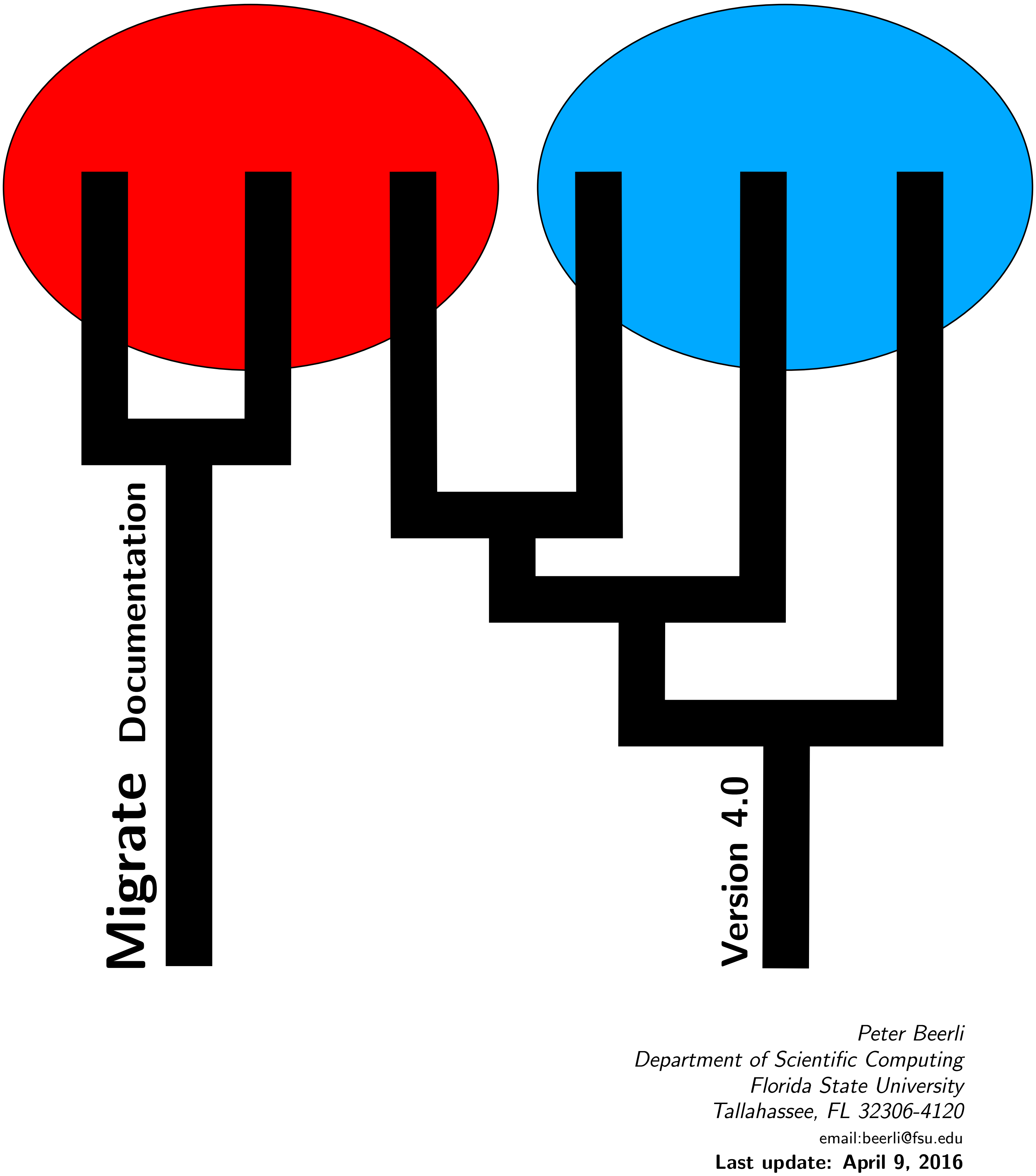

Migrate Documentation

Version 4.0

Peter Beerli
Department of Scientific Computing
Florida State University
Tallahassee, FL 32306-4120
email:beerli@fsu.edu

Last update: April 9, 2016
Started: January 1, 1997

For the impatient

Reading manuals is not a favored task of many, me included. But to achieve some results with `migrate` you should read at least the sections about

- **Data file specifications**
- **Quick guide for achieving "good" results with `migrate`.**

Good luck,
Peter Beerli Tallahassee, July 2014

Improved parts in this manual (April 9, 2016)

- Rewrote several section, and removed the likelihood centric approach, MIGRATE 4.0 does only support Bayesian inference.
- Added new sections: Population splitting, Individual assignment, haplotyping, new data format.
- Tipdate support and skyline plots.

Unfinished parts in this manual (April 9, 2016)

- Hardware support: parallel runs on Macintosh computers using `migrateshell.app`

Contents

Introduction

The program MIGRATE *estimates population size, migration, population splitting parameters using genetic/genomic data.*

For many purposes in biology, we need to know the effective population size of a population and also how well populations interact with other populations. there are essentially two very different approaches to get such information: a behavioral or ecological approach that asks for monitoring of individuals in a focus population and recognize residents and newcomers. Often individuals are marked with tags or other means (banding in birds, toe clipping in amphibians, and more recently inserting magnetic tags under the skin of animals). Such approaches are difficult with large populations, or small number of immigrants, or species that have a hidden lifestyle.

Since 1960 an alternative approach has been used. This approach uses the genetic makeup of an individual as a tag and measures similarities (or differentials) among groups of individuals. This work led to estimators such as F_{ST}, that indicate how isolated populations are from each other and several other measures that are based on allele frequencies within populations or individuals. These methods are most often based on simple population models that were invented by Sewall Wright and Ronald Fisher. The most common applications used the Wright-Fisher population model that assumes that the population does not grow or shrink, that every individual has the same chance to reproduce and that every generation that population of adults is replaced by their offspring. Interestingly, this simply model was (and is) amazingly stable and even applications to species where such a model seems outlandish (Elephants, humans, etc) allowed considerably insight into the history of populations. Unfortunately, practitioners are still using these methods despite considerable advances of population genetic theory. Problematic issues with these allele frequency approaches mostly stem from the fact that the assumptions of symmetric immigration rates and equal population sizes need to be fulfilled (*Beerli*, 2004).

Recent approaches based on the coalescent (*Kingman*, 2000b) allow better formulation of explicit probabilistic model that can handle different immigration rates and different population sizes, and also the addition of additional complications, such as recombination, population splitting etc. MIGRATE in its most simple form can only handle population sizes, immigration rates, and some forms of population splittings, therefore may be not suitable for all datasets. But often, it may help to decide what to do next, despite potential problems with assumption violation (*Beerli*, 2009).

This manual describes the program MIGRATE, its benefits, but also its shortcomings. In detail you will learn about how to use it and what options are available. This manual is only a start, I suggest that you subscribe to the `migrate-support@googlegroups.com` and participate in the community that uses MIGRATE.

Theoretical consideration

A short overview of the math that is used by the program MIGRATE. *If you want to treat* MIGRATE *as a black box, then skip this section.*

The program MIGRATE infers population genetic parameters from genetic data. Essentially we want to find the Bayesian posterior probability distribution of parameters $\mathcal{P}$ of a particular model given the Data $\mathcal{D}$:

$$\text{Prob }(\mathcal{P}|\mathcal{D}).$$

This posterior probability of the population genetics parameters $\mathcal{P}$, such as population sizes or migration rates, can be calculated in principle by integrating over all possible relationships $\mathcal{G}$ of the sample data $\mathcal{D}$ using an expansion of the coalescent theory (*Kingman*, 1982b,a, 2000a) which includes migration (*Hudson*, 1991; *Nath* and *Griffiths*, 1993; *Notohara*, 1990) and/or population splitting (for example, **?**).

$$\text{Prob }(\mathcal{P}|\mathcal{D}) = \frac{\text{Prob }(\mathcal{P},\mathcal{D})}{\text{Prob }(\mathcal{D})} \tag{1}$$

which is equivalent to

$$\text{Prob }(\mathcal{P}|\mathcal{D}) = \frac{\text{Prob }(\mathcal{P})\text{Prob }(\mathcal{D}|\mathcal{P})}{\text{Prob }(\mathcal{D})} \tag{2}$$

A Bayesian would tell us to use this

$$\text{Prob }(\mathcal{P}|\mathcal{D}) = \frac{\text{Prob }(\mathcal{P})\int_G \text{Prob }(G|\mathcal{P})\text{Prob }(\mathcal{D}|G)dG}{\text{Prob }(\mathcal{D})}, \tag{3}$$

to calculate the posterior we will also need to calculate the likelihood

$$\text{L}(\mathcal{P}) = \text{Prob }(\mathcal{D}|\mathcal{P}) = \int_G \text{Prob }(G|\mathcal{P})\text{Prob }(\mathcal{D}|G)dG. \tag{4}$$

The integration over all genealogies is not a simple integral, but a sum over all possible labeled histories and integrals over all possible branch lengths b_i

$$\text{L}(\mathcal{P}) = \sum_T \int_{b_1} ... \int_{b_k} \text{Prob }(T,\underline{b}|\Theta)\text{Prob }(\mathcal{D}|T,\underline{b})db_1...db_k. \tag{5}$$

Older versions of MIGRATE than version 4.0 could use both approaches to estimate the parameters. It became a major burden updating the program to maintain both likelihood and Bayesian inference, so that I decided to strip out the likelihood material and focus on the Bayesian approach, which is often easier to code and maintain. One of the major headaches with the likelihood approach was the maximization of the likelihood function that became more and more complicated with more parameters, this maximization is not needed in the Bayesian approach, although we still need to calculate likelihoods.

Specifically, MIGRATE estimates migration rates, population splitting times, and effective population sizes of 1 to many populations using genetic data (Fig 1). The parameters to estimate are

$$\mathcal{P} = \begin{pmatrix} \underline{\Theta} & \underline{\mathcal{M}} & \underline{\Delta} & \underline{S_\Delta} \end{pmatrix}. \tag{6}$$

Mutation-scaled population sizes

$$\underline{\Theta} = \begin{pmatrix} \Theta_1 & \Theta_2 & ... & \Theta_n \end{pmatrix} \tag{7}$$

where each $\Theta_i = xN_e\mu$ with μ that is the mutation rate per generation and with x that is a multiplier that depends on the ploidy and inheritance of the data, for nuclear data it $x = 4$, for haploid data it is $x = 2$, and for mtDNA in vertebrates with female-only transmission, it is $x = 1$. Life history is important, for example some fish species, such as Grouper, change sex in their lifetime and therefore all individuals can transmit mtDNA resulting in having $x \simeq 2$ and not $x = 1$.

Mutation-scaled immigration rates

$$\underline{\mathcal{M}} = \begin{pmatrix} - & \mathcal{M}_{2\to1} & \mathcal{M}_{3\to1} & ... & \mathcal{M}_{n\to1} \\ \mathcal{M}_{1\to2} & - & \mathcal{M}_{3\to2} & ... & \mathcal{M}_{n\to2} \\ ... & ... & ... & ... & ... \\ \mathcal{M}_{1\to n} & ... & ... & \mathcal{M}_{(n-1)\to n} & - \end{pmatrix} \tag{8}$$

which is the immigration rate per generation m divided by the mutation rate per generation μ, it is a measure of how much more important immigration is over mutation to bring new variants into the population. The traditional number of immigrants per generation xNm is $\Theta\mathcal{M} = xN_e\mu\times m/\mu = xNm$. The mutation rate μ is per site for DNA data and per locus for microsatellite or allelic data. If you compare results of different programs make sure that you understand what μ represents, oftentimes it is per locus even with DNA data!

There seems to be considerable confusion about migration directions in the recent literature. When Sewall Wright discussed migration, he considered immigration rate into a population, and since we do not observe immigrants directly, he also assumed that our data represents the offspring of locals and immigrants. In his framework he did not consider emigration and that immigration will happen instantaneous (e.g no delayed influx of gene through seed banks etc). If we see $\mathcal{M}_{2\to1}$ then we always use a forward time perspective: in the past the individuals was in population 2 and now is in population 1.

The immigration parameter in MIGRATE is a longtime average over the genealogy of the individuals in the sample. This works well for populations that fluctuate around a value, but even with mildly growing populations this works fine, only with strongly growing populations one may get underestimates using MIGRATE.

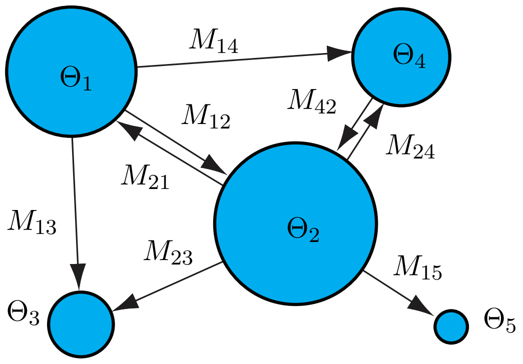

Figure 1: Populations exchanging migrants with rate $m_{j \to i}$ per generations and with size N_e. The parameters are scaled by mutation rate μ which is with sequence data per site per generation. The estimated parameters are therefore: Θ_i which is $xN_e^{(i)}\mu$ and $\mathcal{M}_i$ which is m_i/μ, the migration estimate is often also expressed as xNm which is just $\Theta\mathcal{M}$, x is the inheritance parameter and depends on the data, commonly 4 for nuclear data, and 1 for mtDNA data. The example model is not a complete (full) model because some migration routes are not estimated and set to zero.

Mutation-scaled population splitting times Δ and standard deviations S_Δ

The population splitting times are model differently to other programs that all use the model of IM (**?**). My new model was worked out by **?**. We treat population splitting events as individual events on a lineage. Looking backward in time, a lineage ℓ currently in population κ is at risk of being in a different population either by migration with rate $\mathcal{M}$ or into another population by a splitting event. The Splitting event is proposed during the MCMC run using a hazard function of the splitting time distribution, for simplicity we use a truncated Normal distribution with mean Δ and standard deviation S_Δ. Drawing these splitting events using a hazard function is equivalent to the coalescent or migration event which are also hazard functions. During the MCMC runs many different splitting events will be proposed from the Normal distribution with Δ_j and S_{Δ_j} for j population splits. The current version needs guidance which populations are merging (looking backwards in time), for an example see the Bayes Factor section. In the parmfile and the menu the setting of the splitting parameters is combined with setting the migration model, for an example see definition of the custom migration matrix.

Bayesian inference

MIGRATE estimates the parameters using a Bayesian paradigm (see formula 3), simulation studies of simple models show that there are few differences with the ML runs, although some combinations of parameters might be be easier to estimate with the Bayesian approach (*Beerli*, 2006). One of the bigger problems with the Likelihood approach is the effort that has to be made to calculate the confidence intervals of the best estimates, default analyses often led to narrow support intervals thus making researchers overconfident about the explanatory power of their data; the use of the prior distribution seems to make a Bayesian approach less vulnerable to this problem. Bayesian inference is commonly based on Markov chain Monte Carlo (MCMC) because we usually cannot integrate the function of

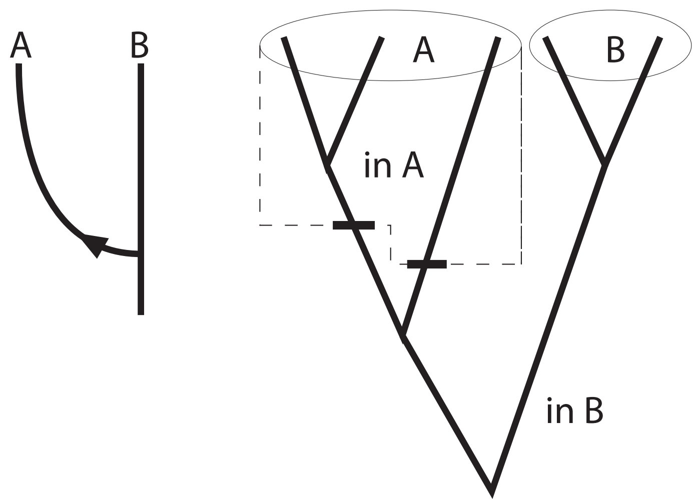

Figure 2: Populations splitting, left is the model specification, two populations A and B, A splits off B. Right, detail view for the gene tree. Each individual can split off by itself.

interest analytically or by simple numerical approach. MCMC was described first by *Metropolis* et al. (1953) and refined by *Hastings* (1970). For an introduction see *Hammersley* and *Handscomb* (1964)or *Chib* and *Greenberg* (1995), and see *Kuhner* et al. (1995) for a first application using MCMC in the context of coalescence theory.

