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

C. Becquet

June 20, 2007

This document describes how to use MIMAR, a program that estimates the demographic parameters of an "isolation-migration" model from recombining loci (see Fig. 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 (Fig 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: 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.

The method uses Markov Chain Monte Carlo (MCMC) to explore the posterior of the parameters given the data. Briefly, the data are summarized for each locus by the four statistics studied by Wakeley and Hey (1997): The number of polymorphisms unique to the samples from populations 1 and 2 (S_1 and S_2 respectively), the number of shared 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 msdoc.pdf provided with ms (Hudson, 2002). We 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.* (in review).

Downloading and compiling

All relevant files are included in the tar file mimar.tar available at http://mplab.bsd.uchicago.edu/dataNprograms.htm. 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 rand`X`.c -lm`

(X is either 1, `2`, `1t` or `2t`) or alternatively, by typing `make`, which contains this compilation line with `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

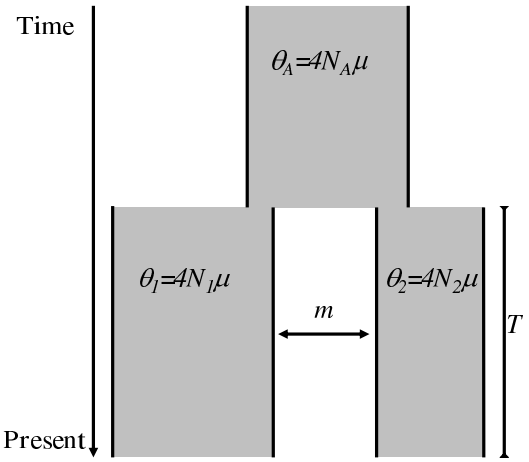

Figure 1: **The "isolation-migration" model**, in which two populations diverged T generations ago from a common ancestral population. The parameters θ_1, θ_2 and θ_A, are the population mutation rates for populations 1, 2 and the ancestral population, respectively. μ is the mutation rate per base pair 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 $N_1 = \theta_1/4\hat{\mu}$). 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. `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. The migration rate is specified in terms of the expected *number* of individuals in population 2 replaced by migrants from population 1 (in forward direction), $M = 4N_1m$, where N_1 is obtained from θ_1 by dividing by $4\hat{\mu}$ ($\hat{\mu}$ is the estimate of μ provided by the user with the switch "`-u`"). Specifically, unless M is fixed, $\ln(M)$ is chosen from a uniform distribution. In the examples I give in the following sections, I assume that some parameters are known, and thus fixed to the known value, while other parameters are estimated, and thus that their values are chosen from prior distributions.

Briefly, `MIMAR`'s estimation relies on the information contained in the four statistics studied by Wakeley and Hey (1997), calculated for multiple loci. The summary statistics are described below. 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 (see 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, 3/4 for X-linked and 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 base pairs, and the locus-specific population mutation rates for other populations are defined analogously. The set of locus-specific population recombination rates, $(\rho_1, \cdots, \rho_Y)$, is referred as $\mathbf{P}$.

The population recombination rate per base pair is defined as $\rho = 4N_1c$, where c is the per base pair per generation recombination rate. We ignore gene conversion, treating all recombination as crossovers alone. We also define an inheritance scalar for recombination, w (i.e. $w = 0$ for the mtDNA and Y, 2/3 for X and 1 for autosomes). There are three options to specify the population recombination rate. Either ρ is fixed across loci, such that the population recombination rate at a locus is $wZ\rho$. Alternatively, if an estimate of the population recombination rate per bp is available for each locus, $\widehat{\rho}$, the scalar w can be set to the inheritance scalar for recombination multiplied by $\widehat{\rho}$ to incorporate this knowledge in the estimation (see details below). Finally, rates can vary for each locus, in which case the locus specific population recombination rate is given by $r \cdot wZ\theta_1$ and the ratio $r = \rho/\theta_1$ is drawn from an exponential distribution prior with mean λ for each locus and each step of the MCMC.

The summary statistics

`MIMAR` requires summaries of the polymorphism data at each locus as input. To summarize the data, we use the statistics introduced by Wakeley and Hey (1997) for this type of inference problem: for each locus, we consider the number of polymorphisms unique to the samples from populations 1 and 2 (S_1 and S_2 respectively), the number of shared alleles between the two samples (S_s), and the number of fixed alleles 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. Specifically, if $0 < f_i \leq 1$ in each population sample, the 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 allele is specific to sample j.

