Documentation for IMa2

By Jody Hey

Department of Genetics, Rutgers University

Table of Contents

I. INTRODUCTION

This document explains how to use the IMa2 computer program. The program implements a method for generating posterior probabilities for complex demographic population genetic models. IMa2 works similarly to the older IMa program, with some important additions. IMa2 can handle data and implement a model for multiple populations (for numbers of sampled populations between one and ten) – not just two populations (as was the case with the original IM and IMa programs). The method for multiple populations is described in two papers (Hey 2010b, a).

For questions about this program, as well as about IM or IMa, users should consult the *Isolation with Migration* discussion group:
http://groups.google.com/group/Isolation-with-Migration

That basic 'Isolation with Migration' model and the use of Bayesian inference and Markov chain Monte Carlo is described in the "Introduction_to_IM_and_IMa" document. To understand the general principals behind the overall approach, all of which apply to the latest program, it is useful to read this introductory document.

II. WHAT IS NEW

Since the last update:

- removal of the twostep heating model
- removal of the option to record migration times from the markov chain
- bug fixes in model comparison and calculation of LLR statistic
- other bug fixes that caused some crashes
- .ti file format has changed, files generated under previous versions cannot be run with the latest program

Since the original IMa program:

The only major conceptual addition to IMa2 that makes it different from the original IMa program is that it can handle data from multiple populations. This requires that the user specify a phylogenetic tree. Importantly the tree must be rooted and the sequence in time of internal nodes must be known and specified. The addition of a tree for multiple populations to the model means that there can be a very large number of parameters (see Figure 1). Obviously the more populations and parameters the more data that will be needed to obtain useful parameter estimates. The general rule of thumb is that when increasing sample size, more loci are better than more individuals, especially for older histories. Since adding populations to a model usually involves adding parameters associated with older times (e.g. older splitting events, population sizes and migration rates for ancestral populations), most investigators will want to think in terms of adding more loci to their data sets. Since the number of parameters is close to having some sort of exponential scaling with number of populations, it might be useful to think in terms of increasing

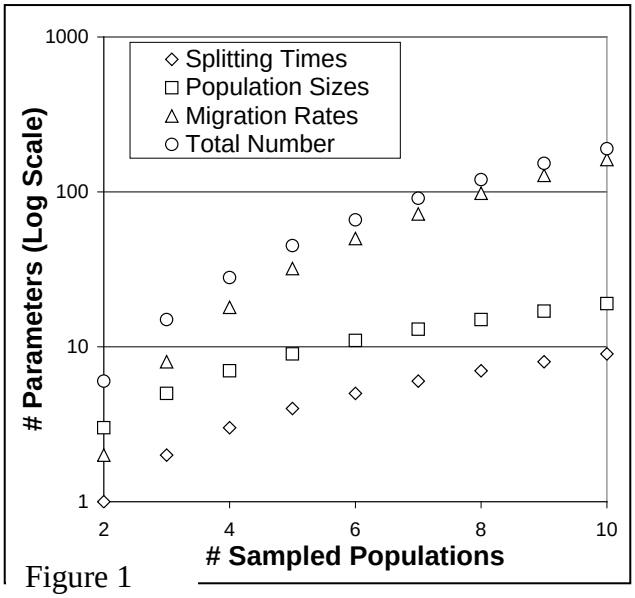

Figure 1

samples sizes in a similar way (e.g. doubling the number of loci for every population added to the model).

Apart from allowing multiple populations, no part of the basic 'Isolation with Migration' model, nor the general MCMC approach that is used (Hey and Nielsen 2007) has changed. The mutation models that are allowed are unchanged from previous programs, and the input file format is only slightly different, in order to allow for specification of the phylogenetic tree.

The program can be used to conduct all of the analyses that the original IMa program can conduct for two populations. The main things that are new in IMa2:

- Allows the study of isolation-with-migration models for multiple sampled populations. Any number of sampled populations, from 1 to 10, can be studied.
- Many options for specifying parameters in the model and prior distributions
- The optional use of exponential priors on migration rates
- The study of population migration (i.e. *2NM*) rates

Many details in the running of the program are different from previous programs. There are more options for controlling the duration of a run and for assessing mixing of the MCMC simulation. The command line interface has also changed.

III. RUNNING IMa2, A BRIEF OVERVIEW

The program is run by typing and entering a command at a command line prompt. If your computer is running a version of Microsoft Windows, then IMa2 will be run in a 'command prompt' window. It is usually simplest to have the program and data files in the same directory (folder), and for the command prompt window to be open in that directory (folder). If the data file, or any other file to be read by the program, is in another directory than the one from which the program is run, then the full path name for the file (i.e. including the directory and the filename) should be entered on the command line. For example if the data file is `mydata.txt` is in the directory `\mydatadir\` then the full name would be used on the command line, i.e. `\mydatadir\mydata.txt`

Each command line flag begins with a dash (`'-'`) and is followed by a letter, which can be upper or lower case. The options specified for the flag may be separated by the letter by zero or one spaces. e.g when using the `'-i'` flag, `–i mydata.txt` and `–imydata.txt` have the same effect.

The distributed version of the code and the windows executable will handle up to 200 loci, 10 populations, and 1000 chains. The upper figure for the number of sampled populations cannot be changed but the numbers of loci and chains can easily be changed in the `imamp.h` file followed by recompiling the program (see **Compilation).** Data is loaded into dynamic memory, so the actual size of a memory model during the run can be large and hard to predict (see **Compilation).**

Like the original version of IMa, version IMa2 is essentially two different programs intertwined. One part runs the MCMC simulation that generates samples of genealogies (i.e. samples from the posterior probability density of genealogies, given the data). The second part does the analyses on the function that is built from these genealogies.

Every new analysis must begin with running the program in 'MCMC mode' and this run must include a burnin period, followed by a period of sampling of genealogies. At the end

of the run, or at intervals specified by the user, the default analyses and additional analyses that were specified on the command line are done. Once the MCMC mode is completed, and if the genealogies were saved, the program can be rerun in the 'Load-Genealogies' mode to do additional analyses using the function generated from saved genealogies. For brevity and clarity, hereafter MCMC mode is identified as **M** mode, and Load-Genealogies mode is identified as **L** mode.

Users will notice that most countable things in the program output are counted beginning with the number 0. For example, in a model with four populations, populations are indexed using 0,1,2 and 3. The same goes for loci and other things. Thus, for example, if an error is reported involving locus 1 it means the second locus in the data file.

III.1. M Mode runs

Running in **M** mode means starting an MCMC simulation, running it until the state of the system is independent of the starting point (burnin) and then running for an additional time, while periodically saving genealogies, until the run is long enough and enough effectively independent genealogies have been saved. I generally try to arrange the duration of a run so that I end up with at least 20,000 genealogies. If the user wishes to estimate the joint posterior density and/or do likelihood ratio tests of nested models than at least 100,000 genealogies should be saved (more if more than two populations are in the model). The limit on the number of genealogies that can be saved or loaded is 300,000. It is possible this can be raised (change the value for `MAXGENEALOGIESSTOSAVE` in `imamp.h`) depending on available memory.

If the program is run in **M** mode, then the MCMC runtime output, that appears in the command prompt window, includes the following:

- The current step number and the current values for the probability of the data, given the genealogies, 'p (D|G)', and the prior probability of the genealogy 'p (G)'. These probabilities are not particularly useful, but it is good to check that they are actual numbers (non-numerical values indicate a floating point problem and the run should be halted).

- The update rates for splitting times and for genealogies. These are the percentage of proposed updates that were accepted based on the corresponding Metropolis-Hastings criteria. By default two types of updates are done for splitting times. One, called `NielsenWakeley,is` based on the method in Nielsen and Wakeley (2001). The other is called `RannalaYang` and is based on the method in Rannala and Yang (2003)). For genealogies, branch slide updates are done, with updates for single branches (identified as 'branch' in the output). For these updates, some fraction of these branch slides also affect the topology and/or the TMRCA and the update rates for these are given as well.
    - In the results file of an **M** mode run, update rates are also available for mutation scalars, and these can be requested for runtime output to the screen using a command line flag (`'-r4'`).
- The current value of splitting times and, depending on the number of loci and the `'-r4'` runtime option, mutation rate scalars and locus specific values of p (D|G).
- A table of Effective Sample Size (ESS) estimates and autocorrelations for splitting times and for the sum of P(D|G) and P(G) (called L[P], this is a useful summary of where the MCMC is at). This table begins to appear after 100,000 steps or so. The ESS estimates are of limited usefulness. For most runs they are pretty chaotic in the way they change over the course of the run.
- If Metropolis-coupling has been invoked then there are several series of numbers
    - The heating terms for each chain (beta values)
    - The swap rates between successive chains. If these are lower than 5% then they are not doing much good.
    - Actual swap counts between successive chains.

During a run in **M** mode it is a good idea to keep on eye on things at the beginning of a run to be sure that quantities are being updated. However, even if update rates are nontrivial it is quite possible that the chain is mixing poorly. Note that even if update and swap rates look ok at the start of a run, they may change as the system approaches stationarity, so it is usually advisable to watch a run for a few tens of thousands of steps before deciding whether or not to restart with different terms.

Update rates for splitting times can be quite low with some data sets (e.g. < 1%). If you are running only a single chain and have very low update rates for splitting time (or any other term), then you have a problem. However if you are running many chains, remember that the total update rate for a splitting time in chain 0 includes the rate of swapping with other chains. If splitting times are updating at higher rates in higher numbered chains, then it may be the case the splitting times for chain 0 have an overall update rate that is acceptable. The trend plots will tell the tale.

III.2. Assessing mixing and run duration in M Mode runs

For most data sets, and for all data sets with more than a few loci or with many gene copies per locus, predicting run duration is a complex problem that depends on how well the Markov chain explores the state space of genealogies. These issues are discussed in the document: *Introduction_to_IM_and_IMA.pdf*. The program provides several tools to help deal with this issue:

- The program will generate plots showing the value of L[P] and *t* over the course of the run. Runs that are mixing well will not show long term trends in these plots.
- The program estimates the autocorrelations of these same quantities. Long term trends appear as non-zero autocorrelations. Effective Sample Size (ESS) estimates are calculated from these autocorrelations.
- At the point when the posterior probability function is generated and an output file is created, the sampled genealogies are divided evenly into two sets: SET1 for the first half, and SET2 for the 2nd half. If the sampled genealogies are not autocorrelated over the full length of the run, such that SET1 and SET2 both contain a fair number of effectively independent samples, then the estimated posterior density functions based on SET1 and SET2 should be very similar. Similarly parameter estimates based on these functions should be similar. The program calculates parameter estimates for both sets.
- For each run multiple Metropolis-Coupled chains can (and typically must) be run to improve mixing over the course of a run.

- Multiple independent runs can be conducted using identical priors and number of coupled chains, but different random number seeds. If the separate runs give similar results, the genealogies from all the runs can be combined in a single **L** mode run for the analysis.

The simplest way of ensuring mixing is to do runs with lots of chains (e.g. the chimp data sets described in the paper were run with as many as 120 chains) and to watch for trends over the course of a couple hours/days/weeks (depends on the problem). If the run seems to be mixing not too terribly, based on the absence of visible trends and similarity of SET1 and SET2 parameter estimates, take note of the runtime settings so the run can be replicated with a different seed value. Then start some new runs using those values and different seeds. Assuming they all show similar and decent mixing, and similar results, you can combine all the results in a single **L** mode run. If you have several computers available then you might consider starting a large number of runs all with different random number seeds. (see **Getting Started Checklist**).

III.3. L Mode runs

If the program is run in **L** mode, then the only output to the screen is an occasional indication of which analyses are being done. L mode runs may take anywhere from a few seconds, for small data sets with only a few thousand genealogies and not many slow analyses, to a day or more. The slowest analyses are estimations of joint-posterior densities and likelihood-ratio tests of nested models.

