "`outputint`" is incomplete; the last two lines are:
   ```
   2580 0.00429539 0.00429539 0.0169149 3632.8 0.005 2.28723 2.28723 2.44217e-13
   2581 0.00429539 0.00429539 0.0169149 3632.8 0.005 2
   ```

2. I relaunched `MIMAR` with the last complete line of the file "`outputint`":
   ```
   ./mimar 11000 0 4 -R outputint-R 2580 0.00429539 0.00429539 0.0169149 3632.8 0.005
   2.28723 2.28723 2.44217e-13 -lf inputmimar -u 2e-8 -t u .001 .01 -ej u 0 1e5 -N
   .005 -M l -2 2 -o exsoutempty-R >emptypost
   ```
   "`exsoutempty-R`" should be ignored (unless one needs the seed). The standard output file `emptypost` contains only the first twelve lines of the file "`outputint`".

3. I reconstructed the full posterior distribution by copying and pasting the COMPLETE lines in the standard output file "`outputint`" at the beginning of the file "`outputint-R`".

To help recover interrupted runs, I also wrote the script "`perlRELAUNCHmimar`". This script recovers information from the summary and standard output files of the interrupted run, so that the new posterior distribution also contains the parameters accepted in the interrupted run. By default, the file names for the interrupted summary and standard output files provided by the user are "`exsoutempty`" and "`outputint`" and `MIMAR` will generate the files "`exsoutempty-R`" (which should be ignored) and "`outputint-R`" (which contains the complete posterior distribution).

Define the duration of the run in minutes

`mimar nsteps bsteps` Y `-lf input -u` μ `-t` θ_1 `-ej` T `-y` t `-o soutput`

By default, `nsteps` (and `osteps` see section "Generate summary output files at regular intervals") is defined in number of steps. However, if "`-y t`" is used (where `t` stands for "time"), `nsteps` (and `osteps`) defines the duration of the run in minutes. For the following command line, `MIMAR` will run for 24 hours (including 1000 steps of burnin):

```
mimar 1440 1000 4 -lf inputmimar -u 2e-8 -t .005 -ej 1e4 -y t -o soutput
```

If the ancestral alleles are unknown use MIMAR_noanc

I provide the program MIMAR_noanc which assumes that the summary statistics of the polymorphism were calculated *without known ancestral alleles*. MIMAR and MIMAR_noanc are identical in all other aspects. I also provide MIMARgof_noanc to perform a goodness of fit test on the estimates of the parameters from MIMAR_noanc.

Use MIMAR_noanc at your own peril! I did not test the program. I recommand performing a simulation study to assure that the estimates provided by MIMAR_noanc are accurate and precise. To to so, use the program MIMARsim_noanc to generate the input file for MIMAR_noanc with the summary statistics calculated assuming unknown ancestral states.

Compiling MIMAR_noanc

To compile the program type:

```
gcc -o mimar_noanc mimar_noanc.c params.c streec.c randX.c -lm
```

See section “Downloading and compiling” for more details.

The summary statistics without ancestral alleles

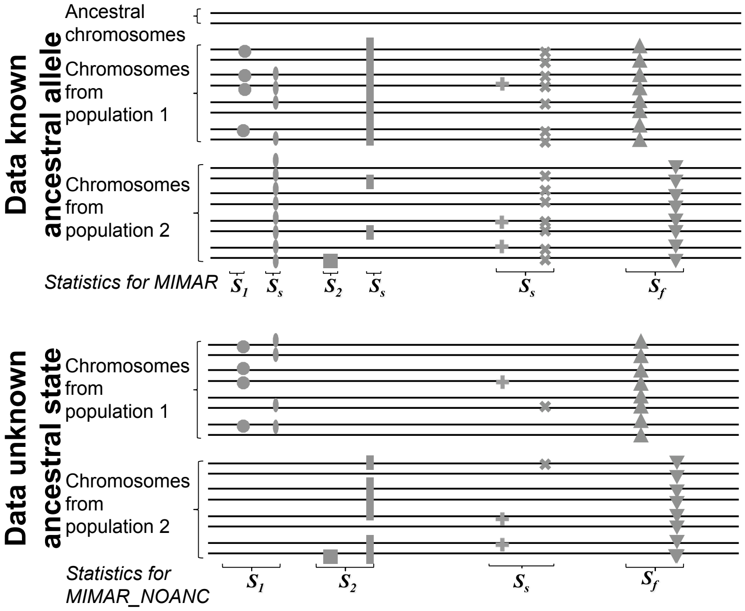

Ancestral chromosomes
Data known ancestral allele
Chromosomes from population 1
Chromosomes from population 2
Statistics for MIMAR
S_1
S_s
S_2
S_s
S_s
S_f
Data unknown ancestral state
Chromosomes from population 1
Chromosomes from population 2
Statistics for MIMAR_NOANC
S_1
S_2
S_s
S_f