Commonly we think of the marginal posterior distribution of each parameter (summing/integrating over all 'nuisance' parameters). We use particular values to drive the MCMC, and in a Bayesian analysis we get them from the Prior distribution, this may help to get good results in situations where the data suggests a very rough landscape of parameter- and tree-space (see Fig. 3 for an example with a smooth surface). MIGRATE allows using several different prior distributions, some are more appropriate for you data than others. I often suggest to use the uniform prior distribution because it is simple and shows obvious deficiencies in an analysis very quickly, but it also tends to increase the credibility intervals because it supports very large or very small parmeter values equally. This may sound as an odd choice because for example we know that the population size of humans never was zero and never was 10^{10}, so a uniform prior distribution with range 0 to 10^{10} does not sound right although such a prior would do fine in an analysis because the data is strong enough to suggest that the posterior probability near a size of zero is close to zero and the probability of a size of 10^{10} is also small.

The parameter r is the a uniform random number from (0,1].

Prior distributions

Uniform prior distribution

The parameters have a uniform distribution between a minimal and a maximal value of the parameters, there is a set of minima and maxima for $\underline{\Theta}$ and $\underline{\mathcal{M}}$. MIGRATE calculates the uniform by using

$$\text{Prob}\,(\mathcal{P}_i) = \frac{1}{\mathcal{P}_{max} - \mathcal{P}_{min}} \tag{9}$$

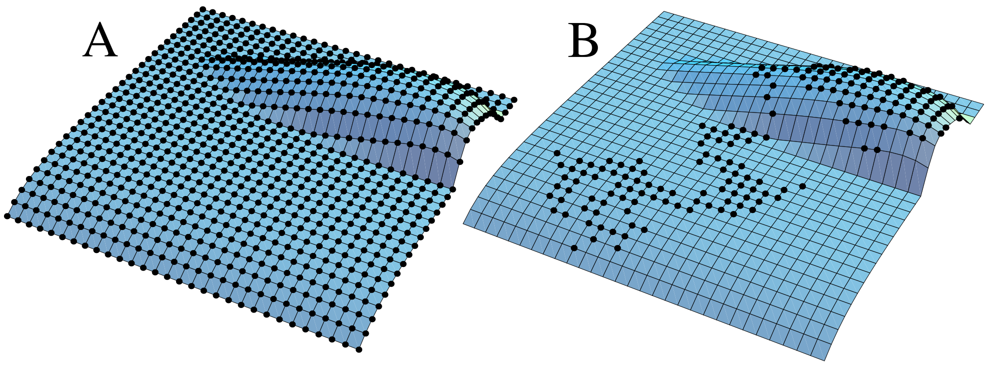

Figure 3: (A) On an imaginary, infinite likelihood surface we would need to sample every possible genealogy and sum all these values which is not possible, but trees with low probability will not contribute much to the final likelihood, (B) by biasing towards better trees we can sample effectively from those trees with high contribution to the final likelihood and can approximate the likelihood (4).

it is implemented using a windowing method with window size Δ, that is preferrably around 1/10 of the whole range.

$$\mathcal{P}_{\text{new}} = \mathcal{P}_{\text{old}} + (2\Delta r - 1) \begin{cases} \mathcal{P}_{\text{new}} < \mathcal{P}_{\text{min}} & \mathcal{P}_{\text{min}} + |\mathcal{P}_{\text{min}} - \mathcal{P}_{\text{new}}| \\ \mathcal{P}_{\text{new}} > \mathcal{P}_{\text{max}} & \mathcal{P}_{\text{max}} - |\mathcal{P}_{\text{max}} - \mathcal{P}_{\text{new}}| \end{cases} \tag{10}$$

Gamma distribution prior

The truncated gamma distribution has four parameters α, β, minimum a, and maximum b. The gamma prior in MIGRATE is defined through the mean μ, α, a, and b

$$\alpha = \mu/\beta \tag{11}$$

$$\beta = \text{minimization of the mean of the truncated gamma and the parameter } \mu \tag{12}$$

$$\text{Prob } (\mathcal{P}_i) = \text{probability of the truncated gamma} \tag{13}$$

Exponential prior distribution

The parameters have a exponential distribution, MIGRATE calculates three versions
Simple exponential prior distribution

$$\text{Prob } (\mathcal{P}_i) = \int_0^\infty exp(-P_i/P_{\text{mean}})/P_{\text{mean}} d\mathcal{P}_i = exp(-P_i/P_{\text{mean}}) \tag{14}$$

$$\mathcal{P}_{\text{new}} = -\mathcal{P}_{\text{mean}}\ ln(r) \tag{15}$$

Exponential prior distribution with fixed window

$$\text{Prob }(\mathcal{P}_i, \mathcal{P}_{\text{min}}, \mathcal{P}_{\text{max}}) = \frac{\int_{\mathcal{P}_{\text{min}}}^{\mathcal{P}_{\text{max}}} exp(-P_i/P_{\text{mean}})/P_{\text{mean}} d\mathcal{P}_i}{exp(-P_{\text{min}}/P_{\text{mean}}) - exp(-P_{\text{max}}/P_{\text{mean}})} \tag{16}$$

$$= \frac{exp(-P_{\text{min}}/P_{\text{mean}}) - exp(-P_{\text{x}}/P_{\text{mean}})}{exp(-P_{\text{min}}/P_{\text{mean}}) - exp(-P_{\text{max}}/P_{\text{mean}})} \tag{17}$$

$$\mathcal{P}_{\text{new}} = -\mathcal{P}_{\text{mean}}\ ln(\frac{r}{exp(\mathcal{P}_{\text{max}}/\mathcal{P}_{\text{mean}})} - \frac{r-1}{exp(\mathcal{P}_{\text{min}}/\mathcal{P}_{\text{mean}})}); \tag{18}$$

Exponential prior distribution with variable window

$$\text{Prob }(\mathcal{P}_i | \mathcal{P}'_i, \mathcal{P}_{\text{min}}, \mathcal{P}_{\text{max}}) = \frac{2\int_{\mathcal{P}'_i-\Delta}^{\mathcal{P}'_i+\Delta} exp(-P_i/P'_i)/P'_i d\mathcal{P}_i}{exp(1) Csch(\Delta/P'_i)} \tag{19}$$

$$= \frac{(exp(\Delta + P_i)/P'_i - exp(1)) Csch(\Delta/\mathcal{P}'_i)}{2exp(\mathcal{P}_i/\mathcal{P}'_i)} \tag{20}$$

$$\mathcal{P}_{\text{new}} = \mathcal{P}'_i - \mathcal{P}'_i ln(exp(\Delta/\mathcal{P}'_i) - 2rSinh(\Delta/\mathcal{P}'_i)) \begin{cases} \mathcal{P}_{\text{new}} < \mathcal{P}_{\text{min}} & \mathcal{P}_{\text{min}} + |\mathcal{P}_{\text{min}} - \mathcal{P}_{\text{new}}| \\ \mathcal{P}_{\text{new}} > \mathcal{P}_{\text{max}} & \mathcal{P}_{\text{max}} - |\mathcal{P}_{\text{max}} - \mathcal{P}_{\text{new}}| \end{cases} \tag{21}$$

Files in MIGRATE

MIGRATE can use many different input methods and output methods, but most of them have a very special purpose, as a minimum you need to supply an input datafile, here called infile.

There are multiple ways to set up things. MIGRATE can use very different ways to manipulate the data and as a result many different files are needed or produced. Minimally, you need the data file, its default name is *infile*, and MIGRATE produces two files that contains results: the *outfile* (ASCII text file) and a PDF output file that contains the same information (well almost, as you see later). The program produces both formats because for quick checking of results the ASCII file can be opened on the command line or with any text viewer, whereas the PDF file requires a PDF reader, for example for macs Preview.app and for windows NitroPDF; unfortunately modern versions of the standard Adobe Acrobat Reader fail to read the PDF files generated by MIGRATE correctly – I will work on porting my PDF writer to a newer system, but this has low priority and will take a while (Older versions of Adobe Reader work fine!)

Input files

Filename	Type	Short description	Necessary?	Name changeable
infile	Input	holds you data	YES	Yes
parmfile	Input	holds options	-	Yes*
geofile	Input	holds a (geographic) distance matrix between the populations	-	Yes
datefile	Input	holds the date (default is years) of the sample. When used then you need also to supply a generation time and and a mutation rate per year in the `parmfile` or the Menu.	-**	Yes

* Under Unix the parmfile name ca be given as an argument to the program
** When different sample dates are used then this file is needed

Main input files

infile if this file is not present in the current directory than the program will ask for a data file, and you can give the path to it, you need to type the path, which is for Macintosh and Windows users probably rather uncomfortable. In the **menu** or **parmfile** you can specify an other default name for your datafile.

datefile When the samples came from different years and you believe that this makes a different specify the date as the time backward from today (for example years before 2007). With this analysis type, you need to specify a mutation rate in the same units as the dates of the samples.

bayesallfile The bayesallfile allows to reuse a previous Bayesian inference run, the data type menu needs to be set to "Genealogy". [This is not yet in the full production stage; use this with caution, make a backup-copy of the bayesallfile before you try this!]

Optional input files

parmfile can hold specific menu options, this file and the possible options for the menu are explained in detail in section **menu and parmfile**.

geofile holds the geographic or arbitrary distances between the populations. When this is used then the migration rates are not only scaled by the mutation rate but also by this distance. This allows to detect environmental barriers when we assume that the genetic potential to migrate is the same in all populations; without a barrier the rates should be all the same per distance unit. The format is like a distance file in the PHYLIP package (**?**), but you can use the # as a commentary character.

```
# Example geofile for 3 populations,
# the order of the population must be the same as in the data file
#
  3
Tallahasse0.0 10.0 150.0
St.Marks  10.0 0.0 160.0
Pensacola 150.0 160.0 0.0
```

The example scales the mutation-scaled migration rates by the 'geographic' distance. If the migration rate is linear with the inverse of the distance then the migration rates between all locations will be the same, here we scale the migration rate per distance unit, therefore if we have a range of 0 to 160 miles, the rates are are scaled per mile. As a result migration rates will be all relatively high because a mile is usually not a large distance for vagile species.

Output files

Filename	Short description	Name changeable
parmfile	holds options, menu can rewrite this file	see menu
outfile	will be created and replace any file with the same name in the same directory	Yes
outfile.pdf	contains the same output as outfile and histograms, you need a PDF viewer to read this file	Yes
bayesfile	contains the histogram data of a Bayesian run (the outfile.pdf used these to generate the posterior distributions.	Yes
bayesallfile	contains the raw data of a Bayesian run, can be run through TRACER when only a single replicate and a single locus is used.	Yes
mighistfile	contain the distribution of migration events over time.	Yes
skylinefile	contains the distribution of the parameter values over time as calculated by using the expected parameter values for a short time intervals.	Yes
treefile	holds genealogies, this file will be created and will replace any file with the same name in the same directory	No
logfile	logs the progress information that is displayed onto the screen into a file	Yes

Main output files

Some conbination of the output files are not possible, for example a standard Bayesian run will not fill values into the treefile, etc.

outfile and **outfile.pdf** Somewhere you want to read the results, that is it! The name **outfile** is the default, but can be changed either in the menu or the parmfile. The PDF file contains graphical representation of some of the table and values. Currently, most of the output is represented in the PDF file, when you used the Bayesian inference setting, with Maximum likelihood there are still some options that are not supported in the PDF file (I still lack a programmer to do all this).

treefile holds all, only those of the last chain, or the best tree(s). The likelihood of each tree is given ($\text{Prob}\ (\mathcal{D} \mid \mathcal{G})$) in a comment. The programs writes trees with migrations using the Newick format with extensions from the Nexus format. Such trees containing migration events can be printed using the program `Eventtree` (or short `ET`) (distributed from http://popgen.scs.fsu.edu/et). Writing trees to a treefile adds some burden to the program, it will run slower, especially with the option BEST. Parallel runs increase the communication with the master node and therefore may slow down your run.

bayesfile holds the posterior histogram data show in the PDF files. You can use other program packages like the matplotlib package in python (http://www.python.org), GNUPLOT (http://www.gnuplot.org), or the GMT package (http:www.soest.hawaii.org/gmt2) to recreate the histograms.

bayesallfile holds the raw posterior values for all parameters. This option reduces the memory footprint by writing all intermediate results to disk and then rereads them for summarizing and printing the final results. This file can be also used to independently test whether MIGRATE converged or not using the program TRACER (**??**), MIGRATE uses a simple 1-step Effective sample size (ESS) calculator that may not always be very accurate, although comparison showed that seeing high autocorrelation in MIGRATE means to see high autocorrelation (small ESS) in TRACER.

mighistfile holds the histogram over time of the frequency of migration and coalescence events, with simulated data these plots show typically an exponential decay. When there were changes of parameters over time then the data will enforce different patterns, that can be used to discuss the results.

skylinefile holds the averages of the expected parameter values at specific times. These plots are similar to the skyline plot reported in BEAST (**?**), although their derivation is an extension of the original skyline plots of (**?**). MIGRATE reports changes of population sizes and migration rates over time and summarizes over multiple loci.

Data models

A short overview of the different datatypes and how multiple loci are summarized.

MIGRATE allows for several different input data types, such as electrophoretic marker data, microsatellite data, sequence data as stretches of contiguous sites and as single nucleotide polymorphisms.

Infinite allele model

This assumes that every mutation will result in a new allele, there is no back mutation (Fig. 4). This model is used in all current implementations of electrophoretic data analyses packages (Biosys-1, GDA among others) and perhaps is appropriate for this kind of data. *Migrate* is calculating the parameters for each locus independently and summarizes at the end taking the likelihood surfaces or Posterior distributions of each locus into account.

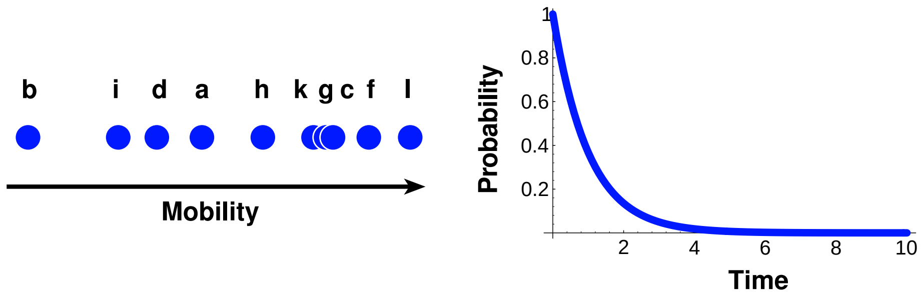

Figure 4: Left: Mobility of electrophoretic marker in an electric field. the alleles a,b,c,.. describe a possible sequence of mutation, their mobility is not correlated with the mutational history. Right: The probability that a given allele is not mutating during some time, this is a simple exponential relationship.

Microsatellite model

Ladder model

The ladder model was invented by citeohta:1973:amm and **?** for electrophoretic markers, but was not as good as expected in describing real electrophoretic alleles. For microsatellites this model seems much more appropriate cite[e.g.][]valdes:1993:afm, but see **?**, here obviously change happens mostly by slippage of the two DNA strands creating with higher probability a new allele which is only 1 step apart from the old than one which 2 steps apart (Fig. 5).

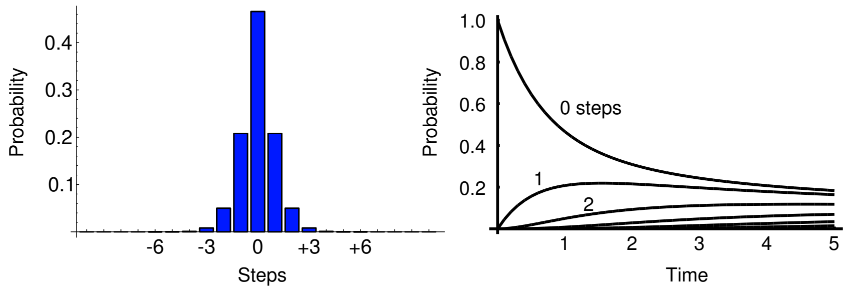

Figure 5: Left: Number of repeat changes of a microsatellite marker. The probability to have a slippage of only one repeat is higher than the slippage of more than one repeat, in a given time, here time=0.1. Right: The probability that a change of 0,1,2,.. steps is occurring during some time.