IV. COMMAND LINE OPTIONS

Executing the program with no commands or only a command line flag of '-h' will cause the following output to the screen:

```
IMa2 Program - copyright 2011 by Jody Hey, Rasmus Nielsen, Sang Chul Choi, Vitor Sousa and Janeen
    Pisciotta
Release date: August 24, 2011
This program is run through a command line interface
To execute the program, type the program name followed by the necessary command line flags and options
-b  Duration of burn  (MCMC mode only)
    - If integer, the number of burnin steps
    - If floating point, the time in hours between writing of burntrend file
         run continues until file IMburn is no longer present
         in the directory, or if present, does not begin with 'y'
-c  Calculation options:
    0 Likelihood of data functions return a constant - posterior should equal prior
    1 Include ranges on mutation rates as priors on mutation rate scalars
    2 Joint posterior density calculations, for LLR tests of nested models use with -w (LOAD-GENEALOGY
    mode only)
    3 Get prior distribution terms from file (requires filename given with -g )
-d  Number of steps between genealogy saving (MCMC mode only) (default 100)
-f  Name of file with saved Markov chain state generated in previous run - use with -r3
-g  Name of file with parameter priors  (requires -c3) default: 'imapriors.txt'
-h  Heating terms (MCMC mode only):
  -hf Heating model: l linear (default); g geometric
  -hn Number of chains
  -hk Number of chain swap attempts per step (default = number of chains)
  -ha First heating parameter, effect depends on heating model
  -hb Second heating parameter, effect depends on heating model
-i  Input file name (no spaces)
-j  Model options:
    1  Migration only between sister populations (no migration between non-sister populations)
    2  One migration parameter for each pair of populations (do not use with -p5)
    3  Migration only between sampled populations (ancestral populations have zero migration)
    4  Add a non-sampled ghost population to the model
    5  Separate population size and migration parameters in each period (lots of parameters)
    6  No migration in the model
    7  Migration prior follows exponential distribution with mean given by -m or in parameter prior
    file
    8  Each ancestral population size is assumed to identical to that of their largest descendant
    population
    9  One single migration parameter for all pairs of populations (do not use with -p5)
-l  Run duration (default: 10000 genealogies sampled per locus):
     If in MCMC mode (i.e. not loading genealogies from a previous run)
       - If integer, the number of genealogies to save
         This value times -d value sets the # of steps in chain after burnin)
            - If floating point, the time in hours between outputs.
```

```
         Run continues until file IMrun is no longer present
           in the directory, or if present, does not begin with 'y'
     If in load-genealogy mode (i.e. using -r0 to load genealogies from previous run)
       - Integer indicates number of genealogies to load from file(s) named with -r
-m  Migration prior value (maximum for uniform,  mean if exponential distribution is used
-o  Output file name (no spaces) default is 'outfile.txt'
-p  Output options:
    0 Turn off trend plots in outfile (default is to print trend plots)
    1 Turn off plots of marginal curves in outfile (default is to print marginal density plots)
    2 Print TMRCA histogram for each genealogy (MCMC mode only)
    3 Print histogram of parameters on demographic scales  (requires mutation rate(s) in data file)
    4 Print histogram of splitting times divided by prior (do not use with -j0 or when only 2 sampled
    populations
    5 Print estimates and histograms of population migration rate (2NM)
    6 Print pairwise probabilities that one parameter is greater than another
    7 Print histograms of the number of migration events (MCMC mode only)
    8 Print joint estimate for splitting times (MCMC mode only, for models with 3, 4 or 5 populations)
-q  Maximum for population size parameters (4Nu)
-r  Run options
    0 LOAD-GENEALOGY Mode - load genealogies from previous run(s); also requires -v
    1 Do not save genealogies to a file (default saves sampled genealogies)
    2 Save the state of the Markov chain in a file - named with extension .mcf (MCMC mode only)
    3 Start run by loading a previously saved *.mcf file; requires -f (data and priors must be the
    same)
    4 Write all mutation related updates rates to stdout during the run (default is to suppress this)
    5 Print burntrend file at end of burnin period; use with -b followed by integer (MCMC mode only)
-s  Random number seed (default is taken from current time)
-t  Maximum time of population splitting
-u  Generation time in years - for use with -p3 (default is 1)
-v  Base name (no extension) of *.ti files with genealogy data  (requires use of -r0)
-w  Name of file with nested models to be tested (LOAD-GENEALOGY mode only), invokes -c2
-y  Mutation rate scalar for relevant loci - for use with -p3
-z  Number of steps between screen output (default is 10000) (MCMC mode only)
```

V. EXPLANATION OF COMMAND LINE TERMS

-b Duration of burn (MCMC mode only)

A sufficient burnin period is required in order to have sampled genealogies be independent of the starting state of the Markov chain. The minimal length of a burn-in chain depends on the data set and cannot be known prior to looking at the results of some runs. If the user is loading a previously saved Markov chain (using `-r3` and `-f` . see below), and if the goal is to essentially just continue the previous run that was used to save the state of the Markov chain, then the burnin can be made very short.

- If '`-b`' is followed by an integer, then this is the number of burnin steps.
- If '`-b`' is followed by a floating point number (i.e. with a decimal point), then this is the duration of the burnin time interval in hours. If there is not a valid file named '`IMburn`' in the current directory/folder, or until the first letter in that file is not a '`y`', then the burnin stops, otherwise it continues. This usage of the '`IMburn`' file is similar to the use of the '`IMrun`' file (see below) for controlling the length of the run. If the burntrend file option is invoked (`-r5`) then following each burnin time interval a burntrend file is written. In this way the user can inspect the state of the burnin and decide when it has been sufficient. Simply renaming or deleting the `IMburn` file will cause the burn to stop at the end of the current period.

-c Calculation options:

Multiple options can be specified at once (e.g. `-c01` invokes both options 0 and 1).

0 likelihood of data functions return a constant

This makes the analysis not dependent on the data and, if the program is working properly, should return the prior distributions in the resulting histograms. This is mostly used for debugging, but it can be useful for checking the priors on things that are not explicitly laid out. For example, when histograms are plotted for time and number of migration events (`-p7`), it is useful to do a run first with the `-c0` option to see what the priors are for these distributions.

1 Include ranges on mutation rates as priors on mutation rate scalars

If mutation rate range priors are included on in the input file, for multiple mutation rates, then the ratios of these limits are used as limits on the ratios of the mutation rate scalars. If two or more loci in the analysis have a prior range, then this allows the prior information on mutation rates to be included in the analysis and to shape the findings. Care must be taken in selecting a prior range, as it really should include all of the sources of uncertainty in the mutation rate. One possibility is to use a likelihood approach. For example, consider a locus that is going to be used in the analysis, and suppose that there is information on the mutation rate for that locus based on a comparison between other species (not in the analysis). Let x be the amount of observed divergence observed between those species, and let τ be the number of years since those species are thought to have separated (in this example assume that τ is large and does not include a component of coalescent variation from the ancestor). Then develop the math for the probability of x, $p(x \mid \tau, r)$. Now decide what the prior range should be on τ (based on whatever information was used to get τ - this will depend very strongly on the source of τ). Then integrate $p(x \mid \tau, r)$ over the prior range of τ to get $p(x \mid r)$ (you can assume a uniform distribution of τ, or something else – whatever makes the most sense), and define this quantity as the likelihood, $L(r \mid x)$. Now solve this expression for the most likely value of r (this will be the value of r you enter into the input file) and for the 95% confidence intervals (using standard likelihood assumptions) and use this interval as the prior range on r in the input file.

2 Joint posterior density calculations, for LLR tests of nested models use with -w (LOAD-GENEALOGY mode only)

Invoke this option to calculate the joint posterior density for demographic parameters (population sizes and migration rates). For accuracy this requires a good sample of at least 100,000 genealogies. For models with more than two sampled populations it is not possible (in a short enough time, with workable

numbers of sampled genealogies) to jointly estimate all parameters. In these cases joint estimates are obtained for all population size parameters, and then for all migration rate parameters. This option is usually used in **L** mode. If likelihood ratio tests of nested models are to be conducted then those models should be specified in a nested model file with the name given using the `-w` option.

`3 Get prior distribution terms from file (requires filename given with -g )`

This option allows the user to specify the upper bounds of uniform prior distributions individually for each of the splitting time, population size, and migration rate parameters. This option requires the use of a priorfile (see **Parameter Prior File Format**). In the absence of this option these are set on the command line (see `-q`, `-m` and `-t` options). If exponential priors are used for migration rates, then the values given in the priorfile are the means of the exponential distributions. This option is not compatible with most model options (i.e. with most `-j` flags).

`-d Number of steps between tree saving (MCMC mode only)(default 100)`

This is the length of the interval (in steps of the MCMC simulation) between the saving of genealogies. The default is 100. If an **M** mode run is desired to result in a specific number of genealogies, (i.e. `-L` is used with an integer) then the total length of the run will be the burnin length (`-b`) plus the product of the integer specified by `-L` and the integer specified by `-d` (or the default for `-d`). It is hard to know how many genealogies should be saved. Values less than 10,000 are probably only useful for two population models. Values greater than 100,000 often take a long time to analyze. For joint parameter estimates, or tests of nested models, at least 100,000 genealogies are needed.

`-f Name of file with saved Markov chain state generated in previous run`

This is used when the user has saved a file containing the state of the Markov chain, from a previous run, and wishes to load file to begin a new run.. When using this

option `-r3` must also be invoked. In general the input file, the priors and the heating model must be the same for the two runs.

`-g Name of file with parameter priors (requires -c3) default: 'imapriors.txt'`

This is used together with the `-c3` option to use prior distributions for population sizes, migration rates and or splitting times that vary among parameters. This option offers an alternative to setting the priors to be the same (e.g. using the `-m –q` and `–t` options). For example it is possible to turn off some, but not all, migration rate parameters and to exclude them from the model by setting their upper bound to zero.

`-h Heating terms`

Most **M** mode runs require the running of multiple *metropolis-coupled* (Geyer 1991)chains in which all chains, with the exception of chain 0, have updates accepted at a higher rate than specified by the Metropolis-Hastings criterion. This is called *heating*, and by swapping the state space of heating and unheated chains, it is possible to get much improved overall mixing of the state space for the unheated chain.

If multiple Markov-coupled chains are run, then it is important to have a heating scheme that leads to sufficient rates of swapping of the chains. When there are n chains, they are numbered from 0 to n-1. For chain i, the Metropolis criteria for parameter updating are raised to a power β_i , where $\beta_i < 1$. For all heating schemes chain 0 is not heated and chains 1 thru n-1 have successively greater values of β. Swapping is attempted by picking chains at random that are within 8 chain steps of one another, but most accepted swaps are between chains with the smallest difference in heating values. Chains with low values of β will be strongly heated, but will have lower swapping rates with chains that are much less heated. In general, heating schemes and data sets that give swapping rates between chains 0 and 1 that are less than 25% do not seem to be very helpful. For data sets where good mixing is achieved, the swap rates are 40% and 80% between chain 0 and chain 1, and generally between chains i and i+1.

It is not uncommon to run a very large number of chains (e.g. 100 or more). The speed of the program with n chains is roughly 1/n as fast as with 1 chain (though it is a bit slower

than this with very large numbers of chains), so it may seem that having 100 chains will cause the program to be incredibly slow. The tradeoff is that with a large number of chains the sampled genealogies are much more independent of each other and they can be sampled more often. When there are many chains and the Markov chain is mixing reasonably well the rate of sampling genealogies can be increased by reducing the interval between samples (i.e. reduced the `-d` value).