The input

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

- The locus name, $\texttt{Name}_y$.
- The length of the locus in bps, Z_y. Note that this length corresponds to the number of bps in the locus, ignoring indels and sites with more that two alleles.
- The inheritance scalar, x_y (i.e., 1 for autosomal loci, 0.75 for X- and 0.5 for Y- and mtDNA-linked loci).
- The mutation rate scalar, v_y (which can be estimated e.g., from divergence data).
- The recombination scalar, w_y (i.e., 1 for autosomal loci, .67 for X- and 0 for Y- and mtDNA-linked loci). Note that if estimates of the recombination rates are available and variable across loci, this scalar can be used to incorporate this information (see below).
- The sample size for the locus in population 1 and 2, n_{1y} and n_{2y}, respectively.
- The summary statistics of the polymorphism (see description above): S_{1y} S_{2y} S_{sy} S_{fy}.

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

$\texttt{Name}_1$ Z_1 x_1 v_1 w_1 n_{11} n_{21} S_{11} S_{21} S_{s1} S_{f1}
$\texttt{Name}_2$ Z_2 x_2 v_2 w_2 n_{12} n_{22} S_{12} S_{22} S_{s2} S_{f2}
...
$\texttt{Name}_Y$ Z_Y x_Y v_Y w_Y n_{1Y} n_{2Y} S_{1Y} S_{2Y} S_{sY} S_{fY}

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.

Here is an example for a four locus data set, in which `locus1` is autosomal, `locus2` is X-linked and `locus3` and `locus4` are Y- or mtDNA-linked:

```
Example of input file for MIMAR

Name length xi vi wi n1 n2 S1 S2 Ss Sf //
locus1 1000 1 1 1 10 10 4 3 4 0
locus2 1000 0.75 1 0.5 10 10 5 4 5 0
locus3 1000 0.25 1 0 10 10 0 2 2 0
locus4 1000 0.25 1 0 10 10 4 2 4 0
```

The basic command line:

`mimar nsteps bsteps` Y `-lf input -u` μ `-t` θ_1 `-ej` T `-o soutput [other parameters][options]`

This line shows the simplest usage of `MIMAR`. There are three arguments to `MIMAR`: `nsteps`, `bsteps` and Y 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` is the number of steps (or minutes if the switch `-y t` is used) until the end of the MCMC run. `bsteps` is the number of burnin steps. Y is the number of loci considered. `input` is the input file name containing the information on the loci with their S statistics (see "The input" section above). μ is the per generation mutation rate per bp. `soutput` is the summary output file name, which contains a sample from the posterior distribution of the parameters represented in 1000 bins. `[options]` can be a list of any options described in the next sections.

θ_1 is the population mutation rate per bp for the first population, $4N_1\mu$. When the other population mutation rates are not specified, they are equal to θ_1. T sets the split time in generations, at which, backward in time, all lineages in population 2 are moved to population 1. 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 below). 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.

Here is an example of the basic command line:

```
mimar 11000 1000 4 -lf inputmimar -u 2e-8 -t .005 -ej 10000 -o exsoutput >outputmimar
```

In this case, the program will analyze the data in `inputmimar` file (see example above). `MIMAR` will output 10,000 steps after 1,000 steps of burnin in the file `outputmimar`, and it will output the marginal posterior distributions to file `exsoutput` (see below for more detail). Note that since in this example θ_1 and T are fixed, the different steps of the MCMC will all have the same parameter values: $\theta_1 = \theta_2 = \theta_A = 0.005$, $T = 10,000$ generations and $M = 0$.

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` θ_2 `-N` θ_A `-M` M

θ_2 is the population mutation rate per bp for the second population, $4N_2\mu$. θ_A is the ancestral population mutation rate per bp, $4N_A\mu$. The switch `-M` M sets the average number of migrants between the two populations to $M = 4N_1m$, 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).

As an example, if one types:

```
mimar 11000 1000 4 -lf inputmimar -u 2e-8 -t .005 -n .003 -ej 10000 -N .004 -M .7 -o
                                    exsoutput >outputmimar
```

`MIMAR` will proceed as for the above example, but all the MCMC steps will have the parameter values: $\theta_1 = 0.005$, $\theta_2 = 0.003$, $\theta_A = 0.004$, $T = 10,000$ generations and $M = 0.7$. Note that for the two examples above, the output will be 10,000 lines with the same parameters values since all the parameters are fixed.