Brownian motion approximation to the ladder model

This replaces the discrete stepwise mutation model with a continuous Brownian motion model The results are very similar to the exact stepwise mutation model, but the parameter estimation is several times faster. This is a crude approximation that has some difficulties when the dataset is not very variable because it uses a cutoff for the the probability that there is no change between two points on a branch, during a time of x the Brownian motion approximation replaces discrete jumps between repeats with a continuous approximation.

DNA/RNA model

Sequence model

Migrate implements the sequence model of Felsenstein (1984) available in `dnaml` (PHYLIP 4.0, Felsenstein 1997)(Fig. 7). The transition probabilities were published by Kishino and Hasegawa (1989). *Migrate* does not allow for recombination within a locus and therefore may over-estimate variability because of recombination, but this bias is not explored well, if in doubt I suggest to try

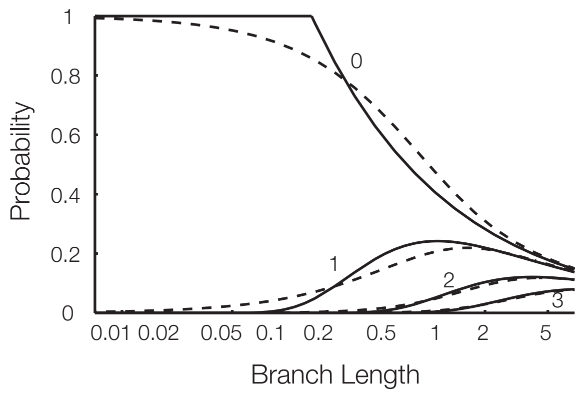

Figure 6: Comparison of stepwise mutation model with Brownian motion approximation (dashed lines). The numbers 0, 1, 2, 3, 4 are the number of steps. The Brownian motion approximation for no change is truncated at 1. With steps of more than 4 there is no differences between the stepwise model and the approximation. X-Axis is in $\log_1 0$

to run MIGRATE, simulated high recombination rate data leads to difficulties with convergence. Applications of recombination tests beforehand may work well, but most of these recombination recognition program use the 4-gamete test that is based on the infinite sites model and therefore will overestimate the importance of recombination.
Like `dnaml`, *Migrate* also allows for different evolutionary rates, mutation categories and autocorrelation, although any use of these additional features can slow done to program to a crawl, but this may change in the future as computers double their speed roughly every 2 years.

Single nucleotide polymorphism data (SNP)

We use a rather restrictive ascertainment models for SNPs **?**. A better approach than using SNPs is the use of short reads which may or many not contain SNPs. I find that SNPs are an inferior datatype because commonly researchers are adding criteria such as a minor SNP allele must occur at a frequency higher than x, and singletons are excluded etc.

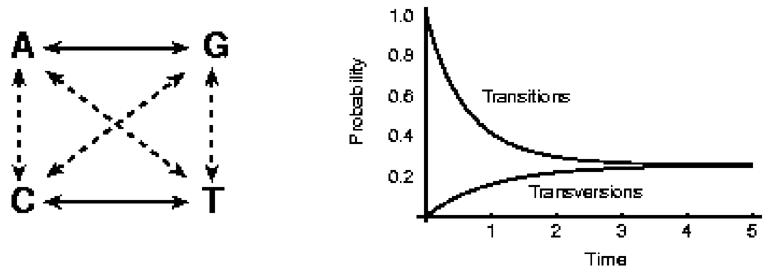

Figure 7: Left: Sequence mutation model. Transitions are are shown in black lines, transversion are shown with dotted lines. Right: The probability that a transition or transversion is occurring during some time. The shown graph uses equal base frequencies, but the used model does not need this restriction.

1. We have found ALL variable sites and use them even if there are only a few members of another alleles present. In principal it is as you would sequence a stretch of DNA and then remove the invariant sites. Each stretch is treated as completely linked. You can combine many of such "loci" to improve your estimates.

This is certainly not how people develop SNPs, but currently the closest we can come up with. The SNP coding is otherwise exactly the same as the coding for DNA data.

Combining multiple loci

MIGRATE calculates all loci estimates independently, the multi-locus estimate is not a simple average over all loci, but takes into account the likelihood or posterior distribution for each parameter at each locus. Loci with flat likelihood curves or flat posteriors will not contribute as much as those with strongly peaked distributions. MIGRATE offers different treatment in the mutation menu of the parameter menu for the mutation rate among loci:

```
Mutation rate among loci
(C)onstant    All loci have the same mutation rate [default]
(E)stimate    Mutation rate
(V)arying     Mutation rates are different among loci [user input]
(R)elative    Mutation rates estimated from data
```

The **Constant** assumption forces each locus to have the same identical mutation rate. Try this first, because it is the least complicated and most often gives fine results. The **Estimate** is the most difficult and needs dated samples, without sample dates do not use this option. The last two options, **Varying and Relative**, are probably the best ones to try if you really need variable mutation rates. When you know the relative differences of mutation rates in your data, you can specify them. Alternatively, let MIGRATE estimate the relative mutation rate using your data. For sequences MIGRATE calculates a simple Watterson's effective population size estimate over all samples and for each locus and then uses that to calculate a relative mutation rate. With microsatellites and allozyme data MIGRATE counts the number of alleles and uses those as a measure of relative mutation rates. The mean of these rates is 1.0.

Data file specification

In detail specification of the data format, without reading and using this information your analysis will most likely faulty.

The data needs to be in a certain form; for us, the following formats were most convenient, but you need to edit your data into this form. There are some programs that can write MIGRATE files, for example the program MSAANALYZER (**?**) that can generate MIGRATE datafiles from excel spread sheets for microsatellite data. The following formats are discussed in detail:

- DNA or RNA sequence data (single locus and multilocus)
- Single nucleotide polymorphism data (two formats)
- Microsatellite marker data (MIGRATE uses **REPEATNUMBERS by DEFAULT!!!!!!!**)
- Allozyme data (or other infinite allele mutation model marker)

General data format

Syntax: a token is either a word, a collection of words, or a character or a number:

$< token >$ the token between the the "angle-brackets" is obligatory

$[token]$ in square brackets are optional.

$\{token\}$ are obligatory for some

$< token1|token2 >$ choose one of the token kind of data. If this is too abstract, look at the examples further down.

A range of numbers in a "word" token as in <`individual1 10-10`> means that this token needs to be 10 characters long. The characters for any word token can normally include special characters, punctuation, and spaces, the token for the individual name "`Ind1 02 @`" is legal. An explanation of the individual parts follows at the end of this section. The most common data file for allozyme data or microsatellite data would look like this (examples follow):

```
<Number of populations> <number of loci> {delimiter between alleles} [project title 0-79]
{#@M <msat1-repeatlength> <msat2-repeatlength> .....}
<Number of individuals> <title for population 0-79>
<Individual 1 10-10> <data>
<Individual 2 10-10> <data>
....
<Number of individuals> <title for population 0-79>
```

```
<Individual 1 10-10> <data>
<Individual2 10-10> <data>
....
```

The delimiter is needed for microsatellite data and the project title is optional. The line starting with `#@M` is not necessary when the data consists of allozyme data or microsat repeat numbers. The line allows to automatically calculate the number of repeats from the fragment length. The `data` will be described in the following sections. **The population name must start with a alphabetical character (not a number). The individual name has to be 10 characters by default** (same as in PHYLIP), but can be changed to another constant in the parmfile, even to a length of 0. [This is one of the most common errors, make sure that your individual names are 10 characters, it does not matter whether they are all alphanumeric, spaces are fine]

For sequences or SNPs, the syntax is slightly different, the following case is for non-interleaved sequence data.

```
<Number of populations> <number of loci> [project title 0-79]
<number of sites for locus1> <number of sites for locus 2> ...
<Number of individuals locus1> <#ind locus 2> ... <#ind loc n> <title for population 0-79>
<Individual 1 10-10> <data locus 1>
<Individual 2 10-10> <data locus 1>
....
<Individual 1 10-10> <data locus 2>
<Individual 2 10-10> <data locus 2>
....
<Number of individuals>  <#ind locus 2> ... <#ind loc n>  <title for population 0-79>
<Individual 1 10-10> <data locus 1>
<Individual 2 10-10> <data locus 1>
....
<Individual 1 10-10> <data locus 2>
<Individual 2 10-10> <data locus 2>
....
```

For each locus one can give different number of individuals, if there is only one number then the program assumes that all loci have the same number of individuals. If there a fewer numbers than loci the last number will substitute for the number of individuals at the other loci. It is important that the population name does not start with a number!

MIGRATE version older than 4.0 supported interleaved sequence formats, Ivstopped supporting this and have therefore removed its description, reformat your data to a non-interleaved format before you translate it into the MIGRATE format. I typically use PAUP* **?** to export a non-interleaved PHYLIP formatted datafile and use that to change into the MIGRATE format.

A data type called **HapMap** is available for SNP data that allows less cumbersome input of SNP data than old versions of MIGRATE, but see current imput format. You still can use a single site as a locus (SNP), but with many loci this will be difficult to manage. The HapMap data type uses this format:

```
<Number of populations> <number of loci> [project title 0-79]
```

```
<Any Number> <title for population 0-79>
<Position on chromosome locus1> <TAB><allele><TAB><number><TAB><allele><TAB><number><TAB><total>
<Position on chromosome locus2> <TAB><allele><TAB><number><TAB><allele><TAB><number><TAB><total>
....
<Any Number>  <title for population 0-79>
<Position on chromosome locus1> <TAB><allele><TAB><number><TAB><allele><TAB><number><TAB><total>
<Position on chromosome locus2> <TAB><allele><TAB><number><TAB><allele><TAB><number><TAB><total>
....
```

The current format assumes that each SNP is biallelic. <allele> contain the nucleotide and the <number> contains the number of individuals with that specific allele, the total number is the sum of both, and is currently not necessary, but I may use this later to accommodate slight extension of this format, currently the total number is read from the program but not further used. This format will extend to more useful analyses that take into account the position on the chromosome, but this is currently not used.

Summary of the individual tokens

`<Number of populations>`	Number of populations. Range: $1, 2, 3, \ldots, n$ where n is a smallish number, remember that the default MIGRATE run estimates n^2 parameters.
`<Number of Loci>`	Number of unlinked loci. Range: $1, 2, 3, \ldots, \ell$ where ℓ can be a large number.
`<Delimiter>`	can be any character that does not occur in some other function in the data set, examples: `@` , . /
`<Number of individuals>`	Number of individuals within one population. Range: $1, 2, 3, \ldots, m$. For exploring MIGRATE I suggest to use around 10 to 20 individuals, much less (for example 1 or 2) or more (for example 1000) will make the analysis more difficult and need more experience and patience.
`<Title for population>`	Title for the population, the first letter must **not be a Number**!
`<Individual>`	Remember that the default for individual names needs 10 characters. Ideally, each individual name is unique and the first letter of the individuals name represents a code for the populations, for example $0, 1, 2, ..$
`<Data>`	See examples for the different data types.
`<Number of sites>`	Number of linked sites. Range: $1, 2, 3, \ldots, S$
`<Position on Chromosome>`	Location on genome measured in sites [not functional yet]
`<Allele>`	For SNP data this is one of the nucleotides: A, C, G, T.
`<Number>`	For SNP data this is the number of <Allel> at that specific site in the sample.
`<Total>`	For SNP data this is the total number of samples at the specific site.

Examples of the different data types

The examples in this section look like real data, but they are only for the demonstration of the syntax, if you try run this "data" it will deliver often very strange values, I have added a "usable" test set of simulated data in the examples directory, see the file examples/README for more information.

Allozyme data (infinite allele model)
The data is given in genotypes, any printable character with ASCII code bigger than 33 ('!') and smaller than 128 can be used. '?' is reserved for missing data. You can use multi-character coding when you use a delimiter (see the examples for microsatellites). If there is enough interest I can work on a input using gene frequencies, although I prefer to work on more interesting things than adjusting input files.

Most simple example with a single locus, 2 population and 5 total individuals.

```
 2 1 Migration rates between two Turkish frog populations
3 Akcapinar (between Marmaris and Adana)
PB1058    ab
PB1059    ab
PB1060    b?
2 Ezine (between Selcuk and Dardanelles)
PB16843   ab
PB16844   bb
```

Example with 2 populations and 11 loci and with 3 and 2 individuals per population, respectively (this data set is only an example of syntax, analyzing this dataset would not make much sense).

```
 2 11 Migration rates between two Turkish frog populations
3 Akcapinar (between Marmaris and Adana)
PB1058    ee bb ab bb bb aa aa bb ?? cc aa
PB1059    ee bb ab bb bb aa aa bb bb cc aa
PB1060    ee bb b? bb ab aa aa bb bb cc aa
2 Ezine (between Selcuk and Dardanelles)
PB16843   ee bb ab bb aa aa aa cc bb cc aa
PB16844   ee bb bb bb ab aa aa cc bb cc aa
```

Same example, but with a different syntax that allows multicharacter allele names (see last locus!). The delimiter is specified as the third parameter in the first line, the delimiter cannot be a standard alphabet character.

```
 2 11 / Migration rates between two Turkish frog populations
3 Akcapinar (between Marmaris and Adana)
PB1058    e/e b/b a/b b/b b/b a/a a/a b/b ?/? c/c Rs/Rf
PB1059    e/e b/b a/b b/b b/b a/a a/a b/b b/b c/c Rs/Rs
PB1060    e/e b/b b/? b/b a/b a/a a/a b/b b/b c/c Rs/Rs
2 Ezine (between Selcuk and Dardanelles)
PB16843   e/e b/b a/b b/b a/a a/a a/a c/c b/b c/c Rf/Rf
PB16844   e/e b/b b/b b/b a/b a/a a/a c/c b/b c/c Rf/Rs
```

Microsatellite data

DEFAULT INPUT SYNTAX

The third argument on the first line has to be a delimiter character, for example a **"."**. The data is given in genotypes. Each individual has two alleles. Alleles are coded as **REPEAT NUMBERS**, so for example your sequence

```
Flanking    msat    Flanking
region              region
---------------------------
ACCTATAGCACACACACACAAATGCGA      6 CA repeats
ACCTATAGCACACACACA--AATGCGA      5 CA repeats
```

contains a microsatellite with 6 repeats. And if with a homozygote individual it needs to be coded as 6.6 or 06.06, where the "," is the delimiter. '?' is reserved for missing data.

Example:

```
 2 3 . Rana lessonae: Seeruecken versus Tal
2   Riedtli near Guendelhart-Hoerhausen
0         6.5 37.31 18.18
0         6.6 37.33 18.16
2  Tal near Steckborn
1         4.5 35.? 18.18
1         4.4 35.31 20.18
```

FRAGMENT LENGTH INPUT SYNTAX

Earlier version of The third argument on the first line has to be a delimiter character, for example a **"."**. The data is given in fragmentlength. Each individual has two alleles. Alleles are coded as **FRAGMENTLENGTH**, so for example your sequence

```
Flanking    msat    Flanking
region              region
---------------------------
ACCTATAGCACACACACACAAATGCGA      27 sites total length
ACCTATAGCACACACACA--AATGCGA      25 sites total length
```

contains a microsatellite with 6 repeats, but you only have measures of the total length, here for the first allele there are 27 sites and the second allele there are 25 sites. This format needs an additional line to tell MIGRATE that we use fragment length and that MIGRATE needs to do the translation to repeat numbers, inspect closely the line that starts with #@M in the example below. The #@M tells the program that here comes a definition of the microsatellite repeats, and the numbers force MIGRATE to assume that the loci are dinucleotide repeats (2) , ot trinuleotide with 3 or tetranulceotides with 4 nucleotides per repeat, and so forth.

And if with a homozygote individual it needs to be coded as 25.25 or 025.025, where the "." is the delimiter. A heterozygote would read 25.27, for example. '?' is reserved for missing data.

Example:

```
  2 3 . Rana lessonae: Seeruecken versus Tal
#@M 2 2 2
2   Riedtli near Guendelhart-Hoerhausen
0         25.27 137.131 218.218
0         27.27  218.216
2  Tal near Steckborn
1         23.25 135.? 218.218
1         23.23 135.131 220.218
```

Sequence data

The sequence data format has received a face-lift: two new formats are now allowed (1) the old format described below and (2) the new format that is more appropriate for genomic type data.