Selecting the optimal heating scheme (i.e. the number of chains and the set of β values) is not always easy. The general approach is to try some values, and then watch for output of the program to the screen and see if the swapping rates between successively numbered chains are ok. One difficulty with finding an adequate heating scheme is that the swapping rates between successive chains are often quite low, for low numbered chains (e.g. between chain 0 and chain 1), even for slight heating, whereas higher numbered chains often have higher swapping rates. The trick is to get sufficient swapping with chain 0, and yet also have sufficient heating in high numbered chains, so that the heated chains do really explore the parameter space.

Unless a small number of chains is desired, the preferred model for β values is what I have called the geometric model (see below). The program has other options, but these are mostly in there as leftovers from earlier programs.

All heating options are specified on the command line with terms that begin with '`-h`':

`-hf` the model for selecting β values. Two different heating schemes are included linear, and geometric (the two-step model is no longer included). Geometric is the model to use in most cases.

`-hfl` Linear model: This is the simplest model and requires only a single user specified parameter (using '`-ha`'). Under the linear scheme, each successive chain receives an additional heating increment. If the value specified with `-ha` is *x*, then the heating term β_i, for chain i, is given by

$$\beta_i = 1/(1 + i \times x)$$

`-hfg` Geometric model: This scheme is specifically intended for large numbers of chains where very small differences are needed between lower numbered chains. Two terms are required. *h1*, given by '`-ha`' on the command line, which specifies the degree of non-linearity, where 0<*h1* (also usually *h1* <1). The case of *h1*=1 gives a linear decline and *h1*<1 gives β values that decline slowly for low numbered chains, and fast for high numbered chains. Sometimes for very large numbers of chains it is useful to use *h1* that is slightly greater than 1. The second term, *h2* given by '`-hb`', is the lowest value of β (i.e. that is used for the highest numbered chain). The formula for β_i is

$$\beta_i = 1 - \frac{(1-h2)\times i \times h1^{n-1-i}}{n-1}$$

To determine *h1*, on the basis of a particular heating value for any specific chain *i*, β_i, the following formula can be used

$$h1 = \left(\frac{(1-\beta_i)(n-1)}{(1-h2)i} \right)^{\frac{1}{n-i-1}}$$

Some examples of heating schemes under the geometric model are given in the section titled: **Getting Started Checklist.**

`-hn` The number of chains. This must be at least 2, and will often be quite large. The maximum value is given in the `imamp.h` header file in the source code if a user needs to change it.

`-hk` The number of swap attempts per step. Usually the default value is sufficient and most users will not need to adjust this. The default value is set to either 1 swap per step, for less than 20 chains, or for higher numbers, one tenth of the number of chains.

Each step one or more swap attempts are made by randomly selecting two chains. If there are many chains then, to avoid picking chains that are unlikely to have an attempted swap (because of a large difference in heating values), swaps are only attempted between chains with index values that are within 7 of each other. Only swaps involving chain 0 ultimately have an effect on the genealogies that are sampled. This means that the swapping rate between chains 0 and other chains

must be fairly high, and also that the actual number of swaps is not small. The total number of swaps depends on the rate and the number of swap attempts. However if most swap attempts between adjacent chains are accepted at a very low rate then most successful swaps will soon be reversed (i.e. the acceptance probability of a reverse swap will be high). For this swapping rate is more important than the actual number of swaps, so long as the number of swaps over the course of a run between successive chains number at least in the hundreds. If the number of chains is small (e.g. < 20) then it is sufficient to have just one swap attempt per step. With more chains it is useful to have more swap attempts per step. There is also a maximum on the number of swaps which is nc(nc-1)/2, where nc is the number of chains.

`-ha` First heating parameter used by the model specified `-hf` . The meaning of this depends on the model (see `–hf`)

`-hb` Second heating parameter used by the model specified `-hf` . The meaning of this depends on the model and not all models require this second term (see `–hf`).

`-i Input file name (no spaces)`

The file with the data. For example, `-i mydatafile.txt`

`-j Model options`

There are various model options. Multiple options can be specified at once (e.g. `-j134` invokes options 1,3 and 4).

`1 Migration only between sister populations (no migration between non-sister populations)`

Removes migration parameters from the model for pairs of non-sister population (i.e. those not directly related by a splitting event). This reduces the number of migration parameters to two per splitting time interval, and greatly reduces the size of the overall model if there are many populations.

`2 One migration for each pair of populations`

For each pair of populations that have two migration parameters, these rates are set equal to each other and only a single parameter is used. Note that this cannot be used with the option invoking estimates of 2NM (-p5).

`3 Migration only between sampled populations (ancestral populations have zero migration)`

Removes migration parameters from the model for migration involving ancestral populations. This model is not terribly realistic, but it is difficult to have a data set large enough to inform on migration among ancestral populations. So it is possible that some users are willing to assume that this has not occurred in order to run a multi-population model with migration.

4 Add a non-sampled ghost population to the model

A ghost population is an unknown and unsampled population that might have affected your data, e.g. by exchanging genes with your sampled populations (Beerli 2004). Invoking this option will add an additional population to the model. The phylogenetic position of the ghost population is assumed to be as an outgroup to all sampled populations. Invoking this option will add two population size parameters and will greatly increase the number of migration parameters in the model.

5 Separate population size and migration parameters in each period (lots of parameters)

An alternative parameterization for a multipopulation (i.e. more than two sampled populations) model is to have different parameter sets for each time period. Under such a scheme a population that persists through two time periods would be represented by two sets of parameters. This framework adds a lot of parameters, but it does allow for more changes in population size and migration rates.

6 No migration in the model

No migration parameters are included and the model becomes a pure isolation model.

7 Migration prior follows exponential distribution with mean given by –m or in parameter prior file

Following the original Bayesian model specified by Nielsen and Wakeley (2001), the basic method uses uniform prior distributions by default. These are so-called ‘uninformative’ priors, and they are simple to work with, and they can provide a posterior density that is proportional to the likelihood. However two common issues arise regarding the use of a flat prior for

migration rate. One issue is that flat priors are only truly non-informative if (a) the user has an actual prior belief about the upper bound (which is not common), or (b) the posterior density reaches zero within the bounds of the prior. In the latter case, increasing the upper bound is not expected to alter the shape of the posterior density. But for many analyses the estimated posteriors for some parameters, particularly migration rates, are fairly flat and do not reach zero within the bounds of the priors that are used.

The second issue is that for many problems we actually do have a basic prior expectation that migration rates are low or zero. This is simply because the analyses are usually done on populations that have diverged somewhat and divergence is strongly retarded in the presence on gene flow.

These issues suggest that it might be useful to consider a prior on migration parameters that has its highest value at zero and that does not have an explicit upper bound. The exponential distribution has these properties, and invoking this option will cause an exponential prior to be used for migration parameters. In this case the value following the `-m` is the mean of the prior distribution or if a prior file is used the values given for migration rates are means of exponential distribution priors. Typically users will want to use, or at least start with, small mean values.

`8 Print joint estimate for splitting times (MCMC mode only, for models with 3 or 4 populations)`

For models with 3 or 4 populations it is possible to record the joint posterior density for splitting times (i.e. in 2 or 3 dimensions). This option causes the joint estimate for splitting times to be printed.

`9 One single migration parameter for all pairs of populations (do not use with -p5)`

Sometimes it is useful to ask about an overall migration rate. One way to get this is to test a nested model with just one migration parameter. However a more complete way is to invoke this option. When this is used all migration,

between all pairs of populations in all time periods occurs under the same parameter. This could be useful for multi-population data sets that are too small estimating multiple migration parameters.

`-l` `Run duration (default: 10000 genealogies sampled per locus)`

In MCMC (**M**) mode: If an integer is entered (e.g. '`-l25000`') it specifies the number of genealogies to save. If a floating point value (i.e. with a decimal point), this sets the duration of the chain in hours between printings of a results file. In this case, the program will run continuously, as long as there is a file named '`IMrun`' in the current directory/folder and as long as that is a simple text file that begins with the word '`yes`' (or the first character in that file is a '`y`'). The only purpose of this file is to be present when the user wants a run to continue, and to be absent when a run should come to an end. When using the '`-l`' flag with a floating point value the user controls the run length by deciding when to remove the `IMrun` file from the directory in which the program is running. At each printing of the results file, the previous results file is renamed to `*.old`. By taking a look at the distributions and trend plots in the output files, while the program is still running, one can run the program as long as desired. To halt the program at the end of the current interval, rename the `IMrun` file or edit it so that the first character is not a '`y`'. Be careful not to have the output file open (e.g. in an editor or a spreadsheet program) when it is time for the program to write to the output file, as the program will probably crash if this occurs. To view the output file, while the program is still running, it is usually best to rename it or to copy it to a new file.

In Load-Genealogies (**L**) mode: An integer given with the `-l` flag will specify the number of genealogies to load from the .ti files. If this integer is greater than the total number of genealogies contained in all files to be loaded, then all genealogies in those files will be used in the analyses. If this integer is less than the number of genealogies in those files, then the specified number of genealogies will be sampled from the total (with even spacing among all

genealogies available in the files). This option allows a user to sample the results from a long **M** mode run, that generated a large number of saved genealogies. This can be useful if a very large number of genealogies have been saved and the **L** mode analysis is slow if they are all used .

`-m Migration prior value (maximum for uniform distribution, mean if exponential distribution is used)`

This sets the upper limit of the prior distribution of the migration parameters, or if an exponential distribution is used it sets the mean of that prior distribution. If the prior is set to zero, it is the same as invoking a model with no migration (i.e. the same as `-j6)`. If the user wishes to set priors individually for each parameter then a priorfile should be used (see `-c3`).

`-o Output file name (no spaces)`

The default is '`outfile.txt`' however it is usually best to provide a more explanatory name.

`-p Output options:`

There are various output options that determine which tables of results are included in the output file. They can be given separately or all can be given at once (e.g. `-p452` invokes options 2,4 and 5).

`0 Turn off trend plots in outfile (default is to print trend plots)`

This turns off the printing of plots of parameter value trends. These plots are an essential tool for assessing MCMC mixing so ordinarily they would not be turned off.

`1 Turn off plots of marginal curves in outfile (default is to print marginal density plots)`

This turns off the printing of plots of the estimated marginal posterior density for all parameters in the output file. Such plots are not suitable for publication as they are based on ASCII characters and have low resolution, so you can turn them off if you prefer. However it is important to look at

these curves, either as ASCII plots or by loading histograms into a spreadsheet or some other program, at some point in the analysis.

`2 Print TMRCA histogram for each locus (MCMC mode only)`

M mode only. This prints a table in the output file of the distributions of times of the most recent common ancestor (TMRCA) for each locus. The units on these times are the same as for the splitting time parameters (i.e. mutation rate times time).

`3 Print histogram of parameters on demographic scales (requires mutation rate(s) in data file)`

This prints values and histograms for the theta and time parameters with the scales adjusted by the generation time and mutation rate, as given in the input file and runtime. If one or more of your loci have mutation rate estimates provided in the input file, then this option is very useful as it generates histograms on the relevant scales. For time to be scaled properly it is necessary to specify the number of years per generations (see '`-u`'). This option also requires the geometric mean of the mutation rate scalars for those loci for which a mutation rate has been provided in the input file. If in **M** mode, this will be calculated directly from the peaks of the mutation rate scalar posterior densities. If in **L** mode the program will not know this value and the user should provide it using `-y` (see below). This value can be obtained from a previous **M** mode run used to generate the '.ti' files used for the **L** mode run. The `-y` flag should also be used in an **M** mode run in which *all* loci have mutation rate information specified in the input file. In this special case, the user necessarily knows the true geometric mean of the scalars (i.e. it must be 1). If a user were to estimate the geometric mean in this case it will happen due to sampling variance that the calculated geometric mean of estimated mutation scalars will not be 1 (though it will probably be sort of close). For this reason, in the case when all loci have mutation rates in the input file, the user should specify `-y1.0` on the command line when invoking `-p3` for both **M** and **L** mode runs.