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 "Estimating parameters of the Isolation-Migration model" section)). 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:

- For θ_1: `-t u` a b draws θ_1 from Uniform$[a, b]$.
- For θ_2: `-n u` a b draws θ_2 from Uniform$[a, b]$.
- For T: `-ej u a b` draws T from Uniform$[a, b]$.
- For θ_A: `-N u` a b draws θ_A from Uniform$[a, b]$.
- For M: `-M l` a b draws $\ln(M)$ from Uniform$[a, b]$.

Here is an example command line with some parameters estimated:

```
mimar 11000 1000 4 -lf inputmimar -u 2e-8 -t u .001 .01 -ej u 0 1e5 -N .004 -M l -2 2 -o
                                    exsoutput >outputmimar
```

In this case, θ_1, T and M are estimated: the prior range for θ_1 is 0.001 to 0.01, the prior range for T is 0 to 10,000 generations, the average number of migrants, M, is between 0.135 and 7.39. θ_2 is not provided so will be equal to θ_1 at every step, while θ_A is fixed to 0.004.

The outputs of MIMAR's analysis

Standard output

An example output for this command line, printed in the file `outputmimar` in this case, would be:

```
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 0.004 0.004
M12 -2 2
M21 0 0

Step # theta1 theta2 T (coalescent unit) T (genealogies) thetaA M12 M21 L
1001 0.00487319 0.00487319 0.0615793 15004.4 0.004 5.72314 5.72314 6.12E-11
...
```

The first line of `MIMAR`'s output is the input file, then the parameter values or ranges of the prior distributions are given in a table. The next line is the header of the result table followed by the results (here, only the first line of 1000 sets of parameters after the burnin is shown). Each line in the result table corresponds to one accepted set of parameters in the MCMC (note that the switch `-i int` defines how

often sets of parameters are recorded, by default `int`=1). In order, the results are: the step number, the parameters (θ_1, θ_2 , $t = T/4N_1$ in coalescent unit, T in generations, θ_A , $M_{12} = 4N_1m_{12}$, $M_{21} = 4N_1m_{21}$) and the likelihood of the data summaries for all loci given the parameters and the set of ARGs generated for this step. We provide t in coalescent units, so the user can estimate T using a different estimate of μ by $T = \theta_1 \times t/\hat{\mu}$. When one estimates a symmetrical migration rate as in this case, M, this value is indicated in both M_{12} and M_{21} columns.

The summary output

An example output for this command line of the first eight lines printed in the summary output (called `exsoutput` in this case)would be:

```
./mimar 11000 1000 4 -lf inputmimar -u 2e-8 -t u .001 .01 -ej u 0 1e5 -N .004 -M l -2 2 -o
exsoutput
3825 64750 52493

# generated steps. Accepted H>1. Accepted H<1. Acceptance rate. Rejected H<1. Rejected L==0.
Rejected outside prior. cpumin. cpusec.
11000 581 551 0.102909 2997 6 6865 0 5.12

theta1 theta2 Tcoal Tgen thetaA M12 M21
0.001004505 0 0.001004505 0 0.001001001 0.003 50.05005005 0 0.002000202 0 0.135606496 0
```

The first line is the command line. The second line lists the random number seeds (see "Downloading and compiling" section). Following these two lines are nine values providing information about the analysis to help you ensure that the burnin was large enough and that the MCMC has converged: (1) the total number of steps generated, the number of steps accepted with the Hasting ratios: (2) $h \geq 1$ and (3) $h < 1$, (4) the acceptance rate, the number of steps rejected (5) when $h < 1$ or (6) because the estimate of the likelihood was 0 for at least one of the locus or (7) because one of the parameter value was outside the prior range, and finally the time of execution in (8) minutes and (9) seconds. 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}, while M_{21} is empty.