After the individual name follows the base sequence of that species, each character being one of the letters A, B, C, D, G, H, K, M, N, O, R, S, T, U, V, W, X, Y, ?, or - . Blanks will be ignored, and so will numerical digits. This allows GENEBANK and EMBL sequence entries to be read with minimum editing. These characters can be either upper or lower case. The algorithms convert all input characters to upper case. The characters constitute the IUPAC (IUB) nucleic acid code plus some slight extensions (Table 1). They enable input of nucleic acid sequences taking full account of any ambiguities in the sequence.

Table 1: IUPAC (IUB) convention for naming nucleotide sites and ambiguous sites

Symbol	Meaning		Symbol	Meaning	
A	Adenine		B	not A	(C or G or T)
G	Guanine		D	not C	(A or G or T)
C	Cytosine		H	not G	(A or C or T)
T	Thymine		V	not T	(A or C or G)
U	Uracil		X,N,?	unknown	(A or C or G or T)
Y	pYrimidine	(C or T)	O	deletion	
R	puRine	(A or G)	-	deletion	
W	"Weak"	(A or T)			
S	"Strong"	(C or G)			
K	"Keto"	(T or G)			
M	"aMino"	(C or A)			

Most simple example with 1 population and a DNA-locus with 50 basepairs.

```
   1 1 Make believe data set using simulated data (1 locus)
50
3  Tallahassee
Peter     ACACCCAACACGGCCCGCGGACAGGGGCTCGAGGGATCACTGACTGGCAC
Donald    ACACAAAACACGGCCCGCGGACAGGGGCTCGAGGGGTCACTGAGTGGCAC
Christian ATACCCAGCACGGCCGGCGGACAGGGGCTCGAGGGAGCACTGAGTGGAAC
```

```
   1 1 Make believe data set using simulated data (1 locus) OLDFORMAT
50
3  Tallahassee
Peter     ACACCCAACACGGCCCGCGGACAGGGGCTCGAGGGATCACTGACTGGCAC
Donald    ACACAAAACACGGCCCGCGGACAGGGGCTCGAGGGGTCACTGAGTGGCAC
Christian ATACCCAGCACGGCCGGCGGACAGGGGCTCGAGGGAGCACTGAGTGGAAC
```

The new format looks very similar in its simplest version

```
   1 1 Make believe data set using simulated data (1 locus) NEWFORMAT
(s50)
3  Tallahassee
Peter     ACACCCAACACGGCCCGCGGACAGGGGCTCGAGGGATCACTGACTGGCAC
Donald    ACACAAAACACGGCCCGCGGACAGGGGCTCGAGGGGTCACTGAGTGGCAC
Christian ATACCCAGCACGGCCGGCGGACAGGGGCTCGAGGGAGCACTGAGTGGAAC
```

Same example, but now with 2 population and a single DNA-locus with 50 basepairs.

```
   2 1 Make believe data set using simulated data (1 locus) OLDFORMAT
50
3  Tallahassee
Peter     ACACCCAACACGGCCCGCGGACAGGGGCTCGAGGGATCACTGACTGGCAC
Donald    ACACAAAACACGGCCCGCGGACAGGGGCTCGAGGGGTCACTGAGTGGCAC
Christian ATACCCAGCACGGCCGGCGGACAGGGGCTCGAGGGAGCACTGAGTGGAAC
3  St. Marks
Lucrezia  ACACCCAACACGGCCCGCGGACAGGGGCTCGAGGGATCACTGACTGGCAC
Isabel    ACACAAAACACGGCCCGCGGACAGGGGCTCGAGGGGTCACTGAGTGGCAC
Yasmine   ATACCCAGCACGGCCGGCGGACAGGGGCTCGAGGGAGCACTGAGTGGAAC
```

In the new format this still looks very similar to the old format except for the line that contains the number of sites, the simple number of sites is replaced by the data type 's' and the parenthesis specifies that the locus is unlinked

More complicated example with 2 population AND with **2 loci**, the sequences are NOT interleaved, I drop the interleaved from because I find it error-prone cumbersome to change and unnecessary.

```
   2 2 Make believe data set using simulated data (2 loci) OLDFORMAT
50 46
3 3   pop1
eis       ACACCCAACACGGCCCGCGGACAGGGGCTCGAGGGATCACTGACTGGCAC
zwo       ACACAAAACACGGCCCGCGGACAGGGGCTCGAGGGGTCACTGAGTGGCAC
drue      ATACCCAGCACGGCCGGCGGACAGGGGCTCGAGGGAGCACTGAGTGGAAC
eis       ACGCGGCGCGCGAACGAAGACCAAATCTTCTTGATCCCCAAGTGTC
zwo       ACGCGGCGCGAGAACGAAGACCAAATCTTCTTGATCCCCAAGTGTC
drue      ACGCGGCGCGAGAACGAAGACCAAATCTTCTTGATCCCCAAGTGTC
2    pop2
vier      CAGCGCGCGTATCGCCCCATGTGGTTCGGCCAAAGAATGGTAGAGCGGAG
fuef      CAGCGCGAGTCTCGCCCCATGGGGTTAGGCCAAATAATGTTAGAGCGGCA
vier      TCGACTAGATCTGCAGCACATACGAGGGTCATGCGTCCCAGATGTG
fuefLoc2  TCGACTAGATATGCAGCAAATACGAGGGGCATGCGTCCCAGATGTG
```

Improved (new) format
The old format asks for the number of sites for each locus on the second line of the datafile and then needs all loci as consecutive blocks within each population. The new format still allows this old style but adds a new format that can take concatenated loci, one line per individual, To mark the new format the number of sites needs to be specified in a modified format, here a few examples:

```
OLDFORMAT 1 locus:  123
NEWFORMAT 1 locus:  (s123)
OLDFORMAT 3 loci:  123 195 2310
NEWFORMAT 3 loci:  (s123) (s195) (s2310)
NEWFORMAT 3 loci:  (s123), (s195), (s2310)
```

The last two examples show some of the difference to the old format, the example with the "," (comma) block the data like the old format, whereas the example just before uses the concatenated scheme, each individual will need 123+195+2310 sites, that are are then separated by the program into 3 independent loci because the "()" syntax suggests that these loci are independent. Other formats like "Brownian motion is specified with (b1), SNPs are specified with (n1) or 4 linked snps are specified with (n4). For allelic data the old format is preferred and the new format may still break for Brownian motion, stepwise, or allelic models (b, m, a).

Advancement of the new format
The main advantage of the new format allows to give a very large sequence that may or may not be a concatenated list of loci, that then can be split by the specification on the 'sites' line (the second line) in the datafile The new format is triggered when the first character on the sites line is a '(', fro example `(s50)`
marks a single sequence locus with 50 sites `(s20 s30)`
marks two LINKED sequence loci with 20 sites and 30 sites. The two loci are on the same line, for example (s3 s5) looks like this

```
2 2 Make believe data set using simulated data (1 locus) NEWFORMAT
(s3  s5)
3  Tallahassee
Peter     ACA CCCAA
Donald    ACA CAAAA
Christian ATA CCCAG
3  St. Marks
Lucrezia  ACA CCCAA
Isabel    ACA CAAAA
Yasmine   ATA CCCAG
```

`(s20) (s30)` marks two UNLINKED sequence loci with 20 sites and 30 sites, respectively.

The old format that specified earlier as two unlinked loci with 50 46 can now be written as `(s50), (s46)` observe the ',' that specifies that the loci are blocked like in the old format, if both loci would be in the sample line as in the example before, then it reads `(s50) (s46)`

Here are more examples: `(s100) (s50 s40) (s10)`
The first locus has 100 basepairs, the second is a compound out of two linked loci with 50 sites and 40 sites each, and third locus has 10 sites.

Currently MIGRATE is ill equipped to run dataset with large sequences (millions of base pairs) automatically without guidance by the user how to break up these into unlinked blocks. But there are several shortcuts for very large genomic sequences, for example assume that you have sequence data of a chromosome. You could want to run 100 loci with length 500 bp distributed over the whole genomic sequence. This would be possible by `[100o500] (s21000000)` and you will also need to specify on the first line that you have 100 loci. This will take the whole sequence of 21×10^6 sites and extract 100 regularly space loci each 500 bp long, the same could be achieved by specifying the location of the locus in the full sequence, using something like:
`(0s500) (210000s500) (420000s500) (630000s500) ....`

Instead of an ordered set of loci one can choose a randomly set of loci using
`[100r500] (s21000000)`
This allows to run different subsets of the data, currently there is no way to use this random site subset to do model comparison because there is possibility to force the same random set for different runs of MIGRATE.

SNP data

The SNP data uses the same nucleotide nomenclature as the sequence data. and the first format is the same as the sequence data but with only one site for unlinked SNPs or more than one site for linked SNPs see example, the datatype to use for this data is either 'N' for nucleotides or 'H' for HapMap. The very first letter forces as specific data model, if that first position is empty than the parmfile or the menu can specify the data type.

```
# using the old SNP data format
N 2 2 Make believe data set using simulated data (2 population and 2 loci)
1 4
3 3   pop1
ind1      A
ind2      A
ind3      A
ind1      ACAC
ind2      ACAC
ind3      ACGC
2    pop2
ind4      C
ind5      C
ind4      TGGA
ind5      TCGA
```

The HapMap format for the same data set looks like this:

```
# PRELIMINARY use this with care and let me know!
# using the HapMap data format, but does produce the same result (yet) as the dataset above
H 2 2 Make believe data set using simulated data (2 population and 2 loci)
3  pop1
1         A    3    C    0    3
1000      A    3    T    0    3
1010      C    3    G    0    3
1011      A    2    G    1    3
1015      C    3    A    0    3
2    pop2
1         A    0    C    2    2
1000      A    0    T    2    2
1010      C    1    G    1    2
1011      A    0    G    2    2
1015      C    0    A    2    2
```

Menu and Options

Most options can be changed through the textual menu.

You can change the options in the menu (Fig. 8) using letters or in submenus numbers. In menu entry `Data type` you need to specify what kind of data you have and according to that type some other menu entries appear, for example: transition/transversion ratio for sequences.

```
 ===========================================================
  POPULATION SIZE, MIGRATION, DIVERGENCE, ASSIGNMENT, HISTORY
  Bayesian inference using the structured coalescent
  ===========================================================
  Using Intel AVX (Advanced Vector Extensions)
  Compiled for a SYMMETRIC multiprocessors (GrandCentral)
  PDF output enabled [Letter-size]
  Version 4.0   [2022]
  Program started at   Tue Jul 15 11:37:15 2014


  Settings for this run:
  D       Data type currently set to: DNA sequence model
  I       Input/Output formats and Event reporting
  P       Parameters  [start, migration model]
  S       Search strategy
  W       Write a parmfile
  Q       Quit the program


  To change the settings type the letter for the menu to change
  Start the program with typing Yes or Y
===>
```

Figure 8: Top menu of *Migrate*

Menu options can also be changed in the `parmfile`, but before you do that, become more experienced with the menu and its interaction with the parmfile (make some changes in the menu, save the parmfile, and then check how these changes were translated. Never ever use an old parmfile from earlier versions to edit by hand, you will miss new options and also potential changes in the parmfile. If you want to use options of an older parmfile, load it into MIGRATE and save it using the menu option, and then manipulate the parmfile with a text editor. MIGRATE **will overwrite currently all user comments added to the parmfile.** All possible options are shown in `parmfile` syntax, but the same items can be changed in the menu as well. All entries in the `parmfile` are not case sensitive and all options can be given with the first letter, e.g. datatype=Allele is equal to datatype=A.

Data type

If you chose `D` in the main menu then will get the data menu (Fig. 9). More options will appear with some choices, for example when you have dated samples you can add a datefile and will also need to specify a mutation rate estimate (Fig. 10). These additional options are meaningless without dated samples and should only be used with that type of ancient DNA or virus datasets.

```
  DATATYPE AND DATA SPECIFIC OPTIONS

  D   change Datatype, currently:                DNA sequence model
  1   change Mutation model, currently:              Felsenstein 84
  2   Haplotyping is turned on:                                  NO
  5   One category of sites?                           One category
  6   One region of substitution rates?                         YES
  8   Sites weighted?                                            NO
 10   Sequencing error rate?             [0.000 0.000 0.000 0.000]
 11   Slow but safer Data likelihood calculation                 NO
 13   Inheritance scalar set                                     NO
 14   Pick random subset per population of individuals           NO
 15   Tip date file                   None, all tips a contemporary


  Are the settings correct?
  (Type Y or the number of the entry to change)
===>
```

Figure 9: Data menu

To change the data type select `1`, the other numbers show options that are relevant for the actual data type. There are several datatypes such as the following:
datatype=<Allele | **Microsatellites** | **Brownian** | **Sequences** | **Nucleotide-polymorphisms** | **HapMap-SNP** | **Genealogies** >
specifies the datatype used for the analyses, needless to say that if you have the wrong data for the chosen type the program will crash and will produce sometimes very cryptic error messages.

Allele: infinite allele model, suitable for electrophoretic markers, perhaps the "best" guess for codominant markers of which we do not know the mutation model.

Microsatellite: a simple electrophoretic ladder model is used for the change along the branches in genealogy.

Brownian: a Brownian motion approximation to the stepwise mutation model for microsatellites us used (this is **much** faster than exact model, but is not a good approximation if population sizes Θ_i are small (say below 10).

Sequences: Data are DNA or RNA sequences and the mutation model used is F84, first used by Felsenstein 1984 (actually the same as in `dnaml` (Phylip version 3.5; **?**), a description of this model can be found in **?**.

Nucleotide-polymorphism:[SNP] the data likelihood is corrected for sampling only variable sites. We assume that the a sequence data set was used to find the SNP. It is more efficient to run the full sequence data set.

```
  DATATYPE AND DATA SPECIFIC OPTIONS

  D   change Datatype, currently:                 DNA sequence model
  1   change Mutation model, currently:                    Tamura-Nei
  2   Haplotyping is turned on:                                    NO
  5   One category of sites?                             One category
  6   One region of substitution rates?     3 categories of regions
  7   Rates at adjacent sites correlated?    NO, they are independent
  8   Sites weighted?                                              NO
 10   Sequencing error rate?               [0.000 0.000 0.000 0.000]
 11   Slow but safer Data likelihood calculation                  YES
 13   Inheritance scalar set                                       NO
 14   Pick random subset per population of individuals              5
 15   Tip date file                                          datefile
 16   Mutation rate per locus and year                 0.000000100000
 17   How many generations per year                            1.0000


  Are the settings correct?
  (Type Y or the number of the entry to change)
===>
```

Figure 10: Data menu with more options that appear with dated samples, and site rate categories

HapMap-SNP:[SNP] the data likelihood is corrected for sampling only variable sites. We assume that the a sequence data set was used to find the SNP.

Genealogies: Reads the `bayesallfile` (see INPUT/OUTPUT section) of a previous runs, currently this option simply recreates the histogram, this allows the readjust some of the printouts but its usability to create new plots is limited.