4 Print histogram of splitting times divided by prior (do not use with only 2 sampled populations)

When there are more than two splitting times in the model the marginal prior distributions are not uniform (Hey 2010b). In this case it can be useful to distinguish how much of the posterior density is due to the data and how much is due to the prior. This option causes histograms to be printed for the estimated marginal posterior for the splitting time *divided* by the prior distribution.

5 Print estimates and histograms of population migration rate (2NM)

It is possible to calculate an estimate posterior density for the population migration rate associated with each migration parameter. In general for migration from population *i* to *j*, backwards in times, we can appreciate the population migration rate as a function of a population size and a migration rate parameter, $2N_iM_{i\rightarrow j} = \theta_iM_{i\rightarrow j}/2 = 4N_iu\times(m_{i\rightarrow j}/u)/2$ (Hey 2010b). Considering this forward in time it is the rate at which population *i* receives migrants from population *j*. This option causes histograms to be printed for the all *2NM* values.

6 Print pairwise probabilities that one parameter is greater than another

From the posterior density for pairs of parameters it is possible to calculate the probability that one term is larger than another. This option causes two tables to be printed, one for population sizes and one for migration rates. Calculations are only conducted for parameters with identical prior distributions.

7 Print histograms of the number of migration events (MCMC mode only)

This generates a file (with extension '.mpt') of the histograms that estimate the posterior distribution for numbers migration events. This is a large file if there are many loci and migration parameters. Histograms are provided in each direction for each locus and for the sum of all loci. These tables and

numbers do not have a direct connection to the migration parameters. These measurements are done for each pair of populations regardless of how migration has been parameterized.

`8 Print joint estimate for splitting times (MCMC mode only, for models with 3 or 4 populations)`

In case a user wants to compare the joint estimates of splitting times to the estimates from the marginal densities, this option can be invoked. It works by setting up a series of bins in multi-dimensional space, and so has real sampling issues and won't work well for short runs. Also it only works at the moment for models with 3 or 4 sampled populations.

`-q Maximum for population size parameters (4Nu)`

This is the value of the upper bound on the prior distribution for all of the population size parameters. These parameters are values of 4*Nu*, where *N* is the effective population size and *u* is the mutation rate per generation. For multiple loci *u* is the geometric mean of the mutation rates of all the loci, per generation. Importantly the mutation rates *are not* per base pair and so it follows that the population size parameters *are not* on a per base pair scale. Users might want to estimate population size parameters ahead of time for their data to help think about where to set the priors. If you do so, be sure not to calculate on a per base pair basis. Users of previous versions of IM and IMa programs please note that unlike in those versions, the default value of the upper bound is the value input on the command line (in previous programs the default was to multiply the user value times a rough estimate of the data that was calculated by the program). If the user wishes to set priors individually for each parameter then a priorfile should be used (see `–c3`).

`-r Run options`

`0 LOAD-GENEALOGY Mode - load genealogies from previous run(s); also requires –v`

This causes the program to run in **L** mode.

`1 Do not save genealogies to a file (default saves sampled genealogies)`

This option is used if no '.ti' file is desired and no **L** mode runs are anticipated.

`2 Save the state of the Markov chain in a file - named with extension .mcf (MCMC mode only)`

Generates a file of the state space of the Markov chain simulation. This file is saved at the end of the burnin and each time an output file is generated.

`3 Start run by loading a previously saved *.mcf file; requires -f (data and priors must be the same as previous run)`

This option causes the program to begin by loading a previously saved state space file. The run should be started with the same data file, priors and heating terms as used in the original run that generated the '.mcf' file.

`4 Write all mutation related updates rates to stdout during the run (default is to suppress this)`

This option causes values and update information for mutation rate scalars to be printed to the screen during the run. If there are many loci this is a large table. Usually mutation rate scalars update at high rates with no problem. This information is always printed to the output file regardless of whether they are requested for screen output.

`5 Print burntrend file at end of burnin period; use with –b followed by an integer (MCMC mode only)`

This option results in a file of trend plots that is printed at the end of a burn period that was started using `–b` followed by an integer indicating the number of steps in the burnin. If the burn was set using a time period, then this option is automatically invoked.

`-s Random number seed (default is taken from current time)`

This is an integer (e.g. '`-s 111`'). It is useful to specify a value for two reasons. First if you want to repeat an identical run. Second if you are starting multiple separate replicate runs on the same data set you will want to make sure they start with different seeds. It can happen on some computers that if the runs start near each other in time they will end up with the same starting seed from the system clock.

`-t  Maximum time of population splitting`

The upper bound on the prior distribution for splitting times. This applies to all splitting times in the model. If the user wishes to set priors individually for each splitting time then a priorfile should be used (see `–c3`).

`-u  Generation time in years - for use with -p3(default is 1)`

Specify the generation time. This is only needed if you want the population size parameters rescaled to reflect effective population sizes. It is used together with the option to print histograms on demographic scales, '`-p3`'. Use of these options also requires that at least one locus have a mutation rate given in the input file.

`-v  Base name (no extension) of *.ti files with genealogy data (requires use of -r0)`

When running in **L** mode the user must specify the base name of the files with an extension of '.ti'. All of the *.ti files to load must be in the same directory. For example if you have two files with genealogies that you want to load and they are named `myrun1.out.ti` and `myrun2.out.ti` you would specify as the base name '`myrun`' (i.e. `–v myrun`). If the .ti files are in a directory that is not the same as the directory with the data file, that directory should be included as part of the base name of the .ti files.

`-w  Name of file with nested models to be tested (LOAD-GENEALOGY mode only), invokes –c2`

To conduct log-likelihood ratio tests of nested models the user must generate a file that specifies precisely which models to test. This option is used to read in the name of file containing those models. This option is used together with `-c2` which invokes the joint posterior density estimation. These options require large numbers of sampled genealogies (e.g. 100,000 or more).

`-y  Mutation rate scalar for relevant loci - for use with -p3`

This is the geometric mean of the estimated mutation rate scalars. This is usually used for an **L** mode run with a mean value obtained from a previous **M** mode run. If all loci in the data file have a mutation rate provided in that file, then use `–y1`. See `-p3` (above) for additional information.

`-z  Number of steps between screen output (default is 10000)`
`(MCMC mode only)`

The default is 10,000, but if the program is quite slow for your data, and you want to look at things immediately, it is useful to set it to a small number (e.g. 100 or 1000).

VI. INPUT FILE FORMATS

IMa2 works with several types of files, some of which must be prepared by the user if they are needed. The primary input file is of course the data file and all analyses require one of these. A user may also prepare a priorfile, which contains information on the prior distributions for each of the parameters. A user may also specify a nested model file, which contains information on what nested models to conduct likelihood ratio tests on.

VI.1. Data File Format

The format for data files for IMa2 is very similar to that for IM and IMa. The differences are that IMa2 requires two extra lines, one for the number of populations and one for the population tree string. The format is as follows:

- line 1 - arbitrary text, usually explaining the content of the file
  - After line 1, but before line 2, comments can be included to provide explanatory information. Each line of comment must begin with a '#'
- line 2 - an integer, the number of populations, `npops`.
- Line 3 – the population names in order, separated by one or more spaces. There must be `npops` names. This order also corresponds to the order in which the populations are numbered in the population tree and the order in which the data occur for each locus. The program numbers populations from `0` up to `npops-1`.
- Line 4 – the population string in modified Newick format. It is important to get this right. The string contains information on the topology of the tree for the sampled populations *and* information on the ordering of the internal nodes in time. These internal nodes correspond to ancestral populations. The ancestral populations are numbered beginning with `npops` for the most recent ancestral population and proceeds up to `2×(npops-1)` for the ancestor of all the sampled populations. Sampled populations in the string are represented by their respective number. Ancestral populations are represented by a colon, i.e. ':', followed by their ancestral population number. Examples of trees and strings for three different topologies for samples from four populations are shown.

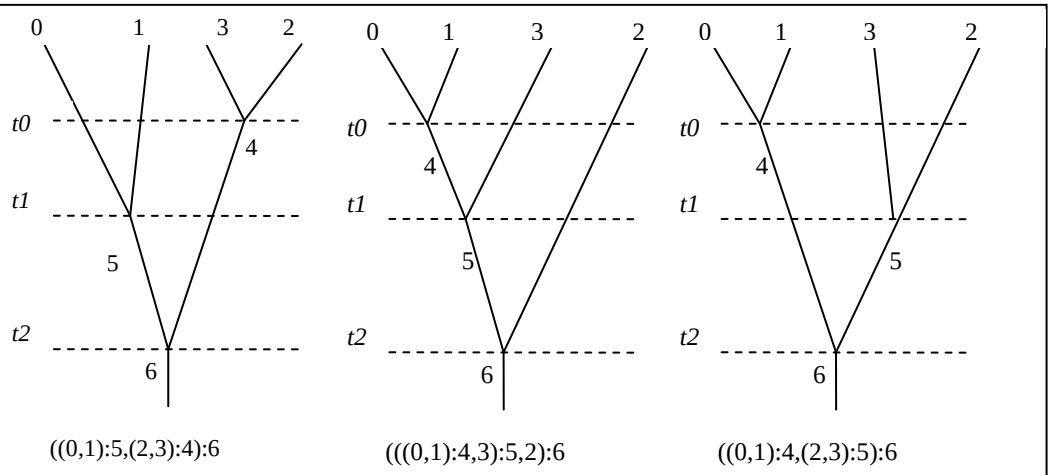

Figure 2

If there is only a single population then the tree string is simply: `0`. If there are two populations then the tree string is: `(0,1):2`

- line 5 - an integer, the number of loci in the data set, `nloci`.
- line 6 - basic information for locus 1. This line contains, in order and each separated by spaces: the locus names; the sample sizes for each population; the size of the locus; the mutation model; the inheritance scalar; possibly a mutation rate; and possibly a range of mutation rates.
    - Locus name (no spaces within the name)
    - n_0 thru $n_{npops-1}$, the sample sizes for the each population for that locus. These numbers do not need to be the same for different loci. If a population is not represented at this locus, a zero is used for that population.
    - the length of the sequence (if SSM model – a number is needed here, but it is ignored, if HapSTR, the length pertains to the sequence portion of the data)
    - Letter indicating the mutation model (I- IS, H - HKY, S - SSM, J - joint SSM and IS = HapSTR). If SSM (S) or HapSTR (J), the letter is followed immediately (no spaces) by the number of linked STR markers within the locus.
    - Inheritance scalar - For example: 1 for autosome, 0.75 for X-linked, 0.25 for Y-linked or mtDNA.

- The mutation rate per year for the locus (not per base pair). This can be left blank, but is needed for at least one locus in the data set if parameters on demographic scales are to be estimated. If there are multiple STRs in the locus then there can be multiple mutation rates on this line separated by spaces. If the locus is a HapSTR, then the first mutation rate given applies to the sequence portion of the locus with subsequent values corresponding to STR markers included in the locus.
- If the mutation rate is given, it can be followed by a range of mutation rates that can be used (with ranges for other loci in the analysis) to set priors on the ratios of mutation rate scalars. The range is entered with an open parentheses, the lowest value, a comma, the highest value, and a closed parentheses (e.g. '(0.00001, 0.00004)'. The range must bracket the rate. For a locus with multiple mutation rates, and multiple ranges, each range follows its corresponding mutation rate immediately on this line.
- line 7 - data for gene copy # 1 from population 0. For this line and all other data lines, the first 10 spaces are devoted to the sample name. The sequence or allele length (for SSM model) begins in column 11 of the file.
  - Note that each line is for a single copy of the locus. If you have two copies of a gene from an individual (e.g. STR genotypes), then use two lines, one for each gene copy.
  - For SSM or HapSTR data, the allele length assumes a step size of 1. This means that data from STRs that are multiples of lengths greater than 1 must be converted to counts of the number of base repeats (e.g. for a dinucleotide 'CACACACACACACACACA' the length would be 9). Any number less than 5 causes the program to stop with an error (it is assumed that such low counts could not evolve under the stepwise mutation model). If the data is for an SSM model locus and there are multiple STRs, then there will be one integer on each line for each STR, separated by a space. If the locus is a HapSTR (joint IS and SSM) then the STR data is given on the line, beginning at column 11, followed by the sequence data. For SSM data, as for other types of

data, only one gene copy is represented on each line of the data file. This is true even if the original data consists of diploid genotypes. In other words, diploid genotype data must be broken up and listed, with one data line for each gene copy.