Figure 5: Examples of polymorphisms for each statistics when the ancestral states are unknown. See legend of Figure 2. The upper panel shows the data with known ancestral alleles. The lower panel show the same locus assuming the ancestral and derived states are unknown. The grey symbols represent minor frequency (or derived in the upper panel) alleles found in the sample: the polymorphisms with the oval and disc mutations will count in S_1 (the minor allele frequencies are $f_1 = 0.5$ and $f_2 = 0$ in both cases), the sites with the square and rectangle mutations will count in S_2 (the minor allele frequencies are $f_1 = 0$ and $f_2 = 0.125$ (square) and $f_1 = 0$ and $f_2 = 0.25$ (rectangle)), the sites with the plus and cross mutations will count in S_s (the minor allele frequencies are $f_1 = 0.125$ and $f_2 = 0.25$ (plus) and $f_1 = 0.125$ and $f_2 = 0.125$ (cross)) and the sites with the triangle mutations will count in S_f ($f_1 = 1$ and $f_2 = 0$ (upward triangle) and $f_1 = 0$ and $f_2 = 1$ (downward triangle)).

MIMAR_noanc requires summaries of the polymorphism data at each locus as input. To summarize the data, we use the statistics from Wakeley and Hey (1997): for each locus, we consider the number of sites polymorphic uniquely to the samples from populations 1 and 2 (S_1 and S_2 respectively), the number of sites with shared alleles between the two samples (S_s), and the number of fixed polymorphisms in either sample (S_f).

The statistics are calculated as follows: Each polymorphic site is assigned to one of the statistics depending of its *minor allele frequency* in the population i, f_i. Specifically, if $0 < f_i < 1$ in *BOTH* population samples, the alleles are shared, if $f_i = 0$, $f_j = 1$, $i \neq j$, the alleles are fixed in either sample, and if $f_i = 0$ and $f_j < 1$, $i \neq j$, the derived allele is polymorphic specifically in sample j. For example, a polymorphic site is unique to sample 1 if the frequencies of the derived allele are $f_1 = 0.5$ and $f_2 = 0$. If $f_1 = 0$ and $f_2 = 0.625$ is polymorphic site is unique to sample 2. Similarly, a polymorphic site is shared if the frequencies of the derived allele are $f_1 = 0.125$ and $f_2 = 0.25$ (See Figure 5).

Summary of command line options

The following options are required

-t θ_1 Set the population mutation rate per bp to $4N_1\mu$ for population 1 (the default population).

-t u a b Set the prior distribution for θ_1 to Uniform$[a, b]$.

-u μ Set the mutation rate per bp to μ.

-lf input Set the input file name.

-ej T Set the time of split to T generations ago. Backward in time, all lineages in population 2 are moved to population 1 at time

-ej u a b Set the prior distribution of T to Uniform$[a, b]$.

-o soutput Set the name of the summary output file.

The following options are not required but are useful for the use of MIMAR. They define more complex models or help set an efficient MCMC.

-n θ_2 Set the population 2 mutation rate per bp to $4N_2\mu$.

-n u a b Set the prior distribution of θ_2 to Uniform$[a, b]$.

-N θ_A Set the ancestral population mutation rate to $4N_A\mu$.

-N u a b Set the prior distribution of θ_A to Uniform$[a, b]$.

-M M Set the expected number of migrants between the two populations each generations to $4N_1m$.

M l a b Set the prior distribution of $\ln(M)$ to Uniform$[a, b]$.

-m i j M_{ij} Set the expected number of migrants from population i into population j each generation, i and $j \in \{1, 2\}$, $i \neq j$, to $4N_1m_{ij}$.

-m i j l a b Set the prior distribution of $\ln(M_{ij})$ to Uniform$[a, b]$.

-r ρ Set the population recombination rate per bp to $4N_1c$.

-r e λ Set the prior distribution of $r = \frac{c}{\mu}$ to Exponential with mean $\frac{1}{\lambda}$.

-r n ν σ Set the prior distribution of $r = \frac{c}{\mu}$ to Normal(ν, σ).

-r 1 Set the locus-specific population recombination rates per bp to the value specified with w in the input file.

-r 2 Set the locus-specific recombination rates per bp to the value specified w in the input file.

-v $V_{\theta_1}V_{\theta_2}V_TV_{\theta_A}V_{M_{12}}V_{M_{21}}$ Set the variances of the kernel distributions.

-x ngen Set the number of genealogies (or ARGs) generated per locus to ngen.

-i int Set the interval between recorded sets of parameters to int steps

-L osteps Set the interval between intermediate summary output files to osteps steps (or minutes).

-R output-cont
last_step
[original_cmd_line] Relaunch MIMAR from an interrupted run. The run will start from the MCMC step last_step (which is the last COMPLETE line of the standard output from the interrupted run). The end of the posterior distribution will be recorded in the file "output-cont". [original_cmd_line]: The rest of the command line is the same as the one for the interrupted run (with the exeption of bstep=0)

-y t Set nsteps (and osteps) in minutes instead of number of steps.

The following options are conserved from `ms`. Use at your own peril!

`-se seed1 seed2 seed3` Specify the random seeds from the command line.

`-f filename` Read command line arguments from file `filename`.

`-c` f λ Set ratio of gene conversion to recombination to f and the track length to λ.

`-G` α Set growth parameter of all populations to α.