The eight last lines of the summary output `exsoutput` provide the point estimates (mean, mode, median), variance and 90% central credible interval of the marginal posterior distributions:

```
...
parameters mean mode var per.05 perc.5 perc.95
Theta1 0.00650578 0.00995946 4.99E-06 0.00278829 0.00615766 0.00994144
Theta2 0.00650578 0.00995946 4.99E-06 0.00278829 0.00615766 0.00994144 Tcoal=Tgen/4N1 0.0817832
0.035035 0.012246 0.00700701 0.039039 0.447447
Tgen 19834.5 15265.3 3.51E+08 3253.25 13363.4 82932.9
ThetaA 0.004 0.0040002 0 0.0040002 0.0040002 0.0040002
M12 1.69047 0.146314 4.6641 0.142845 0.412302 6.88952
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.

- **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 soutput files of two different seeds. Using the switch -L osteps, one can also output a soutput file every osteps steps (or minutes if the switch -y t is used), which allows to monitor the analysis and its convergence over time. I also provided the file MIMARplot.R, which reads the results from the standard output (e.g., outputmimar file) and allows the user to print the marginal posterior distributions with different numbers of burnin steps and different numbers of bins using functions in R. 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 Fig. 2. I provided the file MIMARconv.R to help output those). If the parameter value plots show trends over long portions of the run, or if autocorrelations persist for a large number of steps (see Fig. 2b), this means that the state space is being explored slowly, in which case longer runs are required. 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 the case of ngen ARGs, 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.
  - Alternatively, mixing can be monitored using the acceptance rate in the soutput file, which one usually wants to be at least 0.05. One way to increase the acceptance rate is to increase the burnin (bsteps), reduce or increase the variance of the kernel distribution of the estimated parameters (-v switch), increase the number of genealogies used to estimate the likelihood (switch -x ngen), and/or reduce the ranges of the prior distributions. Getting a good mixing might require experimenting and may not be perfect (i.e. the acceptance rate may still be smaller than 0.05).

Reduce output file size

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

If nsteps is large, the standard output may become large quickly, because by default all accepted parameters are recorded (i.e. int=1). To avoid this inconvenience, use the switch -i int to specify the interval between accepted parameters recorded in the output. The command line:

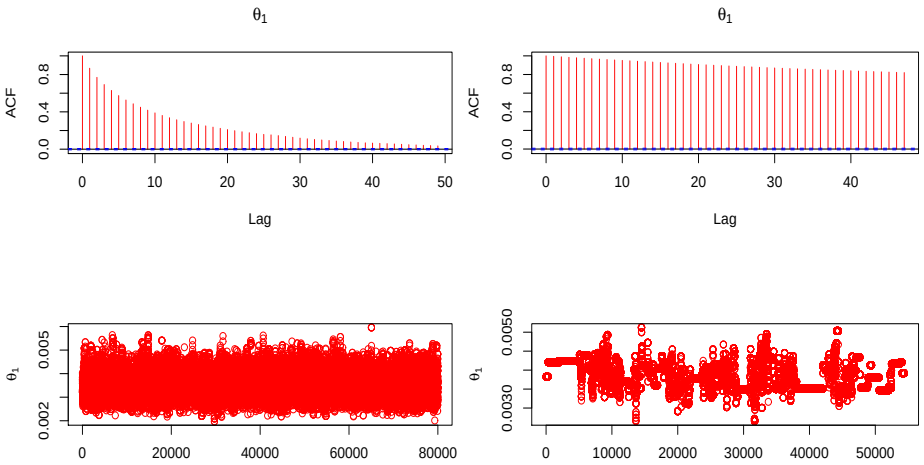

Figure 2: Example of properly mixing (a) and badly mixing (b) MCMC. The top panels are graphs using the `acf()` function, and the bottom panels plot the parameter value (here θ_1) along the run.

```
mimar 11000 1000 4 -lf inputmimar -u 2e-8 -t .005 -ej 1e5 -i 10 -o exsoutput
>outputmimar
```

will output the accepted parameters every 10 steps (the first two lines of the results will be for steps number 1001 and 1011).

Change the number of genealogies generated per locus

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

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 paper). If the acceptance rate is low, it may be an indication that the likelihood is not estimated reliably. In particular, if some sets of parameters are rejected because the estimated likelihood is 0 for at least one of the loci, it is an indication that genealogies often lead to a likelihood estimate of 0 for at least one of the loci. 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 1e5 -x 100 -o exsoutput
>outputmimar
```

Change the variance for the normal kernel distributions

`mimar nsteps bsteps` Y `-lf input -u` μ `-t` θ_1 `-ej` T `-v` V_{θ_1} V_{θ_2} V_T V_{θ_A} $V_{M_{12}}$ $V_{M_{21}}$ `-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. After exploratory simulations, I chose the variances that maximized the acceptance rate for the data I used (Gilks et al., 1996). 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). In particular, if the number of parameters sets rejected because a parameter value was outside its prior range is much greater than that of sets rejected when the Hasting ratio was <1, this indicates that the variances may be too large.