- For a DNA sequence, the sequence for a gene copy is given all on one line without gaps. Base positions with anything other than an A, C, G, or T will be ignored for *all* gene copies for that locus (i.e. the program does not accommodate missing data within a sequence and it ignores indels).

- lines 8 thru line ($6+n_0+n_1+\ldots+n_{npops-1}$) - the remainder of the data for locus 1. Each line contains the data for one sample. The data for locus 1 for population 1 immediately follow those for population 0, and so on.
- Additional lines for additional loci. Each locus begins with a line containing the information for that locus, in the same format as for the first locus. The sample names and sample sizes for additional loci and the inheritance scalars and mutation model for additional loci do not have to be the same as for locus 1 (generally they are not).
- Finally, after all the data the file should end on a blank line.

Here is an example for a tiny three locus data set (a larger example is provided with the source code). In this made-up example the mutation rate per year is known and specified for locus 1, and for each STR in locus 3, but not for locus 2. The inheritance scalars vary from 0.25 for locus 1, 0.75 for locus 2 and 1.0 for locus 3. In one case the sample size for a population is 0 (pop3 in locus 2).

```
Example data  for IMa
# example data set
3
pop0 pop1 pop2
((0,1):3,2):4
3
locus1 1 1 2 13 I 0.25 0.0000000008
pop0_1    ACTACTGTCATGA
```

```
pop1_1    AGTACTATCACGA
pop2_1    AGTACTATCACGA
pop2_2    AGTACTATCATGA
hapstrexample 2 1 0 4 J1 0.75
pop1_1    13 GTAC
pop1_2    12 GTAT
pop2_1    12 GTAT
strexample 2 2 2 1 S3 1  0.00001  0.000015 0.00008
strpop01a 23 12 9
strpop01b 26 10 11
strpop11a 25 10 9
strpop11b 31 11 9
strpop21a 26 12 11
strpop21b 26 13 12
```

VI.2. Parameter Prior File Format

Hereafter a file containing the terms for the prior distributions of model parameters is called a priorfile. A priorfile is used whenever a user wants to tailor their prior distributions for specific parameters. Some examples:

- The user has reason to think that the most recent splitting time is much more recent than the other splitting events in the history of the sampled species and wishes to constrain the first splitting event to fall within this time.
- The user has reason to think that gene exchange did not happen between particular pairs of populations, either between sampled populations or between their ancestors, and so wishes to exclude some migration parameters from the model. This can be done by setting the upper bound on the prior distribution for those terms to zero.

If it is used, the priorfile must specify all priors for all splitting-time, population size, and migration rate parameters. The file is a simple text file with format as follows:

- Any line at any point in the file that begins with a pound sign, '#', is ignored. These lines can be used for explanatory text.
- The first line that does not begin with a '#' gives the population tree string for the sampled data. This must be the *exact* same tree string as is given in the data file. For example, the left-most population tree in Figure 1 has the following

string: ((0,1):5,(2,3):4):6. This means that sampled populations 2 and 3 joined most recently (at ancestral population 4); that sampled populations 0 and 1 joined longer ago (at ancestral population 5); and that ancestral populations 4 and 5 joined longest ago at ancestral population 6.

- The second line repeats the population tree string but also contains information on the upper bounds for the prior distributions for splitting times. Each population number in the string is followed by a colon, which is followed in turn by a floating point number (i.e. with a decimal point) that gives the upper bound of the time that population split from its sister population. No splitting time term is given for the root population, but every other population needs a splitting time upper bound. This means that the upper bound for each splitting time will appear exactly two times in the string (for each of the sister populations that separated at that splitting event). Successive splitting times can have identical upper bounds, but an upper bound for an earlier event can be greater than for a later event. For example, with the population string from above, and supposing that the upper bound on the most recent splitting event is 0.5 and that it is 5.0 for the older two events, the string would be : ((0:5.0,1:5.0):5:5.0,(2:0.5,3:0.5):4:5.0):6.
- The third line again repeats the population tree string but this time it contains information on the upper bound of the population size parameters. Each population number (for sampled and ancestral populations) is followed by a colon and then by a floating point number. For example, if populations 0, 1 and 4 have an upper bound of 5.0 and the remainder have an upper bound of 10.0 the string would be: ((0:5.0,1:5.0):5:10.0,(2:10.0,3:10.0):4:5.0):6:10.0
- After these three tree strings, there is a matrix that provides the prior terms for migration parameters (either upper bounds for uniform priors, or means for exponential priors). With n sampled populations there are a total of $k = 2n-1$ populations (including sampled and ancestral). The migration terms are given in a $k \times k$ matrix, with the migration term for the parameter corresponding to migration from population i to population j (backwards in time, in the coalescent) given in row i and column j of the matrix. The term for the

migration parameter for the reverse direction is of course given in row *j* and column *i* of the matrix. Many cells of the matrix will be zero simply because it is not possible to have migration, given the population tree, between the corresponding row and column populations. In the tree used here for examples - ((0,1):5,(2,3):4):6 - the migration parameters are $m_{0\rightarrow1}$, $m_{1\rightarrow0}$, $m_{0\rightarrow2}$, $m_{2\rightarrow0}$, $m_{0\rightarrow3}$, $m_{3\rightarrow0}$, $m_{1\rightarrow2}$, $m_{2\rightarrow1}$, $m_{1\rightarrow3}$, $m_{3\rightarrow1}$, $m_{2\rightarrow1}$, $m_{3\rightarrow2}$, $m_{0\rightarrow4}$, $m_{4\rightarrow0}$, $m_{1\rightarrow4}$, $m_{4\rightarrow1}$, $m_{4\rightarrow5}$, $m_{5\rightarrow4}$. However there can be no parameter $m_{3\rightarrow4}$ (or many others) simply because the two populations never coexist during any time period (see Figure 1).

For any non-zero element in row *i* and col *j* of the migration term matrix there must also be a non-zero element in row *j* and col *i* and vice versa. Similarly if either is zero, then the other must also be zero. This is a basic constraint of the method - it is not possible in the MCMC simulation to have gene flow in only one direction. If the user wishes they can set the upper bound on one to be a lower value than the other. Necessarily the last row and the last column will include nothing but zero's because they pertain to the basal ancestral population, which can not have had gene exchange with any other population in the model.

Migration terms can be set to zero, with the effect of removing those migration terms from the model. In the following example migration upper bounds are set to 2.0 between sister populations (i.e. 0&1, 2&3 and 4&5), and to zero between all others (this actually mimics the `-j1` option if used with `-m2.0`). Rows are counted down from the top and columns starting from the left.

```
0.0 2.0 0.0 0.0 0.0 0.0 0.0
2.0 0.0 0.0 0.0 0.0 0.0 0.0
0.0 0.0 0.0 2.0 0.0 0.0 0.0
0.0 0.0 2.0 0.0 0.0 0.0 0.0
0.0 0.0 0.0 0.0 0.0 2.0 0.0
0.0 0.0 0.0 0.0 2.0 0.0 0.0
```

```
0.0 0.0 0.0 0.0 0.0 0.0 0.0
```

Preparing this matrix for models with multiple populations takes some care to be sure that the correct values are set.

VI.3. Nested Model File Format

For likelihood ratio tests of nested demographic models it is necessary to load a file that specifies precisely which models are to be analyzed. A file is included with this distribution, for the case of two populations, that specifies all of the possible models in which two or more demographic parameters (i.e. population sizes and migration rates) of the same type are different or set equal to each other, and in which migration is non-zero, or zero, in either or both directions (a total of 24 models not including the full model in which all are non-zero and distinct).

The nested model file is a simple text format file with format as follows:

- Any line at any point in the file that begins with a pound sign, '#', or that is empty is ignored. The lines that begin with '#' can be used for explanatory text.
- The first line that does not begin with a '#' contains an integer that specifies the number of models given in the file.
- Each model begins on a new line with the word 'model' followed by some text explaining the model.
  - For example: `model All population sizes equal`
- After the line that begins with model there are one or more lines, each of which begins with either with the word '`constant`' or the word '`equal`'
- After the word, either '`constant`' or '`equal`', there is a '`p`' or an '`m`' indicated that the constraint applies to either population or migration parameters.
- For lines that begin with the word '`constant`', after the '`p`' or '`m`' there is a floating point number that is the constant value that some parameters must take. The most typical use of this is to set migration rates to zero in the model, but any constants can be used.

- For lines that begin with '`equal`' after the '`p`' or '`m`', or for lines that begin with the word '`constant`' and after the floating point constant value, the parameter numbers are given to which the constraint (i.e. being equal to each other, or constant at the given value) apply. For example
  - Migration parameter 1 is set to 0: `constant m 0.0 1`
  - Population size parameters 2 and 3 are equal to each other: `equal p 2 3`
- The '`constant`' and '`equal`' lines refer to parameters by their parameter number, which in the case of migration rates is not related to the populations they apply to.
  - The numbering of population size parameters is the same as is the index number of the populations themselves (i.e. parameter # 4 applies to population #4).
  - The numbering of migration rate parameters is more complex but it can be found simply by looking at the sequence of migration parameters in tables in the main output file. Tests of nested models are done in **L**-mode runs, so a the results of a previous **M**-mode run can be used to get the numbering of migration parameters.

For models with just two sampled populations it is possible to develop models that impose constraints on both population size and migration rate parameters. Samples of 100,000 genealogies seem to give adequate estimates of the joint posterior density and of the corresponding joint posterior parameter estimates. However for more sampled populations the dimensionality is so high that it is not really possible to estimate a joint density with only 100,000 genealogies. For three or more sampled populations joint parameter estimates and nested model tests are all done either with only population size parameters, or only migration parameters. In these cases the '`constant`' and '`equal`' lines of a model must be all '`p`' or all '`m`'.

The file `All_nested_model_tests_two_populations.txt` gives all 24 nested models for the case of two sampled populations.

VII. OUTPUT FILES

The program generates up to five different types of output files, including: the main results file, genealogy files (ending in .ti), Markov chain state file (ending in .mcf extension) for restarting a run; migration histogram files (ending in .mpt extension) for plotting counts and times of migration events; and burntend files for showing trend lines during the burnin period.

VII.1. Main Results File

For the most part the main results files for **M** and **L** runs are fairly similar in their overall layout because most analyses can be invoked in either type of run. Output files for runs in M mode will contain additional information near the top of the file about the Markov chain simulation.

The Main Results File contains the following sections:

INPUT AND STARTING INFORMATION

This section lists the starting information from the input file and the command line settings, including parameter counts and priors and information on each of the loci in the data file.

LOAD GENEALOGIES (L) MODE INFORMATION (**L** mode only)

If the file was produced by a **L** mode run then information on the base filename for the *.ti file(s) is given as well as information on the number of genealogies loaded from each file. After this is printed the header information from the first genealogy file (*.ti file) which contains information on the original MCMC (**M** mode) run that generated the *.ti files.