Sequence data
If you specified **datatype=Sequence** the following options have some meaning and will show up in the menu (see also details for these options in the main.html and dnaml.html of the PHYLIP distribution
`http://evolution.gs.washington.edu/phylip.html`)

ttratio=$<$ **r1 r2 .....**$>$
you need to specify a transition/transversion ratio, you can give it for each locus in the dataset, if you give fewer values than there are loci, the last ttratio is used for the remaining loci $\rightarrow$ if you specify just one ratio the same ttratio is used for all loci.

freq-from-data=$<$ **Yes** $|$ **No:freqA freqG freqC freqT**$>$
freq-from-data=Yes calculates the base frequencies from the infile data, this will crash the program if in your data a base is missing, e.g. you try to input only transitions. The frequencies must add up at least to 0.9999.
freq-from-data=No:0.2 0.2 0.3 0.3 Any arbitrary nucleotide frequency can be specified.

sequence-error=$<$ $\{$**VALUE,VALUE,VALUE,VALUE**$\}|$**Estimate:**$1|4 >$
The number has to be between 0.00 and 1.00, default is 0.00, which of course is rather far

from the truth of about 0.001 (= 1 error in 1000 bases). The values are considered to be error rates for all sites and sequences. One can in principle estimate the error rate (for for all bases or 4 for each of the bases) through MCMC but this may not work well. Examples are
sequence-error={0.002,0.001,0.0004,0.005}
sequence-error=Estimate:1

categories=<Yes | No>
If you specify **Yes** you need a file named "`catfile` in the same directory with the following Syntax: number_of_categories cat1 cat2 cat3 .. categorylabel_for_each_site for each locus, a # in the first column can be used to start a comment-line. This option is very rarely used. Example is for a data set with 2 loci and 20 base pairs each

```
 # Example catfile for two loci
 # in migrate you can use # as comments
 2 1 10         11111111112222222222
 5 0.1 2 5 23 3 11111122223333445555
```

rates=< n : r1 r2 r3 ..rn>
by specifying rates a hidden Markov model is used for the sequences **?**, also see the PHYLIP documentation. In the `Menu` you can specify rates that follow a Gamma distribution, with the shape parameter `alpha` of that Gamma distribution, the program then calculates the rates and the rate probabilities (**p**rob-rates).

prob-rates=< n : p1 p2 p3 ... pn>
if you specify your own **rates** you need also to specify the probability of occurrence for each rate. MIGRATE is using, like PHYLIP, Laguerre quadrature points to find the discrete rates with their probability [in contrast to other programs that use discrete values at equal probabilities]

autocorrelation=<Yes:value | No>
if you assume hat the sites are correlated along the sequence, specify the block size, by assuming that only neighboring nucleotides are affected you would give a value=2. [this option may not work in version 4.x]

weights=<Yes | No>
If you specify **Yes** you need a file `weightfile` with weights for each site, the weights can be the following numbers 0-9 and letters A-Z, so you have 35 possible weights available.

```
# Example weightfile for two loci
11111111112222222222
1111112222AAAA445XXXX5
```

inheritance-scalars={value1, value2, ...} The inheritance scalar is relative to the locus that is set to 1.0. If that locus is a nuclear marker and the species is diploid then all Θ are equivalent to $4N_e\mu$, if that locus is a segment of mtDNA then all Θ are equivalent to $N_e\mu$ (maternal inheritance, sex ratio 1:1). If you have 3 loci, for example in this order: a nuclear marker, a mtDNA marker, and an X-linked marker then the input for this option is:
inheritance-scalars={1.0, 0.25, 0.75 }
This expresses all loci as $\Theta = 4N_e\mu$; A second example: if you have two loci, the first is Y-chromosome segment and the second is X-linked and you would want to express all in Θ_Y then
inheritance-scalars={1.0, 3.0 }

or if you want to express in Θ_X then
inheritance-scalars={0.333 1.0}
Use for the reference locus the scalar 1.0 and all other scalars relative to that.

random-subset=<**NO** | **number**> MIGRATE can randomly subsample each population. Picking the number specified in the **random-subset**. If the population sample has fewer individuals than the specified number, all samples are taken for that population.

tipdate-file= <NO | YES:datefile >
IF YOU HAVE ONLY CONTEMPORARY DATA DO NOT USE THIS OPTION.
The `datefile` contains sampling-dates for the individuals (the tips of the genealogy). An example is this: `tipdate-file=YES:datefile.bison3`
The datefile format is close to the `infile` format but for obvious content reasons not identical, in generalized form it looks like this:

```
 <Number of populations> <Number of loci>  <Title>
<Number of individuals>  <Population title>
<individual1 1-10>      <Date>
<individual2 1-10>      <Date>
<individual3 1-10>      <Date>
....
<Number of individuals>  <Population title>
<individual4 1-10>      <Dual Date>
<individual5 1-10>      <Dual Date>
....
```

The individual names MUST match the individual names in the `infile` and all names MUST be unique, this is a stringent requirement that is only needed when you use a datefile to guarantee that the right dates and sequences are matched.
The date must be given as a date measured backwards in time (dual time), so if a bison died 164 BC and you are able to extrac DNA from the bones then you should specify that the bison died 2172 years ago (in 2008), MIGRATE will adjust so that the smallest date will be set to date zero. Here an example using the mentioned syntax:

```
 2 1  Bison priscus dated samples
3  Alaska
a2172     2172
a2526     2526
a4495     4495
2 Siberia
s14605    14605
s23040    23040
```

In the example the dates are the years before present, but in principle they can be any units as lon as the mutation rate per 'year' and the generation-per-year is on the same scale.

mutationrate-per-year= {<mutationrate1>,<mutationrate2>,...}
For example: mutationrate-per-year={0.0000005}
IF YOU HAVE ONLY CONTEMPORARY DATA DO NOT USE THIS OPTION.
If you do not know the mutation rate, guess and try out to estimate the mutation rate in the analysis but depending on your data this may be a taxing analysis. For the moment use the mutation rate per generation and not year, see below.

generation-per-year= <value>
IF YOU HAVE ONLY CONTEMPORARY DATA DO NOT USE THIS OPTION.

The `datefile` needs additional information about the spacing of the samples in time, the number of generations per year helps to get this spacing, but we also need the mutation rate (see above). Example: `generation-per-year=1.000000`. Currently the generation time setting needs further tests, a generation time of 1.0 works, but other settings may fail; for the moment just use 1.0, and translate the results in years if needed.

Microsatellite data

Options that are used when the data are microsatellite repeat markers. MIGRATE uses repeat numbers internally, the infile can specify whether the data is in repeat numbers or in fragmentlength. MIGRATE does not use models that behave differently with very small or very large numbers of repeats, It assumes that the mutation rate for a change from, say, 5 repeats to 6 is the same as from 245 to 246.

Stepwise mutation model: If the **datatype=Microsatellite** is used, the following options have some meaning:

include-unknown=<YES | NO>
The default is bfNO. Alleles that are marked with a "?" are stripped from the analysis with include-unknown=NO. Using YES leaves the "?" in the analysis, under some circumstances this might be the preferred way, but for most situations the unknowns can be safely stripped from the analysis.

micro-threshold=value
specifies the window in which probabilities of change are calculated if we have allele 34 then only probabilities of a change from 34 to 35-44 and 24-34 are considered, the probability distribution is visualized in Figure 5 the higher this value is the longer you wait for your result, choosing it too small will produce wrong results. If you get -Infinity during runs of migrate then you need to check that all alleles have at least 1 neighbor fewer than 10 steps apart. If you have say alleles 8,9,11 and 35,36,39 then the default will produce a probability to reach 11 from 35 and as a result the likelihood of a genealogy will be -Infinity because we multiply over all different allele probabilities.
Default is **micro-threshold**=10

usertree=<NO | RANDOM >
The default is **NO** and MIGRATE calculates a starting tree using a UPGMA tree that uses a very simply distance matrix between the samples and then constrains this topology to follow a coalescent.

With the keyword **RANDOM** one can generates a random starting tree with "coalescent time intervals" according to the start parameters. This is generally a bad choice, but in conjunction of many short chains and the **replicate=YES:number** option [number is bigger than 1, see below]. This can help to search the parameter space more efficiently.

For these following options see under *Sequence data* above.

random-subset=<NO | number>

tipdate-file= <NO | YES:datefile >

mutationrate-per-year= {<mutationrate1>,<mutationrate2>,...}

generation-per-year= <value>

Brownian motion approximation: If the **datatype=Brownian** is used, the following options have some meaning:

include-unknown=<YES | NO>
The default is bfNO. Alleles that are marked with a "?" are stripped from the analysis with include-unknown=NO. Using YES leaves the "?" in the analysis, under some circumstances this might be the preferred way, but for most situations the unknowns can be safely stripped from the analysis.

usertree=<NO | RANDOM >
The default is **NO** and MIGRATE calculates a starting tree using a UPGMA tree that uses a very simply distance matrix between the samples and then constrains this topology to follow a coalescent.

With the keyword **RANDOM** one can generates a random starting tree with "coalescent time intervals" according to the start parameters. This is generally a bad choice, but in conjunction of many short chains and the **replicate=YES:number** option [number is bigger than 1, see below]. This can help to search the parameter space more efficiently.

For these following options see under *Sequence data* above.

random-subset=<NO | number>

tipdate-file= <NO | YES:datefile >

mutationrate-per-year= {<mutationrate1>,<mutationrate2>,...}

generation-per-year= <value>

Allozyme data

include-unknown=<YES | NO>
The default is bfNO. Alleles that are marked with a "?" are stripped from the analysis with include-unknown=NO. Using YES leaves the "?" in the analysis, under some circumstances this might be the preferred way, but for most situations the unknowns can be safely stripped from the analysis.

usertree=<NO | RANDOM >
The default is **NO** and MIGRATE calculates a starting tree using a UPGMA tree that uses a very simply distance matrix between the samples and then constrains this topology to follow a coalescent.

With the keyword **RANDOM** one can generates a random starting tree with "coalescent time intervals" according to the start parameters. This is generally a bad choice, but in conjunction of many short chains and the **replicate=YES:number** option [number is bigger than 1, see below]. This can help to search the parameter space more efficiently.

For these following options see under *Sequence data* above.

random-subset=<NO | number>

tipdate-file= <NO | YES:datefile >

mutationrate-per-year= {<mutationrate1>,<mutationrate2>,...}

generation-per-year= <value>

No special variables, but see **Parmfile specific commands**.

Nucleotide polymorphism
Similar to **sequence data**.

Input/Output formats

This group of options specifies input file names and various output file options. These options are somewhat depending on the analysis methods: Maximum likelihood approach (MLA, Fig. **??**) or Bayesian Approach (BA, Fig. 11). The numbering in the menus are not 1,2,3,4,... because I wanted to keep the same numbers for the options that are shared between the two approaches the same.

```
  INPUT/OUTPUT FORMATS
  -----------------------------------------------

  INPUT:
   1   Datafile name is                                  twoswisstowns
   2   Use automatic seed for randomisation?                       YES

  OUTPUT:
   5   Print indications of progress of run?                       YES
   6   Print the data?                                              NO
   7   Outputfile name is                                      outfile
                                                           outfile.pdf
  12   Print genealogies?                                         None
  15   Save logging information?                                    NO
  19   Show event statistics                   mighistfile (all events)
       Events are recorded every                     every sample step
       Histogram bin width                                    0.001000
  20   Record parameter change through time?               skylinefile
       Histogram bin width                                    0.001000


  Are the settings correct?
  (type Y to go back to the main menu or the letter for the entry to change)
===>
```

Figure 11: Input/Output menu of *Migrate*

Input formats

infile=filename

If you insist to have a datafile names other than `infile`, you can change this here, if you do not specify anything here, it will use any file with name `infile` present in the execution directory, if there is no `infile` than the program will ask for the datafile and you can specify the path to it (this may be hard on Macs and Wintel machines). If you use this option, do **N**OT use spaces or "/" or on Macs ":" in your filename. The default is obviously **infile=infile**

random-seed=<Auto | Noauto | Own:seedvalue>

The random number seed guarantees that you can reproduce a run exactly. I you do not specify the random number seed (**seed=Auto**) the program will use the system clock. With **seed=Noauto** the program expects to find a file named `seedfile` with the random number seed. With **random-seed=Own:seedvalue** you can specify the seed value in the parmfile (or in the menu).
Example for own seed:
random-seed=Own:21465 If you want reproducible runs you should replace the **Auto** seed with your own starting number (there are no requirement for the starting number perhaps except 0,] MIGRATE uses the Mersenne-Twister algorithm to generate random numbers). The default is **r**andom-seed=Auto. If you use **random-seed=Own:seedvalue** do nor forget to change the seed for different runs, otherwise the sequence of random numbers is always the same and the result will look the same on the same machine.

Caution: if you run MIGRATE in a simulation study you should set the random number yourself, the AUTO option might produce the same random number seed for runs that are started in the same second: this is quite common under batch-queue systems, when you run the same date from the same seed, you will get always the same result. I tried to improve this by getting a better seed automatically but this is somewhat machine dependent.

title=titletext

if you wish to add an informative title to your analysis, you can do it here or in the infile, the infile will override the title specified here. The length of the title is maximal 80 characters. Example: **title=Migration parameter estimation of populations A and B of species X**.

Output formats

progress=<Yes|No|Verbose> Show intermediate results and other hints that the program is running. Prints time stamps and gives a prognosis when the program eventually will finish, but this is a rather rough guide and sometimes gets fooled. An analogy, the system knows how far to drive and how far we have already driven and the time, but no clue about how many speed bumps (many migration events) and accidents are ahead of us.

Verbose adds more hints (at least for me) and information. The default is **progress=Yes**

print-data=<Yes|No>

Print the data in the `outfile`. defaults is **print-data=No**. If you run your data for the first time trhough MIGRATE turn this option on, because it helps to find problems with data-reading. Especially with microsatellite data it is possible that the program runs but the loci are incorrectly read.

outfile=filename

All output is directed into this file, the default name is outfile. If you use this option, do **NOT** use spaces or "/" or on Macs ":" in the filename. The default is obviously **outfile=outfile**

print-trees=<All | None | Last | Best>

print genealogies into `treefile`. Remember these trees contain migration events, `treeview` **?** and `FigTree` **?** can display such trees, although the migration events do not show on these displays, other program might crash. We have a program `eventree -- ET for short` that can display all the events on the tree, the program can be downloaded from the Migrate website.

None: `treefile` is not initialized and no trees are printed, this is the fastest and the one I recommend.

All: will print all trees (you want to do that only for ridiculously small datasets with too short chains or you have **many Gigabytes** of free storage).

Last: Only the trees of the last long chain are printed, Still you will need lots of space.

Best: Prints the tree with the highest data-likelihood for each locus. This is slow! And does not give a lot of information, except if you are more interested in the best tree for each locus than in the best parameter estimate.

Default is **print-trees=None**

logfile=<NO | YES:logfilename>

Records the output to the screen into a file when turned on, otherwise the screen output will be lost. On windows systems this may be the only option to see what is going on the because the screen buffer is only 80 lines.

mig-histogram=<NO | <ALL | MIGRATIONEVENTSONLY >:binsize:mighistfilename>

Records the frequencies of migration events (with MIGRATIONEVENTSONLY) or of all migration events and coalescence events (ALL) over time using *binsize*, the binsize is not optimal because you need to fix it before you know the range of times. A value 10 to 20$\times$ smaller than the average population size Θ is a good start. The output is a histogram of frequencies for each parameter, and a summary table of the average frequencies and a table of the frequency of the location of the root of the genealogy.

skyline=<NO | YES:binsize:skylinefilename>

If you have only contemporary data, do not trust this output. This options depends on the mig-histogram option, it uses the same binsize and needs some of its data structures, therefore do turn on the mig-histogram=ALL.... before attempt to use this option. With this option MIGRATE will present the changes of parameters through time, this method uses a different approach than BEAST and is may be more crude but can represent migration parameters and can summarize over multiple loci.

Start values for the Parameters

The Parameter menu allows to change the meaning of some of the parameters and allows to set start parameters

Start parameters

```
  PARAMETERS
  ---------------------------
  Start parameters:
  1   First parameter values are?
                           combined start-parameter report not finished

 Gene flow parameter and Mutation rate variation among loci:
  2   Use M for the gene flow parameter                  YES [M=m/mu]
  3   Mutation rate is                                       Constant

  Structured coalescent model and combination of localities:
  4   Sampling localities                                    default
  5   Model is set to                   Full migration matrix model
  6   Geographic distance matrix:                                  NO


  Are the settings correct?
  (Type Y to go back to the main menu or the letter for an entry to change)
===>
```

Figure 12: 'Start value for the parameter' menu of *Migrate*

theta=$<$Prior:{percentvalue}| Own:{value1,value2, ...} | Normal:{mean,std}| Uniform:{minimum, maximum} $>$

The menu option "*Use a simple estimate of theta as start?*" allows to specify a start value for the mutation scaled population size Θ. The 'prior' option is the default, it is set to 10 (=10% of the prior value), this seems to be OK for many data sets, but this depends strongly on your prior, for example if you set an expoential prior with bounds at 0.0 and 10^10 then start value may be crazy high for your data. Setting prior ranges very wide is usually a bad idea. This option is in principle not important because the MCMC run should be long enough so that the starting values do not matter. In praxis good values of start parameters allow much faster convergence than bad ones. Simulations have shown that starting from too low values typically increases the run-length considerably, whereas to high values seem more to help than hurt, although if the start values are very large and the data is not strong then MA can fail without a clear signal of failure; MIGRATE will return a large parameter estimate that does not reflect the data very well. BA is much less vulnerable to this problem. The start genealogy depends on the start parameters because even with a random topology the times are constrained to come from a coalescence process with parameters set equal to the start parameters defined here.

migration= $<$ Prior:{percentvalue}|Own:Migration matrix | Normal:{mean,std}| Uniform:{minimum, maximum} $>$

The menu option *Use a simple estimate of migration rate as start?* allows to specify a start value for the migration parameter. The 'prior' option is the default, it is set to 10 (=10% of the prior value), this seems to be OK for many data sets, but this depends strongly on your prior, for example if you set an expoential prior with bounds at 0.0 and 10^10 then start value may be crazy high for your data. Setting prior ranges very wide is usually a bad idea. The values for **Own** are given in terms of $4N_em$ which is 4 $\times$ effective population size $\times$ migra-

tion rate per generation. The default is **migration=FST**. The **migration matrix** is a n by n table with - on the diagonal and can look like this for four populations `migration=OWN:{ - 1.0 1.1 1.2 0.9 - 0.8 0.7 2.1 2.2 - 2.3 1.4 1.5 1.6 - }`
or like this

```
migration=OWN:{ -   1.0 1.1 1.2
                0.9 -   0.8 0.7
                2.1 2.2 -   2.3
                1.4 1.5 1.6 - }
```

See note on start values above under Θ.