Alternatively, if the results in `MIMARconv.R` show a parameter value plot with trends over long portions of the run or strong autocorrelation (see Fig2b), this indicates that its variance may be too small. 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 entered). 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: $V_{\theta_1} = 10^{-3}$, $V_T = 5 \times 10^5$ and $V_{M_{12}} = V_{M_{12}} = V_{\theta_2} = V_{\theta_a} = 0$:

```
mimar 11000 1000 4 -lf inputmimar -u 2e-8 -t u 0 .01 -ej u 0 1e6 -v 1e-3 0 5e5 0 0 0 -o
exsoutput >outputmimar
```

Define duration of run

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

By default, `nsteps` (and `osteps`) 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 u 0 .01 -ej u 0 1e6 -y t -o exsoutput
>outputmimar
```

Monitor a run

There are two ways to output summary files during a run. The first simply requires that the file "`mimarrun`" be present in the directory and that its first lines begin with "y". `MIMAR` will write the summary output into `mimarrun`, which will now starts with "no". If you want to print a summary output, simply write "yes" in the first line of `mimarrun`.

You can also print summary outputs at regular intervals, using :

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

`osteps` is the number of steps separating summary outputs. Each output will be named "`soutput-x`", where `x=1` when `1`$\times$`osteps` steps are reached, `2` when `2`$\times$`osteps` steps are reached... Friendly advice: do not set `osteps` $<$ `nsteps`$/50$, as the directory may become very large indeed. If `osteps` $<$ `bsteps`, the outputs within the burnin period will contain empty histograms. The following command line will generate five summary outputs, 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 u 0 .01 -ej u 0 1e6 -L 2000 -o exsoutput
>outputmimar
```

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

```
mimar 1440 1000 4 -lf inputmimar -u 2e-8 -t u 0 .01 -ej u 0 1e6 -y t -L 60 -o exsoutput
>outputmimar
```

More complex models

Crossing over:

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

- **Fixed ρ across loci.** For each locus y with recombination scalar w_y>0 , `MIMAR` will generate genealogies under a coalescent model with recombination (Hudson, 1983). To include crossing-over in the model, use the `-r` switch and specify the population cross-over rate parameter, $\rho = 4N_1c$, where c is the probability of cross-over per generation per bp. The locus specific recombination rate is $w_y Z_y \rho$. For example, for the following command line, the recombination rate for `locus1` and `locus2` in `inputmimar` (see input file example above) is 1*1000*0.005=5 and .67*1000*0.005=3.35, respectively:

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

- **Fixed locus specific ρ.** When an estimate of the per bp population recombination rate is available for the locus y, $\widehat{\rho}_y$, you can set $w_y = s_y \cdot \widehat{\rho}_y$, where s_y is the recombination scalar due to locus chromosomal location (i.e. 1 for autosomal loci, .67 for X- and 0 for Y- and mtDNA-linked loci). Note that you must specify $w_y = s_y \cdot \widehat{\rho}_y$ for all the recombining loci. By fixing ρ in the command line to 1, the recombination rate for locus y will be $w_y Z_y \cdot 1 = s_y \cdot \widehat{\rho}_y Z_y$. For example, if you know that the population recombination rate per bp for `locus1` is .005 and for `locus2` is .002, then the recombination scalars will be $w_1 = 0.005 \star 1$ and $w_2 = 0.002 \star 0.67 = 0.00134$ (see "The input" section and file `inputmimarrho`), and you can use the following comment line to analyze `inputmimarrho`:

```
mimar 11000 1000 4 -lf inputmimarrho -u 2e-8 -t u .001 .01 -ej u 0 1e5 -r 1 -o exsoutput
                                        >outputmimar
```

- **Variable locus specific ρ.** The recombination rate can be allowed to vary across loci and across steps. In this case, the user needs to use the switch `-r e` $1/\lambda$. Every step of the MCMC, a new ratio, $r_y = \rho_y/\theta_1$, is drawn from an exponential distribution with mean λ for each locus. The locus specific recombination rate becomes $r_y w_y Z_y \theta_1$. Note that the recombination rates for the loci are nuisance parameters, and are chosen independently across loci and across steps. For example, to obtain an average of 0.6 recombination events per mutation, one enters:

```
mimar 11000 1000 4 -lf inputmimar -u 2e-8 -t u .001 .01 -ej u 0 1e5 -r e 1.667 -o
                                        exsoutput >outputmimar
```

Asymmetrical migration rates:

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

In the original paper, we only considered symmetrical migration rates. However, there is also an option to fix or estimate asymmetrical migration rates. Thinking forward in time, the average number of individuals that migrate from population i into population j 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 `l` a b, in which case $\ln(M_{ij})$ is drawn from Uniform$[a, b]$.