MCMC INFORMATION (**M** mode only)

This section, which appears only in a file generated by an MCMC (**M** mode) run, summarizes information about the Markov chain simulation over the course of the run, beginning with the length and time of the run and the number of measurements made. These are followed by the highest values observed for the probability of the data, given the genealogy and the parameters, P(D|G) and the highest values for the prior probability of

the genealogy P(G). These values are not used in any tests and are not of much use, but there may be times when it could be useful to compare them between runs.

Next in the section summarizing the MCMC simulation is important information on the rates at which the Markov chain was updated. For the most part these are the same numbers that are written to the screen during an **M** mode run. For each type of measurement of the state space, there is shown the parameter, the type of update, the number of attempted updates ('#Tries'), the number of accepted updates ('#Accp'), and the acceptance rate ('%'). The first table of update rates is for the splitting time parameters, followed by a table for genealogy update rates, followed then a table for update rates for mutation scalars and other parameters associated with specific mutation models. If multiple chains are invoked then the last table of acceptance rates are for the swapping rates between each chain and its neighboring chain with a higher heating rate.

Following the update rates, the output file has a table of parameter autocorrelations and ESS (effective sample size) estimates. These are the same as appear on the computer screen during the course of the run. ESS values are estimates of the number of independent points that have been sampled for each parameter. They can be useful, but are often highly unstable and should be used only in conjunction with other assessments of mixing**.**

PARAMETER COMPARISONS, GREATER THAN PROBABILITIES (with –p6)

These tables are invoked with the –p6 and show the probability that one parameter is greater than another, for all pairs of population size parameters and then all pairs of migration rate parameters. Calculations are done only for parameters that have the same prior distribution. These calculations are done by integrating over the joint posterior density estimate for pairs of parameters. These are estimated probabilities and will have some (unknown) variance depending on how many genealogies have been sampled.

MEANS, VARIANCES and CORRELATIONS

These are tables of means and variances of population size and migration parameters. The table of correlations between parameters is also printed. For shorter runs it is possible that some correlations cannot be calculated. High correlations are indicated by an '$' or a '*' because they are a sign that the model may be over specified for the data. The idea here is that if two parameters are very strongly correlated, then there is in effect only one parameter.

MARGINAL PEAK LOCATIONS AND PROBABILITIES

This is a listing of tables of the parameter estimates for population sizes and migration rate terms obtained from the location of the peaks of the estimated marginal densities for each term. For each time period there are tables for population size parameters and migration rate parameters. Also if the population-migration option (`-p5`) is invoked, values are given for population migration rate (i.e. *2NM*) terms. For each parameter set in each period, the parameters are listed as a table, with each parameter receiving two columns, one for the parameter value and one for the probability associated with that value. Only those parameters that appear first in that period are listed for that period. For example, if a population size parameter extends through periods 1 and 2, then results will be listed under period 1 and not period 2. The table has the following rows:

- 'Param' this row has the parameter labels alternated with the letter 'P' (for probability). For migration rates and population migration rates the parameters refer to migration backwards in time (i.e. "in the coalescent"). This is important to remember, because when you write up your results you don't want to get things backwards. For migration and population migration rates the first number in the label is the population in which the genes started and the second number is where they went to. For example `m1>2` means the genes went from population 1 to 2 backwards in time in the coalescent direction (i.e. from population 2 to 1 forwards in time). For `2N0M0>3` the genes went from population 0 to 3 backwards in time and from 3 to 0 forwards in time.
- '`Set0`' this row has the parameter estimates and the probabilities calculated using ½ of the genealogies (the first half of all those that are in memory).

- 'Set1' this row has the parameter estimates calculated using the second ½ of the genealogies that are in memory. If there are enough genealogies and they are sufficiently independent of one another, then the Set0 and Set1 values should be quite similar. This is yet another way to check to see if the MCMC simulation sufficiently explored the state space.
- 'All' parameter estimates and probabilities using all the genealogies. These numbers will not be very useful if the Set0 and Set1 values are not similar.
- 'LR95%Lo' estimated lower 95% confidence limits under assumptions of likelihood ratio tests. For this number the posterior density is treated as a likelihood and the lower 95% limit is the point on the curve, for a parameter value less than the estimate, where the curve is 1.92 units (log scale) below its highest point. In general the LR95%Lo and LR95%Hi values are not as useful as the 95%HPD (highest posterior density) values that are given in the histograms.
- 'LR95%Hi' same as 'LR95%Lo' but for a value higher than the estimate.
- 'LLRtest' For migration rates and population migration rates the likelihood ratio test of Nielsen and Wakeley (Nielsen and Wakeley 2001) is done. See also Hey (2010b) . Statistical significance is indicated by asterisks, or lack thereof by 'ns'.
- 'LastPeriod' this is the number of the oldest period in the phylogeny in which that parameter exists. Periods are numbered beginning with zero.

Joint Peak Locations and Posterior Probabilities(**L** mode only)

To make this table the analysis must be done in L mode and the –c3 option must be invoked. This table can also report the results of likelihood ratio tests of nested models if a nested model file is loaded using the –w option. The table reports the maximum joint posterior density, and the estimated parameter values at that maximum, for various models. At the least a full model is analyzed, which for two population models means a model with all three population size parameters and both migration rate parameters. For more than two populations there are two 'full' models, one for all population size parameters and one for all migration rate parameters, since a model with all of both requires too many sampled genealogies and too much time to find the joint posterior density estimate.

For each model the log of the posterior probability is shown with the parameter estimates. Parameter values that are shown as bracketed (e.g. [0.01]) represent the values for parameters that are not in the model, either because they were set to a fixed value or because they were set to be identical to another parameter.

For likelihood ratio tests the table lists the degrees of freedom (which may not apply in cases of parameter values fixed at the boundary of a prior distribution, e.g. migration set to zero, see (Hey and Nielsen 2007)) and the LLR statistic which can be compared with a χ^2 distribution. Also shown is the effective sample size (ESS) for the number of genealogies used to estimate the parameters. For portions of the posterior density that are high and flat (e.g. as expected near the peak) there are likely to be several genealogies that contribute substantially to the calculation of the posterior density. However for parts of the surface that have lower probability and are less flat, there is likely to be a much wider variance among genealogies in their contribution to the estimate of the posterior. Typically many nested models have their peak posterior density at a low point on the surface and for which there may be very few genealogies that contribute much. These models will usually have an ESS value of 1.0 which means that there will be a wide variance in the actual estimated value of the posterior density and in the estimated parameter values for that model. This is a much bigger issue for models with really low probability, so the LLR tests are likely to still be valid (and indicate rejection of the model). This is because, even though the LLR values have a wide variance, they are also probably very large. In other words, most of the time when an ESS value of 1.0 turns up (when many genealogies have been used), the LLR value will be vastly larger than a critical value from a χ^2 distribution and so it will be safe to reject the null model, even though there is not much confidence in the particular LLR value.

HISTOGRAMS

Here begins a set of large tables that are suitable for importing into a spreadsheet. For each set there are actually two tables, the first of which summarizes the key things about the larger, second table. The larger table includes 1000 rows and gives the estimated posterior

probability for each of 1000 values of a model parameter. These tables can be used for estimating parameter values and are suitable for importing into a spreadsheet program and generating a plot. For demographic parameters (population size and mutation rate parameters) the estimates obtained from these tables should be very similar to those obtained from the table labeled `MARGINAL PEAK LOCATIONS AND PROBABILITIES.`

The histogram tables are generated for a variety of summaries of the data, but all include the following information:

- Histogram group # and title for the histogram group.
- One or more lines of explanatory information
- '`Summaries`' is the title of the table of summary information which includes:
    - '`Value`' this line contains labels for this terms in the histograms
    - '`Minbin`' the midpoint value of the lowest bin of the histogram with a non-zero value
    - '`Maxbin`' the midpoint value of the highest bin of the histogram with a non-zero value
    - '`HiPt`' the bin with the highest value in the histogram
    - '`HiSmth`' the bin with the highest value after smoothing. This is only used for histograms of values sampled from the MCMC simulation and will usually be a more useful estimate than the value given by '`HiPt`'. For histograms calculated from posterior density function no smoothing is done and the highest point should be taken as the estimated value. Values calculated from posterior density functions tend to be pretty smooth already.
    - '`Mean`' the mean value calculated from the histogram. For terms with posterior density functions this should be close to the mean calculated for the tables of means and variances (see above).
    - '`95%Lo`' the estimated point to which 2.5% of the total area lies to the left. This is probably not very useful. Note that this is different from the 95% lower limit calculated on the basis of likelihood ratio assumptions and

reported in the table on MARGINAL PEAK LOCATIONS AND PROBABILITIES (See above).

- ‘95%Hi’ the estimated point to which 2.5% of the total area lies to the right- see info for ‘95%Lo’.
- ‘HPD95Lo’ the lower bound of the estimated 95% highest posterior density (HPD) interval. The 95% HPD interval is the shortest span (on the X axis) that contains 95% of the posterior probability. A question mark, ‘?’, is added if the HPD interval did not appear to be contiguous (in which case the HPD estimates are not reliable) which can happen with multiple peaks or if the surface is rough. A hatch symbol, '#', is added if the posterior density does not reach low levels near either the upper or the lower limit of the prior. In such cases the HPD intervals will obviously change with if the prior distribution is changed.
- ‘HPD95Hi’ - the upper bound of the estimated 95% HPD interval (see notes for ‘HPD95Lo’.

- Below the summary table the main histogram table begins. At the top are parameter labels, alternating with the letter ‘P’. Below this is a row that gives the bin value and probability for the highest probability in the table. If a pair of columns is imported into a spreadsheet the first column will be the X axis values and the second column the Y axis values. Below the table is a row title ‘SumP’. This is the sum of the probability values in the table. The actual value of this sum will vary depending on what is in the table. If the user needs to generate a plot in which all of the curves have the same area (or a particular area such as 1.0) then the value of this sum can be used to normalize the values for plotting. Also at the bottom of the table is a row titled ‘After’ and a row titled ‘Before’. These usually have zero’s in them, but there are a few circumstances where there is some probability for values of parameters either to the left (i.e. before) or to the right (after) of the values listed in the histogram.

ASCII Curves - Approximate Posterior Densities

Here begins a set of curves that are estimates of the marginal posterior densities for model parameters. These plots are not suitable for publication as they are based on ASCII characters and have low resolution, however they are useful for getting a rough picture of the marginal densities. For population size and migration parameters, the plots are based on the estimated marginal density functions. For terms that are in the MCMC simulation (like splitting times and mutation rate scalars), the plots are based on recorded values. In **M** mode plots are provided for all terms, both those in the MCMC and the population size and migration terms that are not in the MCMC. In **L** mode it is not possible to print plots for mutation rate scalars as there is no MCMC and these values are not saved in the '.ti' files. However recorded splitting time values are saved in the .ti files so a **L** mode run will generate ASCII plots using the splitting time values that correspond to the genealogies that were used from the .ti files.

ASCII Plots of Parameter Trends

These are plots of the values for parameters in the MCMC simulation over the course of the run. They are ASCII-based and crude, but are still quite useful for assessing how well the parameters are mixing. Any sign of a trend over a substantial portion of the plot, or from one end of the plot to the other, is indicative of too short a run. Similarly a plot in which a substantial region of the parameter space is visiting only once or a small number of times, is indicative of too short a run. A plot is also provided for Log(P) = Log(P(Data|Genealogy)) + Log(P(Genealogy)), which is useful for an overall sense of how well the chain is mixing. In the case of **L** mode, trends are provided only for the Log(P) values and the splitting time values that are saved in the '.ti' files.

VII.2. Genealogy (.ti) file

If genealogies are being saved during an **M** mode run, then at the end of the run a file with a '.ti' extension that contains all the information for the sampled genealogies will be saved. These files can be loaded later in an **L** mode run. At the top of each file is a summary of information about the run, followed by a table with the summary statistics for each

genealogy. Each row of this table contains all of the sufficient statistics for a single genealogy, as well as values for P(D|G), P(G) and the splitting times associated with that genealogy.