Gene flow parameter and mutation rate variation among loci

The gene flow parameter can be presented as xNm or M. M is the mutation scaled immigration rate m/μ that represents the importance of variability brought into the population by immigration compared with the variability created by mutation, m is the fraction of the new immigrants of the population per generation. xNm represents the number of immigrant per generation scaled by x where x depends on the data: $x = 1$ for haploid, uniparental inheritance (mtDNA, Y), $x = 2$ for haploids (bacteria), $x = 3$ for the X chromosome in the mammal X-Y systems, $x = 4$ for diploid organisms (nuclear DNA), etc.

use-M=<YES | NO>

mutation=<NoGamma | Constant | Estimate | Gamma:alpha| Varying | Relative >

If there are more than one locus the program averages the parameter distributions over all loci (this is different from the average of the most likely parameter values, loci that contribute more peaked parameter distributions are weighted more heavility than parameter distributions that have very flat distributions.]). The mutation rate over all loci can be manipulated in a couple of ways, This options should not be used for first trials with MIGRATE. The menu presents you with these choices:

```
(C)onstant   All loci have the same mutation rate [default]
(E)stimate  Mutation rate
(V)arying     Mutation rates are different among loci [user input]
(R)elative    Mutation rates estimated from data
```

The **Estimate** flag ("estimate mutation rate") allows for the variation of the mutation rate of each locus proposing new mutation rate values from the prior distribution. For most dataset this will not work, because you will need date samples. The options **Varying** allows you to input your own mutation rate modifier. MIGRATE is modifying your values so that the average rate will be 1.0.

The option **Relative** estimates a rough rate modifier for the mutation rate using the data. For sequence data the Watterson estimator is used to get relative rates, this takes into account the different numbers in the sample. For microsatellite and allozyme the allele counts are used to generate a rough value of the rate for each locus. These rates average to 1.0.

With **Nogamma** or **Constant** no special calculations are done. The summarizing step is simply finding the best parameters by maximizing the sum of the log-likelihoods of each locus. The default is **mutation=Nogamma**

Migration model

If you do not specify anything the posterior probability densities of all $n \times n$ parameters are found.

custom-migration=<NONE | migration-matrix>
The migration matrix contains the migration rates from population j to i on row i, and the Θ are on the diagonal. The migration matrix can consist of connections that are

- 0: not estimated
- m: mean value of either Θ or $\mathcal{M}$.
- s: symmetric migration [symmetric $\mathcal{M}$ not xNm]
- S: symmetric migration [symmetric xNm not $\mathcal{M}$]
- c: constant value (toghether with `migration=OWN..` or `theta=OWN..`)
- *: no restriction

(the x means a multiplier: 4 for diploid nuclear markers, 2 for haploid markers, 1 for haploid markers transmitted only through one sex, such as mtDNA and Y chromosomes)

The values can be spaced by blanks, newlines A few examples for 4 populations:

Full model: **custom-migration={******
****
****
******}**
N-island model: **custom-migration={m m m m**
mm mm
m mmm
mmmm}
Stepping Stone model with symmetric migrations, and unrestricted Θ estimates:
custom-migration={*s00 s*s0 0s*s 00s*}
Source-Sink (the first population is the source (Figure 13)):

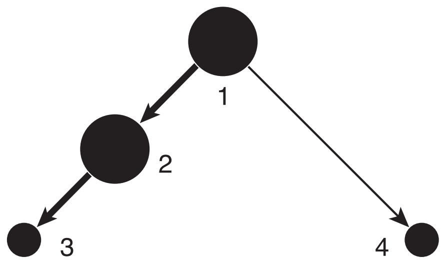

Figure 13: Source-sink example

custom-migration={*000000**0*00*}**

Geographic distance between locations

You can specify a distance matrix between your populations. the distance file has the same syntax as a PHYLIP distance file [see example below].

geofile=<NO | YES:filename>
The distance matrix contains the distances between pairs of populations, if you choose for example distance units in kilometers you will get migration rate estimates that are scaled as M = immigration rate / (mutation * kilometer), if you restrict the migration rate to an average value for all connections between population you are calculating a dispersion coefficient based on discrete populations. This coefficient should be in the limit the same as the one calculated from a isolation by distance population model.
There is no requirement that the distances from i to j are the same as from j to i, although interpretation might be difficult with an unequal distance matrix. the default filename for this distance file is *geofile*.

Example *geofile*

```
     3
Ermatingen0.0 10.4 12.4
Schachen  10.4 0.0 1.0
Heiden    12.4 1.0 0.0
```

Search strategy

This section is the key to good results and you should not just use the defaults, for guidance how I myself would do this check out the section **how long to run**.

Maximum likelihood inference

```
 SEARCH STRATEGY

 0   Strategy:                         Maximum Likelihood
 1   Number of short chains to run?                    10
 2   Short sampling increment?                         20
 3   Number of recorded genealogies in short chain?   500
 4   Number of long chains to run?                      3
 5   Long sampling increment?                          20
 6   Number of recorded genealogies in long chain?   5000
 7   Number of genealogies to discard at
     the beginning of each chain? [Burn-in]         10000
 8   Combine chains or runs for estimates?             NO
 9   Heating:      YES (  4 chains, swap interval is   1)

-------------------------------------------------------------
Obscure options (consult the documentation on these)

10   Sample at least a fraction of new genealogies?    NO
11   Epsilon of parameter likelihood             infinity
12   Use Gelman's convergence criterium?      YES:Summary


 Are the settings correct?
 (Type Y to go back to the main menu or the number for a menu to change)
===>
```

Figure 14: 'Search strategy' menu with the Maxim likelihood approach

The terminology of **short** or **long** chains is arbitrary, actually you could choose values so that short chains are longer than the "long" chains. Anyway, Markov chain Monte Carlo (MCMC) approaches tend to give better results when the start parameters are close to the maximum likelihood values. One way to achieve this is running several short chains and use the result of the last chain as starting value for the new chain. This should produce better and better starting values, if the short chains are not too short.

Strategy
With version 2.0 you have a choice of either using a maximum likelihood procedure or a Bayesian approach, on nice data both method will work about the same, for some example runs it seems that the profile likelihoods and the Bayesian posterior distribution agree quite fine on the distribution of the parameter value. The options specific to the Bayesian approach are explained in the next section.

Number of short chains to run? (short-chains=value
we run most of the time about 10 short chains, which is enough if the starting parameters are not too bad. Default is **short-chains=10**.

Short sampling increment? (short-inc=value)
The sampled genealogies are correlated to reduce the correlation between genealogies and to allow for a wider search of the genealogy space (better mixing), we sample not every genealogy, the default is **short-inc=20** means that we sample a genealogy and step through the next 19 and sample then again.

Number of steps along short chains? (short-steps=value)
The default number of genealogies to sample for short chains is 500. But this may be to few genealogies for your problem. If you big data sets it needs normally bigger samples or higher increments to move around in the genealogy space.

Number of long chains to run? (long-chains=value)
I run most of the time 3 long chains. The first equlibibrates and the last is the one we use to estimate the parameters. Default is **long-chains=3**.

Long sampling increment? (long-inc=value)
The default is the same as for short chains.

Number of steps along long chains? (long-steps=value)
The default number of genealogies to sample for long chains is 5000. I often choose the "long" chains about 10 times longer than the "short" chains.

Number of genealogies to discard at the beginning of each chain? (burn-in=value)
Each chain inherits the last genealogy of the last run, which was created with the old parameter set. Therefore the first few genealogies are biased towards the old parameter set. When **burn-in** is bigger than 0, the first few genealogies in each chain are discarded. The default is **burn-in=10000**.

Combine chains for estimates The use of this option is recommended for difficult data sets. It allows to combine multiple chains for the parameter estimates when you use **replicate=YES:LongChains**. With **replicate=YES:number** where number is, well, a number bigger than 1. (e.g. replicate=Yes:5), you run the program "number" times and the results of their last chains are combined, The method of combination of chains is the same as in **?** and is based on the work by **?**. The LongChain option does not need much more time than the single chain option, but the full replication needs exactly "number" times a normal run. But is sampling the search space much better than any other option, I use this often in conjunction with random starting trees (randomtree=YES).

Heating (heating=<NO | YES | ADAPTIVE <:waitnumber:{cold,warm,hot,boil,....}>
This allows for running multiple chains and swap between them, when these chains are run at different temperatures, the "hotter" chains explore more genealogy space than the "cold" chains. An acceptance-rejection step swaps between chains so that the the "cold" chains will sample from peaks on the genealogy surface proportional to their probability. This scheme is known as MCMCMC (Markov coupled Markov chain Monte Carlo), it is based on the work of **?** and uses for four or more chains at different temperatures, the hotter chains move more freely and so can explore other genealogies, this allows for an efficient exploration of data that could fit different genealogies, and should help to set the confidence intervals more correctly

than a single chain path could do. You need to set the temperatures yourself because there is no default. The ADAPTIVE heating scheme manipulates these temperatures according to their swapping success. If a neighboring temperature pair is not swapping after 1000 trials the temperature difference between them is lowered by 10%, if a pair is swapping more than 10 times in a 1000 the gap is increase by 10% (these values are arbitrary, but cannot be changed in the menu, yet). Adaptive heating is definitely no cure-it-all, I typically prefer the static heating, but it helps find good values to try for the static heating scheme, I have seen pathological behavior of long adaptive runs where all chains essentially converged to values very very close to 1.0 (the cold chain) and stopped swapping.

If you use a STATIC heating scheme then you need to experiment a little because you want that the different chains swap once in a while, but not too often and certainly more often than never. The swapping seems to depend on how good the data describes a given genealogy. I would start with 4 chains and temperatures that are {**1. 1.5 3.0 10000.0**}. The temperatures are ordered from cold to boiling, the coldest temperature **MUST** be 1 (one). The default for the heating option is **heating=NO**. If you use this option sampling will be at least 4 times slower, except if you have a multiprocessor machine and a POSIX compliant thread-library (often called with slight variations but containing word parts such as pthread, thread, linuxthread), then you can compile the program using "make thread", this will improve speed somewhat, but lately I do not gain more than 170% CPU usage out of this. It is probably easier and faster to use all cores on new computers using the parallel version of MIGRATE.

The **waitnumber** is the number of trees to wait before the differently heated chains are check whether to swap or not. I normally use 1. I have little experience whether, say, using 10 improves mixing over using 1.

Obscure options

If you are not experienced with MCMC or run *Migrate* for the first, second, ... time, do not bother about the options here.

Sample at least a fraction of new genealogies? (moving-steps=<Yes:ratio | No >)

With some data the acceptance ratio is very low, for example with sequence data with more than 5000 bp the accpetance ratio drops below 10% and one should increase the length of the chains. One can do this either by increasing the **long-inc**, or **long-steps** or by using **moving-steps**. The ratio means that at least that ratio of genealogies specified in **long-steps** have to be new genealogies and if that fraction is not yet reached the sampler keeps on sampling trees. In unfortunate situation this can go on for a rather long period of time. You should always try first with the default **moving-steps=No**. An example:
You specified **long-steps=2000**,and **long-inc=20** and the acceptance-ratio was only 0.02, you have visited 40,000 genealogies of which only 800 are new genealogies so that you have maximally sampled 800 different genealogies for the paramter estimation. In a new run you can try **moving-steps=Yes:0.1**, the sampler is now extending the sampling beyond the 40000 genealogies and finally stopping when 4000 new genealogies were visited.

Epsilon of parameter likelihood (long-chain-epsilon=value)

The likelihood values are ratios

$$\frac{L(\mathcal{P})}{L(\mathcal{P}_0)} = \frac{1}{n}\sum_i \frac{\text{Prob }(G_i|\mathcal{P})}{\text{Prob }(G_i|\mathcal{P}_0)} \qquad \text{(Beerli and Felsenstein, 1999)}$$

When the Likelihood values are very similar then the ratio will be close to 1, or 0 when we use logarithms. This means that the sampler is not improving drastically between chains: (a) it found the maximum likelihood estimate or (b) it is so far from the maximum likelihood estimate that the surface is so flat that all likelihood values are equally bad. using a smaller value than the default **long-chain-epsilon=100.00** for example a value of 1.0 would guarantee that the sampler keeps on sampling new long chains as long as that log-likelihood-difference drops below 1.0. In some cases this will never happen and the program will not stop.

Gelman's convergence criterium If you specify "Yes" then the number of last chains get extended until the convergence criterium of Gelman is satisfied (the ratio should be smaller than 1.2 for **all** parameters. This can take a very long time. [In the parallel version this fails, turn it off there [this is a bug, but I had not time to find and fix it]).

Bayesian method

```
 SEARCH STRATEGY

   0   Strategy:                                    Bayesian Inference
   1   File for recording posterior distribution?                   NO
   2   File for recording all parameter values?                     NO
   3   Number of bins of posterior [Theta,M]?                 200, 200
   4   Plotting type of posterior distribution? up to ~100% percentile
   5   Frequency of tree updates vs. parameter updates?           0.50
   6   Proposal distribution?        Theta:Slice Mig:Slice Rate:Slice
   7   Prior distribution?           Theta:Unif. Mig:Unif. Rate:Unif.
   8   Number of long chains to run?                                 1
   9   Sampling increment?                                          20
  10   Number of recorded steps in chain                          5000
  11   Number of steps to discard at
       the beginning of chain? [Burn-in]                         10000
  12   Running multiple replicates:                                 NO
  13   Heating:                          STATIC (  4 parallel chains)
  14   Sampling at least fraction of new genealogies:         0.000000
  15   Convergence diagnostic for replicates:              YES:Summary


  Are the settings correct?
  (Type Y to go back to the main menu or the number for a menu to change)
===>
```

Figure 15: 'Search strategy' menu with the Bayesian approach

File for recording parameters? (bayesfile=<NO | YES:bayesfile>) this file contains the raw histogram for all parameters and all loci and their combination, figure 16 shows the first few lines of an example, see under section **Bayesian posterior explained** further uses of this file.

```
# Raw data for the histogram of the posterior probabilities for all parameters
# and loci  produced by the program migrate-n 2.0.3
# (http://evolution.gs.washington.edu/lamarc/migrate.hml)
# written by Peter Beerli 2004, Tallahassee,
# if you have problems email to beerli@csit.fsu.edu
#
# The HPC values are indicators whether the parametervalue is in the
# highest-posterior credibility set, a 0 means it is outside and a 1 means
# the value is inside the credibility set.
#
# Delta for Theta and M 0.001000 0.001000 9.995000 9.995000
# ---------------------------------------------------------------
# Locus Parameter 50%HPC  95%HPC (parameter-value count) frequency
# ---------------------------------------------------------------
1 1 0 0 0.002499 327 0.001635
1 1 0 0 0.003498 1634 0.008169
1 1 0 1 0.004498 4612 0.023058
1 1 0 1 0.005497 8970 0.044846
1 1 1 1 0.006497 13576 0.067874
1 1 1 1 0.007496 17320 0.086592
1 1 1 1 0.008496 19492 0.097451
1 1 1 1 0.009495 20537 0.102676
1 1 1 1 0.010495 19504 0.097511
```

Figure 16: First few lines of a bayesfile: the header explains the columns

File for recording all parameter values? (bayes-allfile=<NO | YES:number:bayesallfile>)
this file contains the raw histogram for all parameters and all loci and their combination, figure 17 shows the first few lines of an example, see under section **Bayesian posterior explained** for further uses of this file. This file can be very large depending on your options, it is still hard to work with files larger than 10 GB, so choose you settings carefully, there will be samples×loci×replicates sets of n^2 parameters and some additional values. If you need more samples to get good results and your data is highly autocorrelated increase the long-inc options (see there). If you specify this option (recommended) the memory foot print of the program is smaller than when this option is set to NO. This is important particularly for the parallel MIGRATE runs.