The command line:

```
mimar 11000 1000 4 -lf inputmimar -u 2e-8 -t .005 -ej 1e5 -m 1 2 l -2 2 -m 2 1 l -2 2 -o
exsoutput >outputmimar
```

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).

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 `-o soutput -r` ρ `-c f` λ

See `msdoc.pdf` for details.

Exponentially growing or shrinking population size

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

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 t, 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 $t \times T$ in generations. The arguments which follow the time parameter specify populations and other relevant parameters, as indicated in the following list:

`-eG` t α	Set all growth rates to α at time $t \times T$ generations.
`-eg` t i α_i	Set growth rate of population i to α_i at time $t \times T$ generations.
`-eb` t x	Set all population mutation rates to $x\theta_1$ at time $t \times T$ generations.
`-en` t i x	Set population i mutation rate to $x\theta_1$ at time $t \times T$ generations.
`-eM` t x	Set the symmetrical migration rate to x at time $t \times T$ generations.
`-em` t i j x	Set $4N_1 m_{ij}$ to x at time $t \times T$ generations.

For example:

```
mimar 11000 1000 4 -lf inputmimar -u 2e-8 -t u 0 .01 -ej u 0 1e6 2 1 -o exsoutput -en 1
1 .1 -en .8 1 1 >outputmimar
```

specifies the first population mutation rate to one-tenth of its current value (in this case θ_1) between T and $.8T$ generations.

Summary of command line options

The following options are required

`-t` θ_1	Set the population 1 mutation rate per bp to $4N_1\mu$.
`-t u` a b	Set the prior distribution for θ_1 to Uniform$[a, b]$.
`-u` μ	Set value of the mutation rate per base pair.
`-lf input`	Set the input file name.
`-ej` T	Set the time of split to T generations ago.
`-ej u` a b	Set the prior distribution of T to Uniform$[a, b]$.
`-o soutput`	Set the summary output file name.

The following options are not required, but useful for the use of `MIMAR`.

`-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 average number of migrants between the two populations 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 average number of migrants from population i into population j, 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` $1/\lambda$	Set the prior distribution of $r = \rho/\theta_1$ to exponential with mean λ.
`-r 1`	Set the locus-specific recombination rates to $w \star Z$ (given in the `input` file).
`-i int`	Set the interval between recorded sets of parameters to `int` steps
`-x ngen`	Set the number of genealogies (or ARGs) generated per locus to `ngen`.
`-v` $V_{\theta_1}V_{\theta_2}V_TV_{\theta_A}V_{M_{12}}V_{M_{21}}$	Set the variances of the kernel distributions.
`-y t`	Define `nsteps` (and `osteps`) in minutes instead of number of steps.
`-L osteps`	Output summary output files every `osteps` steps (or minutes).

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

`-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 $t\times T$ 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 `-ej` and `-eN` are used

only once.

`-eG` t α	Set all growth rates to α at time $t \times T$ generations.
`-eg` t i α_i	Set growth rate of population i to α_i at time $t \times T$ generations.
`-eb` t x	Set all population mutation rates to $x\theta_1$ at time $t \times T$ generations.
`-en` t i x	Set population i mutation rate to $x\theta_1$ at time $t \times T$ generations.
`-eM` t x	Set the symmetrical migration rate to x at time $t \times T$ generations.
`-em` t i j x	Set $4N_1 m_{ij}$ to x at time $t \times T$ generations.

Downloading other programs and documentations

`MIMARgof` is found in `mimar.tar` available at `http://mplab.bsd.uchicago.edu/dataNprograms.htm`. `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.* (in review).

Gilks, W., Richardson, S., and Spiegelhalter, D., 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., 1983. Properties of a neutral allele model with intragenic recombination. *Theoretical Population Biology*, **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.