VII.3. Markov chain state file (.mcf)

If the state of the Markov chain is being saved during or at the end of an **M** mode run, so as to allow restarting a new run from that same point, then there will be a file with a '.mcf' extension. This file can be loaded at the start of another **M** mode run. The mcf file is unlikely to contain any useful information for the user and will not ordinarily be used for any purpose other than starting an analysis.

VII.4. Migration count and time histogram file (.mpt)

If histograms are generated for the time and number of migration events, then these results are written to a file with a '.mpt' extension.

VII.5. Burnin tend plotfile

If trend plots are generated during the burnin phase, these are saved in a file that ends in '.burntrend.out'.

VIII. THE IMfig PROGRAM

IMfig is a program for generating a figure of an isolation-with-migration model from the output file generated by running the IMa2 program. IMfig generates an eps (encapsulated post script) file which can then be printed or converted to a PDF file. This kind of figure represents an estimate of history as a fat phylogenetic tree made up of boxes (for sampled and ancestral populations), horizontal lines (for splitting times) and curved arrows (for migration). Time is represented as depth on the vertical axis, with the sampled populations/species at the top of each figure at the most recent time. Population size is represented as width along the horizontal axis. Each population is represented by a box, the height of which refers to how long it has lasted and the width of which refers to its effective size. The confidence interval for a populations size is given both by a double headed arrow extending from the right margin of that population's box and by faint boxes representing the lower and higher 95% Highest Posterior Density (HPD) intervals. Similarly 95% HPD intervals for splitting times are given by dashed lines and doubled-headed arrows. Migration arrows are printed depending on the users wishes. One option only prints arrows if 2NM values are statistically significant by the test of Nielsen and Wakeley (2001). Figure 3 was generated using IMfig for a there population model for common chimpanzees (Hey 2010a).

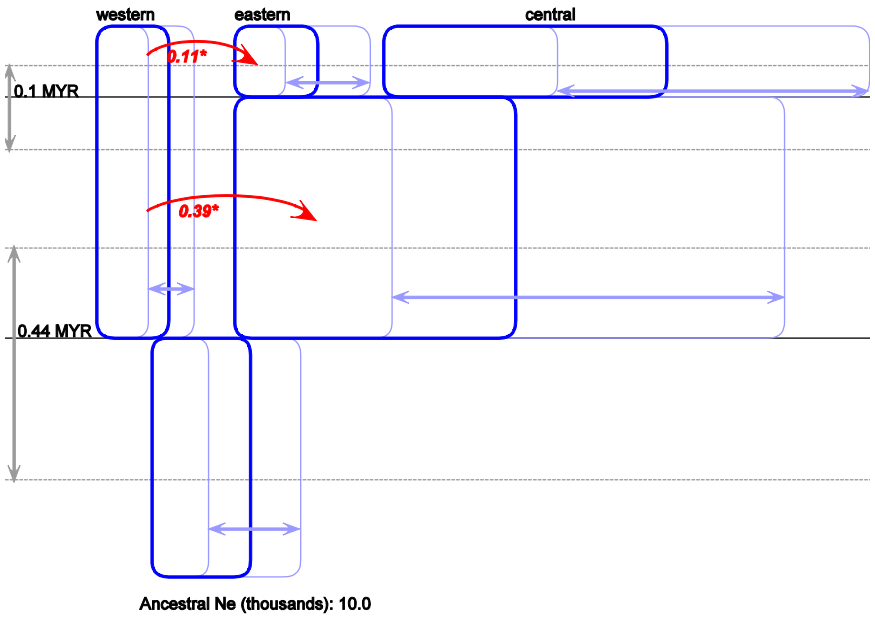

Figure 3

IX. GETTING STARTED, AND SUGGESTIONS FOR RUNNING IMa2

Just as when running the IM and IMa programs, but even more so, running the IMa2 program requires that the user have some understanding of the complexities, including model assumptions, MCMC mixing issues, and issues related to prior distributions. Some of this knowledge can really only be gained by reading the original papers and related literature. Other necessary information lies in the program manuals.

IX.1. Getting Started Checklist

Here is an ordered checklist of suggestions for how to get up and running so that your runs are likely to produce useful results. There are five steps: (1) build the data file; (2) set upper bounds for the prior distributions; (4) do a short preliminary run to see that the data loads; (5) do runs to assess mixing and set heating terms; and (6) do runs to get results you will use. After this final step the user may find that they need to revisit some issues about their data or their priors and to begin again.

1. The first step is to assemble the data to be used and to construct a data file. Be sure to review the section of this manual on the Data file format.
    a. For sequence data make sure that the data do not contain obvious signals of past recombination events. Hey and Nielsen (2004) discuss this issue at some length.
    b. Remember that for DNA sequences any base positions with missing data (i.e. ambiguous bases) or indels will cause all sequences for that locus to be ignored at those base positions.
    c. For microsatellite (STR) loci convert the allele designations to integers that differ by the number of repeats. It is ok if you know only the relative number. For example if you have STR allele lengths of 154, 156 and 158 and you know the base repeat is of length 2 but you don't know the length of the flanking sequences (so you don't know the absolute number of repeats), you can convert this simply to 77,78 and 79.
    d. For STR loci missing data and null alleles are not allowed. Also the data must fit the Stepwise mutation model (SSM), so do not use the locus if you

have alleles with lengths that are not multiples of the base repeat or have other reasons to doubt the SSM model. An gene copy with a repeat count of less than 5 will cause an error.

e. Obtain estimates for as many loci as you can of the mutation rate per year (for the complete locus, not per base pair). These are usually obtained by comparing one of your sampled populations with another population for which you have a divergence time estimate in years. Include this information in the data file (see Data file format).

f. If you have more than two populations, figure out your phylogenetic tree for your sampled populations. This is known of course for models with one or two populations.

2. The next step is to figure out what values to use for the upper bounds on your prior distributions. In this checklist it is assumed that you are setting priors by using the `-q, -m` and `-t` flags and that you are using uniform (not exponential) priors for migration rates.

   a. A general rule of thumb for picking priors to start with is as follows:

      i. Estimate the geometric mean of the population mutation rate, *4Nu*, across your loci, for each of your sampled populations. You can use whatever estimator you prefer (e.g. nucleotide diversity, Watterson's estimator, $\delta\mu^2$ for microsatellites, etc). Be sure that your estimate for each locus is for the entire locus and not on a per base pair basis. For example suppose you have three loci (two for which you are using the Infinite Sites mutation model and one microsatellite locus for which you are using the stepwise model) and suppose your first sampled population has estimated values of *4Nu* of : 1.3, 10.8 and 112.5 (the last value is high, as it might be for a microsatellite locus). The geometric mean in this case is 11.34.

      ii. Let the largest value of these geometric means, across your sampled populations, be *x*.

      iii. Set your upper bound for your population size parameters to be 5*x* (e.g.. if *x*=2.0 then use `-q10.0`)

iv. Set the upper bound of your splitting times to be 2*x* (i.e. if *x*=2.0 then use `-t4.0`)

v. Set the upper bound of your migration rate parameters to be 2/*x* (i.e. if if *x*=2.0 then use `-m1.0`)

b. The justification for this rule of thumb is as follows:

i. Simple estimators of population mutation rates (i.e. *4Nu*) terms provide an accessible guess for the sorts of values that will end up having high posterior densities. However you want the upper bound of your prior distribution to be higher than where you think the estimate is likely to be, so this is why I suggest starting with 5 times your ballpark value.

ii. Geometric means are used because all parameters are scaled by the geometric mean of the mutation rate across loci.

iii. For the prior on splitting times you also want to work from your population size parameters. Splitting time and population size parameters are on the same scale, and the splitting times of your species (in generations) will generally be of the same order as 4*N*. If it was much older than that, then you won't be able to study it much anyway because it will have been so long ago that drift will have removed the information about ancestral populations and your genealogies will coalesce with few lineages present in the ancestral populations. So a reasonable strategy for your first run, is to pick a splitting time prior that is based on your ballpark estimates of population size parameters. Another thing to keep in mind is that if you do not have a lot of information in your data for splitting times and you use a high upper bound for splitting times, then you may find that your analysis suggests splitting times at that upper bound. What seems to be going on in such cases is that the an island model seems to fit your data (i.e. ancient splitting time with steady gene flow since then). If you don't think such a model is realistic then start with a

lower upper bound on splitting times. If you have a lot of data then this issue is much less likely to be a problem.

iv. For the prior on migration rates it is usually best to start with a value not much higher than would correspond to a value of $2NM=(4Nu\times M/u)/2 = 1$. This is moderately high migration as it is, and because many data sets can't resolve migration posteriors very well it is easy to get in a situation where you have a high prior on migration and where mixing goes very badly. For this reason I suggest starting with an upper bound on the migration prior that is a few times the inverse of the ballpark estimate of the population size. I suggest 2 times the inverse, but 5 might also be fine. Unless you have a lot of data, starting with much higher values is likely to lead to a poorly mixing MCMC simulation.

3. Start a trial run that stops pretty quickly after a burnin and a run. The purpose of this is to see that your data is loading, and that you are getting updating. For example if your priors are '`-q2 –m1 –t3`' and your input file is called '`mydata.txt`', you could try the following command: `IM –imydata.txt –otrialrun.out –q2 –m1 –t3 –b10000 –l100 –s123` (note that because the `–l` is followed by the number of genealogies saved, and the default number of steps between saving genealogies is 100, this command line specifies 10000 steps for the run). After the run is completed you will have a file called `trialrun.out` that shows the results. After such a short run the results file won't be of any use for parameter estimates, but it will contain summary information on your data and on the MCMC update rates.
4. If the program runs with your data, then it is time to design a command line for an actual run to assess MCMC mixing. If you data set is on the small side, or you just want to see how things go with only one chain, then you can try a run with just a single chain. Most of the time you will end up doing runs with multiple chains. It is very common for users to start multiple runs, each with different numbers of chains and heating terms. The best heating terms are

those that have a high rate of swapping between adjacent chains (which can be found with short runs) and that lead to good mixing (which can only be found with long runs).

a. It can be useful to set the burnin to an indefinite period so that you can look at the trend plots and decide when to stop the burnin (see notes for `-b` and the use of the `IMburn` file). For example, to generate a burntrend file every 6 hours you would use `-b6.0` and have an `IMburn` file in the directory with your data. Then just look at trend plots in the burntrend file that is rewritten every 6 hours of the run. Typically these plots will show something like a curve that plateaus. When all the trendplots, including those for Log[P] and *all* splitting times have reached a sort of plateau after which there are no clear trends, you can delete or rename the `IMburn` file. Then when the current burn period reaches its end, one last burntrend file will be written and the actual run will begin.

b. For the actual duration of the run it is often useful to use `-l` with a floating point number (# of hours between generation of output files) and an `IMrun` file, much as was done for the burnin period. If `-l` is followed by an integer then this is how many genealogies will be saved and it will dictate the duration of the run (i.e. `IMrun` is not used). A key command line input that is relevant for this is the integer that follows `-d`. This is the number of Markov chain steps between genealogy saves (the default is 100 steps).

c. Almost all data sets with multiple loci require multiple Metropolis-coupled chains, often a great many. The guidelines here are to be sure to pick heating terms that lead to high swap rates between all adjacent pairs of chains and to have enough heating that the overall MCMC simulation mixes well. Don't scrimp on the number of chains (see notes under `-h`). Below are some examples of heating schemes for a couple types of problems. These are simply based on my experience running infinite sites data. If you have microsatellite data, then you will probably need a great deal more heating than suggested here:

- Small to medium sized data set (e.g. < 15 loci, < 20 individuals per locus), low heating: `-hfg -hn20 -ha0.96 -hb0.9`
- Small to medium sized data set, medium heating:
  `-hfg -hn40 -ha0.975 -hb0.75`
- Small to medium sized data set, high heating:
  `-hfg -hn80 -ha0.999 -hb0.3`
- Larger data set, low heating:
  `-hfg -hn40 -ha0.96 -hb0.9`
- Larger data set, medium heating:
  `-hfg -hn100 -ha0.99 -hb0.75`
- Larger data set, high heating:
  `-hfg -hn150 -ha0.999 -hb0.3`

5. Finally, when everything is set, including priors and heating terms, the user can set up two or more runs (using different starting seed values). In the end you will want to have at least 20,000 saved genealogies from at least two well mixed runs, and if you plan to estimate joint posteriors and do likelihood-ratio tests you will need at least 100,000 genealogies. One strategy is to do multiple **M** mode runs that save a lot of genealogies but that do only basic analyses (e.g. don't use `-c2` which can take a long time), and then after all these runs are done to do an **L** mode run that uses all the .ti files and that invokes all the analyses that you want. When you do an **L** mode run you can specify how many genealogies to load.