Number of bins of posterior (bayes-posteriorbins=<thetabins Mbins <ratebins>>)
The number of bins for the posterior needs to be pre-specified (to save memory). The default for Θ, M is 200 bins. This number is probably to small if the range of the prior distribution is very large. If the PDF histograms look course rerun after increasing the binsizes. The ratebins are used when the mutation rate modifier with many loci is estimated in the Bayesian analysis, this may sometimes fail, because there is little information about rate differences among loci in some datasets.

Plotting bins of posterior (bayes-posteriormaxtype=<TOTAL | P100| P99| MAXP99 >)
The posterior distribution often covers only a short range of the prior distribution, therefore

```
# Migrate debug 3.0 (Peter Beerli, (c) 2008)
# Raw results from Bayesian inference: these values can be used to generate
# joint posterior distribution of any parameter combination
# Writing information on parameters (Thetas, M or xNm)
# every 2 parameter-steps
#
# --  Steps
# --  Locus
# --  Replicates
# --  log(Posterior)
# --  log(prob(D|G))
# --  log(prob(G|Model))
# --  log(prob(Model))
# --  Sum of time intervals on G
# --  Total tree length of G
# Order of the parameters:
# Parameter-number Parameter
#@      1    Theta_1
#@      2    Theta_2
#@      3    M_(2,1)
#@      4    M_(1,2)
#
# --  Thermodynamic temperature = 1.000000
# --  Thermodynamic temperature = 1.500000
# --  Thermodynamic temperature = 3.000000
# --  Thermodynamic temperature = 1000000.000000
# --  Marginal log(likelihood) [Thermodynamic integration]
# --  Marginal log(likelihood) [Harmonic mean]
#
#$ -------------------------------------------------------------------
#$ begin  [do not change this content]
#$ Model=****
#$ Mode2=****
#$ 1 2 4 0 1 1
#$ pop00
#$ pop01
#$ end
#$ -------------------------------------------------------------------
#
# remove the lines above and including @@@@@, this allows to use
# Tracer (http://tree.bio.ed.ac.uk/software/tracer/) to inspect
# this file. But be aware that the current Tracer program (October 2006)
# only works with single-locus, single-replicate files
# The migrate contribution folder contains a command line utility written
# in PERL to split the file for Tracer, it's name is mba
# @@@@@@@@
#Steps  Locus   Replicate       lnPost  lnDataL lnPrbGParam     lnPrior treeintervals   treelength      Theta_1 Theta_2 M_2_1   M_1_2
100     1       1       -22365.119577   -22620.671234   255.551656      -17.034386      95      0.092424        0.00449 ...
200     1       1       -22367.961876   -22622.328216   254.366340      -17.034386      95      0.093002        0.00379 ...
300     1       1       -22368.867271   -22618.681322   249.814051      -17.034386      95      0.092687        0.00460 ...
```

Figure 17: First few lines of a bayesallfile: the header explains the columns, the data section is truncated at the right and bottom

displaying the **TOTAL** range of the prior distribution is often not advised, the P99 presents 99% of the posterior distribution, cutting of 1% of the posterior, this is a good way to visualize posterior distributions with very long (thin) right tails. P100 takes 99.99% of the values and excludes strange outliers. MAXP99 is cutting of at 99% credibility, but using the parameter with the highest value for Θ, and M, in principle this forces the same scale in the output for the parameters (this needs more testing because I most often use P100).

Frequency of tree updates versus parameter updates (bayes-updatefreq=$<$ value $>$)
The *value* specifies the ratio of genealogy updates and parameter updates, 0.5 means that the genealogy is updated roughly every second time, and one of the parameters is updated every second time. A value of 1.0 means that the parameters are never updated, A value of 0.0 is not advised because the genealogy does not adjust the migration events and so does not really test the parameter distribution for a specific tree.

Proposal distribution
bayes-posterior=$<$ $<$ THETA $|$ MIG $|$ RATE $>$ $<$ SLICE $|$ METROPOLIS $>$ $>$)
There are two ways the generate posterior distributions: SLICE and METROPOLIS. METROPO-

LIS is using the standard Metropolis-Hastings algorithm that proposes a new state not taking into account the data and then accepting or rejecting using the fit if the data to the old and new state. For some data the rejection rate is very high and many computer cycles are wasted because the MCMC chain does not move. SLICE sampling uses the current posterior distribution (taking into account the data) to generate a new state, every new state is compatible with the data, therefore the acceptance ratio is always 1.0. This comes at a price because the calculations are more demanding than the MH algorithm, and therefore may be slower. On data with lots of information SLICE sampling is great, but fails with poor data. SLICE is the default in MIGRATE.

Examples:

```
bayes-proposals= THETA SLICE Sampler
bayes-proposals= MIG SLICE Sampler
bayes-proposals= RATE SLICE Sampler
```

Prior distribution

bayes-priors=< < THETA | MIG | RATE > < PRIORSPECIFICATION > >)
There are several prior distribution available, but the list is still short. For each prior distribution you need to specify additional parameters:

Distribution	parameter 1	parameter 2	parameter 3	parameter 4
Uniform	Minimum	Maximum	Window size	-
Exponential	Minimum	Mean	Maximum	-
Windowed Exponential	Minimum	Mean	Maximum	Window size

Examples:

```
bayes-priors= THETA EXPPRIOR: 0.000000 0.250000 0.500000
bayes-priors= MIG WEXPPRIOR: 0.000000 500.000000 1000.000000 100.000000
bayes-priors= RATE UNIFORMPRIOR: 0.010000 100.000000 5.000000
```

Number of long chains to run? (long-chains=<value>)
Use 1 long chain because multiple long chains will do little to help the analysis, if you want to combine over replicated runs use the replicate option.

Sampling increment? (long-inc=<value>)
Samples are taken every *value* cycle, the default is 20.

Number of recorded genealogies in chains? (long-steps=value)
The default number of genealogies to sample for long chains is 50000. With the default increment this means 1,000,000 genealogies will be visited This is short for many datasets.

Number of genealogies to discard at the beginning of each chain? (burn-in=value)
The chain is not equilibrated at the beginning of the run, and we discard those aberrant values and trees. The default is **burn-in=10000**.

Combine chains for estimates The use of this option is recommended for difficult data sets. It allows to combine multiple chains for the parameter estimates when you use **replicate=YES:number**; where the number is bigger than 1. (e.g. replicate=Yes:5). You run the program "number" times and the results are combined (similar to MrBayes). The option

replicate=YES:number works for both Bayesian inference and Maximum Likelihood. The work is number times more than without replication. For ML there is another option: **replicate=YES:LongChains** that allows the combination of long chains only, this is the same method as used by **?** and is based on the work by **?**. The LongChain option does not need much more time than the single chain option.

Replication allows a better sampling of the search space than the single chain. in particular when used on parallel cluster computers. I use this often in conjunction with random starting trees (randomtree=YES).

Heating (heating=<NO | YES | ADAPTIVE <:waitnumber:{cold,warm,hot,boil,....}> This allows for running multiple chains and swap between them, when these chains are run at different temperatures, the "hotter" chains explore more genealogy space than the "cold" chains. An acceptance-rejection step swaps between chains so that the the "cold" chains will sample from peaks on the genealogy surface proportional to their probability. This scheme is known as MCMCMC (Markov coupled Markov chain Monte Carlo), it is based on the work of **?** and uses for four or more chains at different temperatures, the hotter chains move more freely and so can explore other genealogies, this allows for an efficient exploration of data that could fit different genealogies, and should help to set the confidence intervals more correctly than a single chain path could do. You need to set the temperatures yourself because there is no default. The ADAPTIVE heating scheme manipulates these temperatures according to their swapping success. If a neighboring temperature pair is not swapping after 1000 trials the temperature difference between them is lowered by 10%, if a pair is swapping more than 10 times in a 1000 the gap is increase by 10% (these values are arbitrary, but cannot be changed in the menu, yet). Adaptive heating is definitely no cure-it-all, I typically prefer the static heating, but it helps find good values to try for the static heating scheme, I have seen pathological behavior of long adaptive runs where all chains essentially converged to values very very close to 1.0 (the cold chain) and stopped swapping.

If you use a STATIC heating scheme then you need to experiment a little because you want that the different chains swap once in a while, but not too often and certainly more often than never. The swapping seems to depend on how good the data describes a given genealogy. I would start with 4 chains and temperatures that are {**1. 1.5 3.0 10000.0**}. The temperatures are ordered from cold to boiling, the coldest temperature **MUST** be 1 (one). The default for the heating option is **heating=NO**. If you use this option sampling will be at least 4 times slower, except if you have a multiprocessor machine and a POSIX compliant thread-library (often called with slight variations but containing word parts such as pthread, thread, linuxthread), then you can compile the program using "make thread", this will improve speed somewhat, but lately I do not gain more than 170% CPU usage out of this. It is probably easier and faster to use all cores on new computers using the parallel version of MIGRATE.

The **waitnumber** is the number of trees to wait before the differently heated chains are check whether to swap or not. I normally use 1. I have little experience whether, say, using 10 improves mixing over using 1.

Sampling at least fraction of new genealogies (moving-steps=<NO | YES:value >) This allows to specify that a minimum number of different genealogies need to be sampled, it is expressed as the ratio of sampled genealogies. If the frequency is not reached at the end of the specified number of samples, MIGRATE will continue until the ratio is satisfied, with high numbers the program may run forever. I rarely use this option.

Convergence diagnostic for replicates (gelman-convergence=< NO | YES:<Sum | Pairs > > This collects information about the convergence rate of two replicated chains (use two or more replicates). *Sum* reports the an average value over all whereas *Pairs* using the pairs of replicates to. Version 3.0 has some difficulties with this option and I hope to fix this in the next minor release, but on some machine and under some conditions the diagnostic fails.

Parmfile specific commands

Important parmfile options

menu=<Yes|No>
defines if the program should show up the menu or not. The default is **menu=Yes**.

end
Tells the parmfile reader that it is at the end of the parmfile.

Options to change the lengths of words and texts

If you change these, you should understand why you want to do this.

nmlength=number
defines the maximal length of the name of an individuum, if for a strange reason you need longer names than 10 characters (e.g. you need more than 10 chars to characterize an individual) and you do not need this very often then set it to a higher value, if you have no individual names you can set this to zero (0) and no Individual names are read. the default is **nmlength=10**, this is the same as in PHYLIP.

popnmlength=number
Is the length of the name for the population. The default is **popnmlength=100**

allelenmlength=number
This is only used in the infinite allele case. Length of an allele name, the default should cover even strange lab-jargons like `Rvf` or `sahss` (*Rana ridibunda* very fast, *Rana saharica* super slow) The default is **allelenmlength=6**

How to run MIGRATE

If you have compiled and installed the program successfully (see Installation) and your data is in a good format (section data format) and perhaps has the name infile, just execute

Command	Parameters	Comments
`migrate-n`		No option will take the default `parmfile` if present
`migrate-n`	parmfile.test	opens the file `parmfile.test` if present otherwise creates a new file that can be save through the menu
`migrate-n`	parmfile.test -menu	forces the program to show the menu
`migrate-n`	parmfile.test -nomenu	forces the program to NOT show the menu and start running immediately (use the -nomenu option for batch scripts and batch queue system.

On some systems you need to call MIGRATE using `./migrate-n`.

On most graphical systems you can start MIGRATE by double-clicking its icon, but the results are different among the different computer systems (Linux, MacOS 10, Windows). On Macs home directory and that is most likely not the location where your files sit. It is actually easier to open the Terminal.app (in /Applications/Utilities) and learn a couple of shell commands (a minimal set of cd, mv, cp will probably do for a start) (see for example this online tutorial http:). Within the terminal window you change to the directory with the data and then execute the program that either is in the same folder using the commands above. For windows double-clicking opens also a terminal window that is located at the same directory location as the icon, if your data is also in that same location your are set, but you can use the "Run..." command from the Startup menu to open a terminal window and then use chdir, copy, rename to operate the windows shell similarly to the UNIC shell.

Without any **parmfile**, *Migrate* will display a menu, in which you can change all the sensible options. For hints how to use the parmfile, look into section **Menu and Options** or the `parmfile`. Once you know how to customize the options with the **parmfile** you will probably more often edit the parmfile than making the changes in the menu. Be careful, some complex options are most easily set through the menu.

Bayesian inference

From a practical viewpoint we can think of Bayesian inference of a combination of knowledge: the prior knowledge and the knowledge gained through the data and model. The prior knowledge is used to to treat the parameters of interests as random variables with a distribution that is typically independent of our investigation, the prior distribution. The posterior distribution is the product of the prior distribution and the probability of the data given the parameters (the likelihood). The prior distribution needs to cover the interesting part of the range of the parameters, essentially the posterior distribution should fit within the range of the prior distribution. It is important to inspect the posterior for probably truncation by the prior, if that occurs ou should rerun the analysis. If the prior is much larger than the parameter region of interest, in many circumstances the analysis will take a long time because most values proposed from the prior do not fit well with the data and are rejected.

Prior distribution

Currently there are three distributions available: uniform, exponential with boundaries, exponential with boundaries and windowing:

Uniform prior You need to specify a lower and an upper bound, this prior distribution is similar to other programs, such as those by Hey and Nielsen (2004). Uniform assume that all parameter values are equally likely, an assumptions that often is not justified.

Exponential prior with boundaries This proper prior distribution (it integrates to 1) needs a lower and and an upper bound and a mean, if one specifies 0.0 and ∞ as boundaries, this distribution is the same as a simple exponential distribution. Preliminary runs show that this distribution is superior (aka converges faster) than the uniform distribution prior. Typically the boundaries are chosen so that there is a large set of possible values in between, the method is picking randomly in this range and so from one step to the next large differences in parameter values can occur, this large differences might lead to a larger rejection rate.

Exponential prior with boundaries and window Same as exponential prior with boundaries except that you need to specify an additional parameter that specifies the window size in from which changes in parameters are drawn. The chain will less often reject parameter values because they will be closer to the last value. This prior distribution seems to produce the best results so far, but it needs some fidgeting with the window. If the window is too small very long chains need to be run to explore the whole distribution, if the window is too large than the method reduces to the exponential prior with boundaries.

Proposal distribution: Slice sampling versus Metropolis-Hastings sampling

MIGRATE allows to use two different proposal functions for the evaluation of the parameters, but only one for the evaulation of genealogies: Metropolis-Hastings sampling **?** and Slice sampling (**?**). Metropolis-Hastings (MH) sampling is standard in most applications in population genetics, but Slice-sampling is not. I know that Paul Lewis (University of Connecticut) is working on a phylogeny program that uses slice-sampling but I have not see the program in the wild. Paul helped me to understand the slice-sampling method in 2006 at the molecular evolution workshop in Woods Hole. Slice sampling uses the data and the prior distribution to choose a new prior value, because the data already is comaptibel with this new value a MH-rejection step is not necessary and the new value is always accepted. In contrast to that, MH-sampling picks a value from the prior and then uses the data later in the rejection-acceptance step to accept or reject the new value. Experiments have shown that slice-sampling converges typically faster and produces smoother posterior distributions with less steps on the MCMC chain.

Posterior distribution

MIGRATE prints a file *bayesfile* that contains the raw histogram values for all parameters, the columns in that file allow to use graphing program such as GNUPlot (http://www.gnuplot.info/) to plot the distribution. My favored program to plot such graphs is GMT (General mapping tool, http://gmt.soest.hawaii.edu/) that produces postscript output, in the contribution directory I added a shell script (sorry, no MS Windows utility) that uses GMT and that produces posterior distributions that display the 95% credibility set.

MIGRATE also allows to save the raw parameter values that are used for the posterior distribution (*bayesallfile*). This file contains all information necessary to recreate the posterior histograms from scratch, This file also is compatible with the program TRACER (**?**) when you analyze a single locus without replication. There is a command line utility in the contribution directory. This utility *mba* allows to separate the large bayesallfile into files per locus and replicates. It also allows the assembly of different files, that can then be feed back to MIGRATE to recreate the posterior histograms. If you run MIGRATE on a cluster in parallel use turn on this option because the memory footprint of MIGRATE is much smaller than when this option is turned off.