`-g` i α_i Set growth rate of population i to α_i.

The following options specify events occurring at time τT generations. Up to 10 such switches can be used. It is the user's responsibility to specify times that are compatible with the isolation-migration model. Note that the switch "`-ej`" can be used only once.

`-eG` τ α Set all growth rates to α at time τT generations.

`-eg` τ i α_i Set growth rate of population i to α_i at time τT generations.

`-eb` τ x Set all population mutation rates to $x\theta_1$ at time τT generations.

`-en` τ i x Set population i mutation rate to $x\theta_1$ at time τT generations.

`-eM` τ x Set the symmetrical migration rate to x at time τT generations.

`-em` τ i j x Set $4N_1m_{ij}$ to x at time τT generations.

List of parameters and symbols

$\theta_i = 4N_i\mu$	The population mutation rate per bp per generation for population $i \in \{1, 2, A\}$.
μ	The generational mutation rate per bp.
N_i	The diploid effective size for population $i \in \{1, 2, A\}$.
T	The split time in generations, at which, backward in time, all lineages in population 2 are moved to population 1.
$t = \frac{T}{4N_1}$	The split time in coalescent unit.
τ	The ratio of the time of the event divided by the split time of the isolation-migration mode
$M = 4N_1m$	The expected number of individuals in population 2 replaced by migrants from population 1 each generation (in forward direction).
m	The symmetrical fraction of a population that is made up of migrant from the other population each generation.
$M_{ij} = 4N_1m_{ij}$	The expected number of individuals in population j replaced by migrants from population i (in forward direction), i and $j \in \{1, 2\}$, $i \neq j$.
m_{ij}	The fraction of population j that is made up of migrant from population i each generation.
$\rho = 4N_1c$	The population recombination rate per bp per generation.
c	The generational recombination rate per bp.
$r = \frac{c}{\mu}$	
Y	The number of loci considered.
Z_r	The initial sequence length of a locus in base pairs before filtering.
Z	The locus length in base pairs after filtering.
n_i	The sample size for the locus in population $i \in \{1, 2\}$.
x	The inheritance scalar reflecting copy number differences for a locus.
v	The mutation rate variation scalar for a locus.
w	The inheritance and rate variation scalar for recombination rate for a locus.
ω	The ratio of the locus-specific population recombination rate per bp over ρ.
S_i	The number of derived polymorphisms unique to the samples from population $i \in \{1, 2\}$.
S_s	The number of polymorphisms with shared derived alleles between the two samples.
S_f	The number of polymorphisms with fixed alleles in either sample.

List of files in the directory "mimardir"

Program files for MIMAR	**Examples of input files**	**Examples of summary output files**	**Examples of standard output files**	**Scripts and files to help with MIMAR analysis**	**Other or documentation files**
mimar.c	inputmimar	exsoutempty	emptypost	MIMARchart.xls	makefile
mimar.h	inputmimar_4Nc	exsoutempty-R	outputint	MIMARconv.R	MIMARdoc.pdf
params.c	inputmimar_c	exsoutput	outputint-R	MIMARplot.R	mimar_noanc.c
rand1.c		exsoutput0	outputmimar	perlRELAUNCHmimar	mimarrun
rand1t.c		exsoutput1	outputmimar0		msdoc.pdf
rand2.c		exsoutput2	outputmimar1		seedmimar
rand2t.c			outputmimar2		
streec.c					

Downloading other programs and documentations

MIMARgof is found in "mimar.tar" available at http://przeworski.uchicago.edu/cbecquet/download.html.
ms and "msdoc.pdf" are available at http://home.uchicago.edu/~rhudson1/source/mksamples.html.
R is available at http://www.r-project.org/.

Acknowledgments

Many thanks to G. Coop and M. Przeworski for helpful comments on the documentation files for MIMAR and the related programs, and to R. Hudson for his permission to use some paragraphs from the "msdoc.pdf" file.

References

Becquet, C. and Przeworski, M., 2007. A new approach to estimate parameters of speciation models with application to apes. *Genome Res.*, **17**:1505–1519.

Becquet, C. and Przeworski, M., 2009. Learning about modes of speciation by computational approaches. *Evolution*, **63**:2547–2562.

Gilks, W. R., Richardson, S., and Spiegelhalter, D. J., 1996. Implementation. In *Markov Chain Monte Carlo In Practice*, chapter 1.4, pages 8–19. Chapman and Hall/CRC, Boca Raton, Florida.

Hey, J. and Nielsen, R., 2004. Multilocus methods for estimating population sizes, migration rates and divergence time, with applications to the divergence of *Drosophila pseudoobscura* and *D. persimilis*. *Genetics*, **167**:747–760.

Hudson, R. R., 1983. Properties of a neutral allele model with intragenic recombination. *Theor. Popul. Biol.*, **23**:183–201.

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

Kimura, M., 1969. The number of heterozygous nucleotide sites maintained in a finite population due to steady flux of mutations. *Genetics*, **61**:893–903.

Wakeley, J. and Hey, J., 1997. Estimating ancestral population parameters. *Genetics*, **145**:847–855.