IX.2. How Many Populations can be in the Model?

IMa2 can accommodate anywhere from one to ten sampled populations. In the case of a single population there is only one demographic parameter (i.e. population mutation rate) and there is no phylogenetic tree (the phylogenetic tree string in the text file should just be '`0`'). With more than two populations it is important that the user know the true population phylogeny, including the sequence in time of the population splitting events in that phylogeny.

In addition to the phylogeny question, analyses with more than two populations present a number of difficulties because of the large number of model parameters. In the first place the amount of data required to study more than two populations is likely to be quite large. Users should probably think in terms of at least doubling the amount of data (particularly in terms of the number of loci) for every additional population added to the model. Thus if 15 loci proved useful in a case with two populations, it would probably be good to have at least 30 for the case of three populations, and so on. In the case of chimpanzees where a study with 48 loci worked well for pairs of populations (Won and Hey 2005), a data set of 73 loci was only marginally adequate for three and four population models (Hey 2010a).

One approach than can be useful for the study of multi-population models is to conduct analyses on smaller models with pairs of populations before building up to larger models. In the case of the chimpanzees many, but not all aspects of the divergence portrait that emerged in a four population model, were consistent with initial runs on pairs of populations (Hey 2010a).

Another possibility for smaller data sets from multiple populations, is to run a model with just a single migration parameter (`-j9`).

IX.3. How to Analyze Large Data Sets for which Runtimes seem to Take too Long

Depending on the amount of data, or the type of data (e.g. sequences vs microsatellites), or just the pattern of variation within the data, the mixing of the Markov chain simulation may be very slow. Hypothetically a user might have data from 1000 loci from five populations, only to find that the program requires a month or more just to achieve a suitable burnin. Realistically such analyses cannot be done on a single computer. In principle one way to circumvent this difficulty is to have a program that runs on multiple processors. Unfortunately the current version of IMa2 cannot be run in this way. However IMa2 can be run independently on

different machines in such a way as to use many computers for a single analysis. The general procedure is as follow:

1. Do preliminary runs to find heating terms that lead to both high heating values in the most heated chains and high swap rates between adjacent chains. This might require one hundred, or multiple hundreds of heated chains.
2. Start multiple jobs on multiple computers, using the same command line for each job but with different random number seeds. For each job use something like –b24.0 in conjunction with a IMburn file so that a burntrend file is generated every 24 hourse. Also make sure each job is set to save a Markov chain state (.mcf) file after each burn period. The purpose of these runs is to generate multiple mcf files from independent Markov chains that have each moved on to a point that is independent of where they started. The actual number of genealogies to sample (using –l) for these runs can be set to a low, nominal value (e.g. –l10).
3. Once these chains have run sufficiently long that there are no perceivable trends in the trendplots, they can be stopped. Simply remove or change the name of the IMburn file and the burn will end after the current period. If the actual genealogy sampling is set to be short, the run will finish soon after the end of the burnin period.
4. Now a new set of runs for genealogy sampling can be started by reloading the .mcf files. These runs can have short burnin periods. Also it is possible to start more runs for sampling genealogies than were used for the burnin. Each .mcf file generated can be used to start multiple sampling runs, provided each is given an additional burnin period so that they become independent of each other. This time may be much less than was required for the initial burnin.
5. These sampling runs can be conducted using -l24.0 and using an IMrun file so that they can be run indefinitely or until the maximum number of genealogies that the program can handle has been saved (this is a constant value of 300,000).
6. If many processors are available a user could do dozens or hundreds of runs.

7. Finally the results of these many independent runs can be combined in a single L-mode run. One wrinkle here is that in order to generate histograms for effective population sizes and splitting times in years (i.e. parameters on demographic scales), the user will need to provide the geometric mean of the mutation rate scalar estimates for those loci in the data file that have a mutation rate (using `-y` on the command line, this is 1 if all loci have mutation rates provided) . These can be obtained from the output files of the genealogy sampling runs.

IX.4. Avoiding Confusion about Migration Parameters and the Direction of Migration

One persistent source of confusion for users of IM or IMa has been the direction of migration. One might think that if we represent a migration event with an arrow (e.g. *from the population that contribute a gene → to the population that receives a gene*) that a conventional interpretation of time is implied (i.e. the contributing population had the gene, then at some later point in time the gene moved into the recipient population). But for better or worse, because the time scale for the genealogies in the MCMC is in the coalescent direction (i.e. the present is time 0 and time increases into the past), the directionality of migration events and migration events in these programs has been the reverse of what many users initially assume. In the main output file that results from an IMa2 run migration parameters are labeled with 'm' followed by 'source population number', a right arrow '>', and the recipient population number. For example, `m0>1` is the name of the migration rate parameter for migration of genes from population 0 to 1, backwards in time, in the coalescent direction. Interpreted forward in time, in the usual way that most people think about migration, `m0>1` is the rate at which population 0 receives genes from population 1.

IX.5. Understanding Population Migration Rates

The units of the migration rate parameters in IMa2 and previous programs are migrations per mutation, $m=M/u$, where M is a migration rate per generation and u is the mutation rate per generation (actually the geometric mean of mutation rates across loci). A numerical estimate of $m=M/u$ is not usually very useful except when it is zero, in which case the migration rate is zero regardless of its scaling. A far more useful measure of migration is the rate at which a population receives genes. It is possible for a population to be receiving genes at a high value of M/u, but if the population is very large then it may not be having much affect. Similarly a small population will more impacted by a given value of M/u. For this reason we usually wish to consider the population migration rate, which is the product of the effective number of gene copies in a population (i.e. $2N$) times the rate at which its genes are supplanted by incoming migrating genes. For population 0, with population mutation rate parameter $4N_0u$, the rate at which genes migrate into it from population 1 (with time moving in the conventional forward direction, not in the coalescent) is $M_{0\rightarrow1}/u$. Then the population migration rate, at which the genes of population 0 are supplanted by genes from population 1, is $2N_0\mathrm{M}_{0\rightarrow1} = 4N_0u \times (M_{0\rightarrow1}/u)/2$.

X. FINAL CAUTIONS AND SUGGESTIONS

One of the greatest challenges is knowing whether the chain is mixing sufficiently. Update rates, trend plots, and comparisons within and between runs must all be considered for deciding if your runs are long enough or have sufficient heating. One of the tools for assessing mixing is the ESS value. However it is recommended that users *not* rely strongly on ESS estimates of splitting times. These values are highly unstable over the course of a run, and so ESS values should be used in conjunction with other measures of mixing and by using multiple independent runs.

Regarding the mutation model, and the type of data to use, the program runs fastest with the infinite sites (IS) model (Kimura 1969), which may be a reasonable model for many nuclear gene loci sampled from closely related species. However the IS model is certainly not completely accurate and it may be a poor model for your data. For mitochondrial data the IS model is usually not reasonable and the HKY (Hasegawa et al. 1985) model is more likely to provide a reasonable fit (though even the HKY is often not adequate for control region or D-loop data). The program will handle microsatellite (STR) data, but be forewarned that runs will require many, many heated chains, that burnin times will be quite long, and that a complete analysis may take multiple processors weeks, or months or longer. The runs in Hoelzel et al., each took about a month (Hoelzel et al. 2007). These were for two populations, 15 loci, and a small number of alleles at each STR locus.

If the data set is small, and sometimes even if not, it can happen that a large portion of the likelihood surface is very flat over the range of some parameters. In particular it is not uncommon to have high, flat likelihoods for high values of *t*, for ancestral population sizes, or for migration rates. One may find for example in the curve for a splitting time (typically that for the oldest split in the phylogeny) a peak at a low value, and then a plateau that extends indefinitely to the right at an even higher value. This is awkward because the highest likelihood appears to be associated with an infinitely wide range of parameter values. In these situations, the data does not contain enough information to clearly identify the model under the prior you have specified.

XI. COMPILATION

A Windows executable is provided (`IMa2.exe`). This has been compiled under Microsoft Visual C++ . The source code is available if you need to change program constants or need to compile for another computer system. In particular, large data sets or data that requires extensive Metropolis-coupling may require that MAXLOCI and MAXCHAINS be increased in the imamp.h header file.

Sang Chul Choi has prepared an installation for unix/linux/mac. To install the linux/unix/mac arhive:

- save the archive to a suitable directory:
- to decompress, type at the command prompt: tar zxf ima2-8.26.11.tar.gz
- move to the ima2-8.26.11 directory (note the date numbers on the file may change with updates before this documentation is updated).
- type at the command prompt: ./configure
- then type: make
- the executable (called 'IMa2') will be in the src directory. It can be run by typing: ./IMa2

This program and the source code files are copyrighted. You may modify them as needed to recompile for different computers, or with different runtime constants, as needed to analyze your data. The source code may not be incorporated into other programs without permission from the authors .

XII. REFERENCES

Beerli, P. 2004. Effect of unsampled populations on the estimation of population sizes and migration rates between sampled populations. Mol Ecol 13:827-836.

Geyer, C. J. 1991. Markov chain Monte Carlo maximum likelihood. Pp. 156-163 *in* E. M. Keramidas, ed. Computing Science and Statistics, Proceedings of the 23rd Symposium on the Interface. Interface Foundation of North America, Seattle, WA.

Hasegawa, M., H. Kishino, and T. Yano. 1985. Dating of the human-ape splitting by a molecular clock of mitochondrial DNA. J. Mol. Evol. 22:160-174.

Hey, J. 2010a. The Divergence of Chimpanzee Species and Subspecies as Revealed in Multipopulation Isolation-with-Migration Analyses. Mol. Biol. Evol. 27:921-933.

Hey, J. 2010b. Isolation with Migration Models for More Than Two Populations. Mol. Biol. Evol. 27:905-920.

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

Hey, J., and R. Nielsen. 2007. Integration within the Felsenstein equation for improved Markov chain Monte Carlo methods in population genetics. Proceedings of the National Academy of Sciences of the United States of America 104:2785–2790.

Hoelzel, A. R., J. Hey, M. E. Dahlheim, C. Nicholson, V. Burkanov, and N. Black. 2007. Evolution of Population Structure in a Highly Social Top Predator, the Killer Whale. Mol. Biol. Evol. 24:1407-1415.

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

Nielsen, R., and J. Wakeley. 2001. Distinguishing migration from isolation. A Markov chain Monte Carlo approach. Genetics 158:885-896.

Rannala, B., and Z. Yang. 2003. Bayes estimation of species divergence times and ancestral population sizes using DNA sequences from multiple Loci. Genetics 164:1645-1656.

Won, Y. J., and J. Hey. 2005. Divergence population genetics of chimpanzees. Mol. Biol. Evol. 22:297-307.