Prior distributions: choice and problems

to come

Model selection

[This section is not finished] MIGRATE allows to calculate the probability of the model using three approaches:

1. Akaike's information criterion (AIC) for maximum likelihood inference An option the parmfile allows to turn on a search for all migration model that are subsets of the model that was used to sample genealogies [this may break on several models with low number of estimated parameters]. This option may use a very long time (longer than you want to wait) when there are more than 4 (!) populations. The number of migration models increases hyper-exponentially with number of populations, AIC tests with more than 6 populations will take forever. These tests are only approximate because only the full model was evaluated through the MCMC run.

2. Bayes factors Bayes factors evaluate the merit of hypotheses and models in a Bayesian context. BF do not need to compare nested hypotheses (necessary for likelihood ratio tests). Evaluating Bayes factors is problematic because the marginal likelihoods needed to calculate the BF are difficult to evaluate. In a Bayesian inference program we normal only need to record the parameter values to construct the posterior distribution (histogram). For the marginal likelihood we need to estimate the denominator of the Bayes formula, we can integrate by recording all priors and likelihoods. Two methods are implemented in MIGRATE:

   (a) Harmonic mean estimator: described by Kass and Raftery (1996) . This method is know to be fast but inaccurate. It is implemented in many other programs (BEAST/Tracer, MrBayes)

   (b) Thermodynamic integration: described by Gelman and Meng (2003). This method needs multiple chains that run at different temperatures (use static heating because the other methods are not well explored yet). This methods can be very accurte but time consuming.

Performance of MIGRATE

Markov chain Monte Carlo programs are difficult to use and despite what people tell you very error-prone. This chapter tries to convince you that MIGRATE *often is doing the correct thing, and when something goes wrong that you perhaps can find out why and how it went wrong.*

Markov chain Monte Carlo samplers have the proven property that when they are run infinitely long they converge to the correct value, but since we cannot run the program infinitely long, we are interested how many samples we need to get before we start to get "acurate" result. This is true for maximum likelihood and Bayesian inference modes of the program. Despite the huge literature about measures when to stop sampling, there is still no good universal criteria available. MIGRATE reports some measures, such as the effective sample size of an MCMC run, or the Gelman-Rubin statistic. The problem of difficulty to converge can be divided into three simple categories:

1. Programming errors, typically programs of this complexity will always contain some errors, programmers certainly try to make every effort to make sure that there are no errors in the main calculations, but testing is typically very difficult especially when interactions among multiple options, different hardware need to be tested.
2. The sampler was not run long enough, this is data dependent and some general guidelines could be given, but NSF panels do not seem too keen to fund projects that would do that. To my knowledge, no study has explored effects of sample size, sequence lengths/variability of sequence for more than a single population (**???**). You have to explore this with your own data.
3. The assumptions of the model are not met, all data will violate some of the assumptions but typically the method is quite tolerant.

I will discuss some ways to investigate these three sources of problems in the following paragraph, highlighting the potential source of error.

The program is sampling form the right distribution: running the sampler with no data (e.g. sequence data with all "?" data) should result in the distribution $\mathrm{Prob}\ (G|\mathcal{P}_0)\mathrm{Prob}\ (D|G)$, the one we sample from [checks **(1)**]. With Bayesian inference the uninformative data runs will return the prior distribution [checks **(1)**].

Large simulation studies show that we can recover parameters and population structure that was used to create the data [checks **(1,2)**]. Such simulations need to be planned very carefully because silly parameter combination may suggest that the method does not work, but we perhaps would hope that under biological useful parameter ranges the program should deliver good results, an example of a study where the parameter range was not optimal is a paper by **?**. Real data may have difficulties to deliver consistent results, the most common source of this problem seems that either the model is heavily

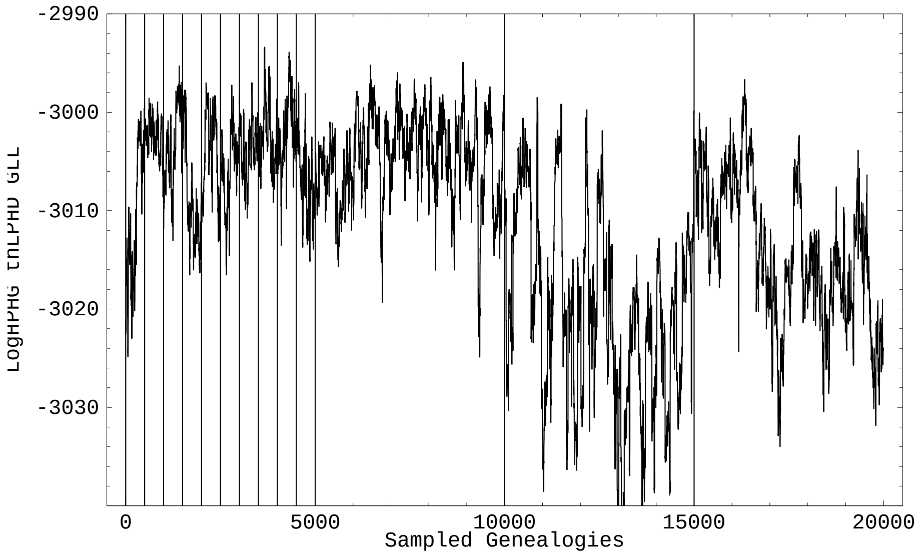

Figure 18: Data likelihood $\text{Prob}\ (D|G)$ for all sampled genealogies: A sample run of migration estimation using 2 populations, the very long vertical lines mark chain boundaries (10 short and 3 long chains). Totally, 10 short chains $\times 500$ sampled genealogies $+3$ long chains $\times$ sampled 5000 genealogies were sampled out of total 400,000. The values for not recorded trees are not shown.

violated (non-neutral loci, non-random mating, very high rate of recombination). For many data sets this seems not to be a problem, so.

The program is sampling many different genealogies; one can show this by plotting a curve showing on the x-axes all sampled trees and on the y-axis the likelihood of the genealogy (in our case this is $\text{Prob}\ (D|G)$, Figure 18). A plot of a sequence of $\text{Prob}\ (\mathcal{P}|G_i)\text{Prob}\ (D|G_i)$ is not useful because the genealogies contain different number of time intervals, and they are **not** comparable.

One can show that starting from random start parameters, the estimates converge rather quickly after a few short chains (Figure 19), the updating of the start parameters over several short chains moves the estimates to the proper region and the remaining uncertainty is only driven by the often huge uncertainty about the parameter estimates in the data, the likelihood surface is flat for many parameter combinations and the data. [checks **(2)**]

Comparison with other programs produce similar results. I compared MIGRATE with GENETREE (**?**) and with `fluctuate` (**?**). The comparison with GENETREE used two populations (England and Ghana: 2.5 kb sequence data for the beta-globin locus (**?**)) and the results were very similar. For my paper on n-population I have worked out a 100-locus data set simulation that shows that GENETREE and MIGRATE deliver the same estimates, and approximative confidence intervals, although GENETREE is very slow compared to MIGRATE for that specific data set (**?**).

The comparison with `fluctuate` was for one population, yes you can run MIGRATE with only one population, and for a data set created using a $\Theta = 0.01$ MIGRATE delivered $\Theta = 0.0123$ with a 50% confidence interval of 0.08 to 0.017, while `fluctuate` delivered a point estimate of $\Theta = 0.0119$.

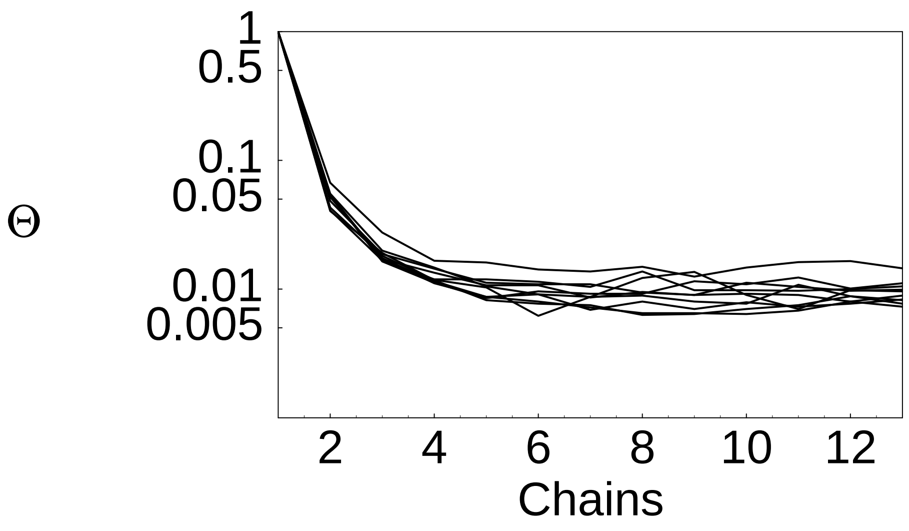

Figure 19: Convergence to the true parameter region. Ten runs were started from a $\Theta = 1.0$. The data was generated using a $\Theta = 0.01$. Totally, 10 short chains $\times 500$ sampled genealogies $+3$ long chains $\times$ sampled 5000 genealogies were sampled out of total 400,000.

In 2007, **?** published a new method to infer population-scaled mutation rates, their findings helped me to find a problem with my microsatellite estimator and now accuracy and speed are very similar to their estimator (Figure 20 **?**). [checks **(1,2)**]

MIGRATE Version 2.2 and newer print out statistics that help to assess whether the program was run long enough: (1) effective sample size [ESS] and (2) Rubin-Gelman statistic to assess convergence. I am not a strong believer of such measures because they only show the worst problems. For example effective sample sizes of 1000 may seem a lot but it certainly depends on the number of other parameters and the correlation among parameters. The program TRACER (Rambaut et al. 2005) flags effective sample sizes below 100; this is very low for population genetic purposes, I suggest that you strive to get at least 1000 or more for all parameters including the likelihood of the genealogies.

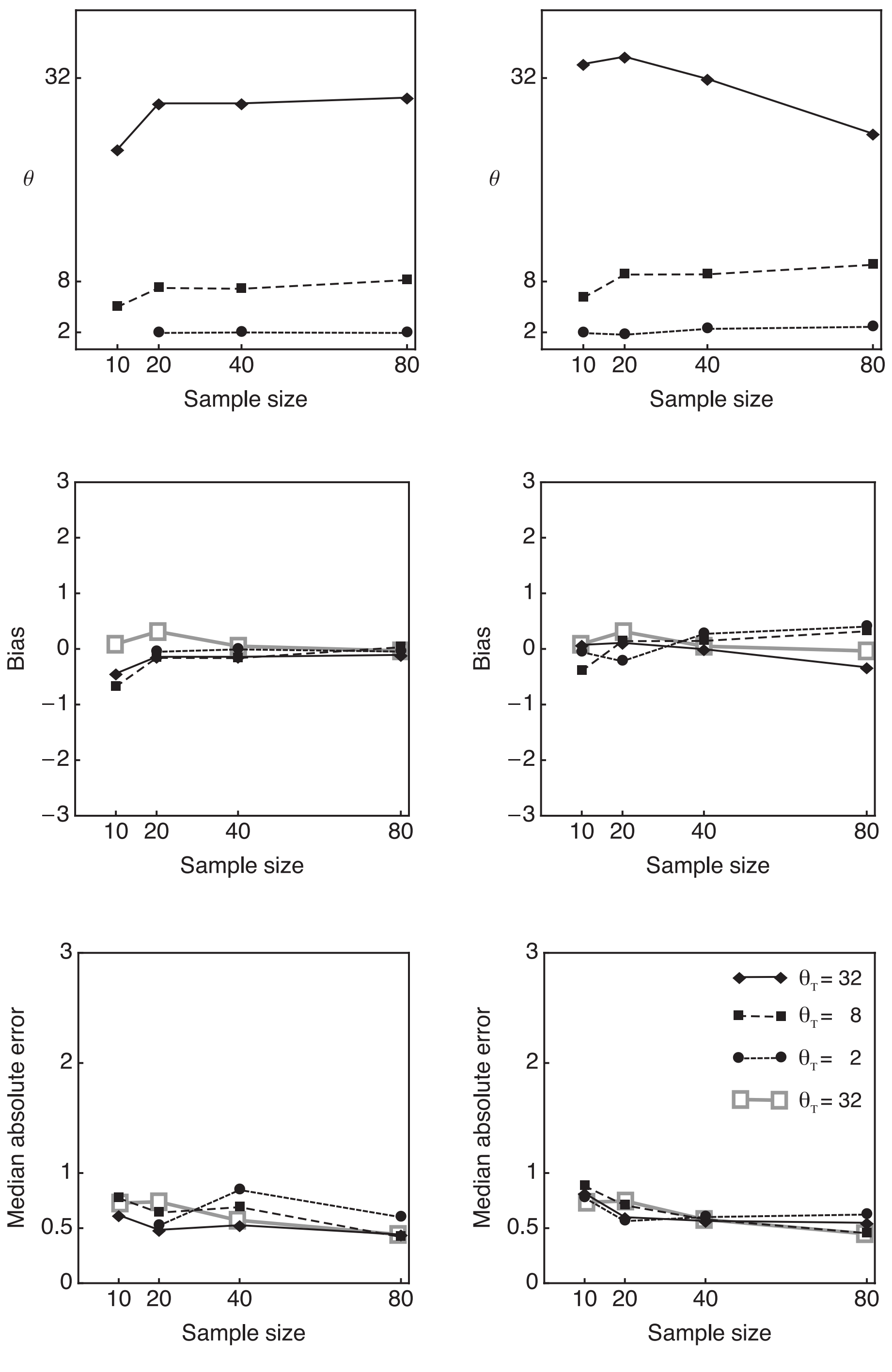

Figure 20: Mutation-scaled population size estimated from microsatellite data. Bias and absolute error for MIGRATE version 2.3. Left column: using the stepwise mutation model. Right column: using the Brownian motion approximation, Scale and calculations of bias and absolute error are the same as in Figure 1 in **?**. The open squares are the values for θ=32 from their paper.

Bibliography

Beerli, P., 2004 Effect of unsampled populations on the estimation of population sizes and migration rates between sampled populations. Molecular Ecology **13:** 827–836.

Beerli, P., 2006 Comparison of Bayesian and maximum likelihood inference of population genetic parameters. Bioinformatics **22:** 341–345.

Beerli, P., 2009 How to use MIGRATE or why are Markov chain Monte Carlo programs difficult to use? In *Population Genetics for Animal Conservation*, edited by G. Bertorelle, M. W. Bruford, H. C. Hauffe, A. Rizzoli, and C. Vernesi, volume 17 of *Conservation Biology*, pp. 42–79, Cambridge University Press, Cambridge UK.

Chib, S. and *Greenberg, E.*, 1995 Understanding the Metropolis-Hastings algorithm. American Statistician **49:** 327–335.

Hammersley, J. M. and *Handscomb, D. C.*, 1964 *Monte Carlo Methods*. Methuen and Co., London.

Hastings, W. K., 1970 Monte Carlo sampling methods using Markov chains and their applications. Biometrika **57:** 97–109.

Hudson, R. R., 1991 Gene genealogies and the coalescent process. Oxford Surveys in Evolutionary Biology **7:** 1–44.

Kingman, J., 1982a The coalescent. Stochastic Processes and Their Applications **13:** 235–248.

Kingman, J., 1982b On the genealogy of large populations. In *Essays in Statistical Science*, edited by J. Gani and E. Hannan, pp. 27–43, Applied Probability Trust, London.

Kingman, J. F., 2000a Origins of the coalescent. 1974-1982. Genetics **156:** 1461–1463.

Kingman, J. F. C., 2000b Origins of the Coalescent: 1974-1982. Genetics **156:** 1461–1463.

Kuhner, M. K., Yamato, J., and *Felsenstein, J.*, 1995 Estimating effective population size and mutation rate from sequence data using Metropolis-Hastings sampling. Genetics **140:** 1421–1430.

Metropolis, N., Rosenbluth, A. W., Rosenbluth, N., Teller, A. H., and *Teller, E.*, 1953 Equation of state calculation by fast computing machines. Journal of Chemical Physics **21:** 1087–1092.

Nath, H. B. and *Griffiths, R. C.*, 1993 The coalescent in two colonies with symmetric migration. Journal of Mathematical Biology **31:** 841–851.

Notohara, M., 1990 The coalescent and the genealogical process in geographically structured population. Journal of Mathematical Biology **29:** 59–75.