Documentation for IMa3

By Jody Hey

Department of Biology, Temple University

Table of Contents

INTRODUCTION

This document explains how to use the IMa3 computer program (Hey, et al. 2017). Unlike previous IM programs, IMa3 allows an investigator to estimate the topology of the phylogenetic tree for the sampled populations or species. In this document the topology will often be referred to as Φ (Phi). Once an estimate of Φ has been obtained, IMa3 can be run using that fixed value of Φ in order to estimate the parameters of an Isolation-with-Migration model, just as one can do with IMa2 which requires a fixed value of Φ.

IMa3 is based on the older IMa2 and IMa2p programs and uses the same input file format. Nearly all of the options and analyses that were available with IMa2 are present in IMa3, although many of the command line options have changed. If compiled with MPI (`MPI_ENABLED` is defined during compilation) then IMa3 can be run with multiple heated chains distributed across multiple cores.

1.1. A brief History of IM programs and methods

IMa3 is the latest of a series of programs that implement analyses under an Isolation-with-Migration model. They include, in order:

- MDIV implements a new Markov-chain Monte Carlo (MCMC) approach to estimating the parameters of an Isolation with Migration (IM) model for two populations and 1 locus, with equal population sizes and equal migration rates (Nielsen and Wakeley 2001).

  http://people.binf.ku.dk/rasmus/webpage/mdiv.html

- IM implements a full six-parameter IM model for multiple loci (Hey and Nielsen 2004).

  https://bio.cst.temple.edu/~hey/software#im-div

- IMa implements a full six-parameter IM model with the prior on genealogies calculated by integrating over the prior for population size and migration rate parameters. This greatly reduces the dimensionality of the state space of the Markov chain simulation and allows for

calculation of the joint posterior probability density for all rate parameters, which in turn allows for likelihood ratio tests of nested demographic models (Hey and Nielsen 2007).

https://bio.cst.temple.edu/~hey/software#im-div

- IMa2 implements a full IM model for more than two populations using the same approach as IMa does for two populations. IMa2 requires a user-specified population phylogeny (Hey 2010).

  https://bio.cst.temple.edu/~hey/software#ima2-div

- IMa2p is a parallel version of IMa2 (Sethuraman and Hey 2015).

  https://bio.cst.temple.edu/~hey/software#ima2p-div

- IMGui a graphical user interface for running IMa2 or preparing IMa2 command lines (Knoblauch, et al. 2017).
- IMa3 provides for estimation of the posterior probability of the population phylogenetic topology by using a new kind of data augmentation called a 'hidden genealogy' (Hey, et al. 2017). IMa3 also implements hyperpriors.

1.2. What to read

The IM programs implement a complex set of analyses for Bayesian and Likelihood-based inference of evolutionary models as implemented in a series of papers: (Hey and Nielsen 2004; Hey and Nielsen 2007; Hey 2010; Hey, et al. 2017) . Furthermore these analyses are done using samples from a Markov-chain Monte Carlo (MCMC) simulation, which is complex as well; in part because it introduces a set of sampling issues (e.g. MCMC mixing) which must be taken into account if further downstream analyses are to be useful. Users of the these programs will want to have some understanding of how MCMC works.

The 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 available from the HeyLab website. 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.

1.3. Some conventions

In this document, which tries to be consistent with recent publications on IM models from the Hey lab, we will use some variables to refer to important quantities:

- Φ (Phi) is the ordered rooted topology of a population tree. It does not include the branch lengths. By "ordered" we mean that the sequence of internal nodes in time is considered to be part of the topology
- τ (tau) is a vector of the times of common ancestry of the populations, also called 'splitting times'. If we have an estimate of τ for a given value of Φ, then the τ values give the lengths of the branches of Φ,
- Θ (Theta) is a vector of all the population size parameters and migration rate parameters. Θ is not in the MCMC simulations, but rather is estimated using a sample of genealogies.
- G is a genealogy for a locus, or if a data set has multiple loci, it refers to a set of genealogies, one per locus.
- μ is a vector of mutation rate scalars, usually one per locus.

On a different note, 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.

1.4. Major changes from IMa2

The following are the main changes from IMa2:

- Implements phylogenetic topology estimation using a new hidden genealogy approach (`-j0`). Users can specify non-uniform topology priors if desired (see `-x`).
- Implements hyperprior distributions (`-j3`), which can be used to improve performance when estimating topologies, and for choosing priors when estimating Isolation-with-Migration parameters.
- Implements Message Passing Interface (MPI) allowing running on multiple cores (also in IMa2p).
- More runtime options for doing long runs with repeated command lines (`-r7`).
- Multiple changes to command line menu options.
- The priorfile format has changed.

1.5. Running IMa3, a Brief Overview

The program is run by typing and entering a command at a command line prompt. Running the program without any additional text on the command line will return help text to the screen. 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 as the name of the file.

The new command line term of greatest note is `-j0`, which tells the program to run an MCMC simulation over phylogenetic topologies. When this is used the output to the screen and results file, contains information on sampling of topologies, whereas when `-j0` is *not* used the output contains information on sampling of genealogies, splitting time values and mutation rate scalars.

By virtue of a parallel implementation, it is possible to analyze larger data sets than with earlier non-parallel IM programs, however this does not circumvent the challenges that arise with larger numbers of loci. Like its predecessors, the MCMC simulation of IMa3 implements some kinds of updates that apply to all genealogies in the simulation (one per locus per chain). So adding more loci has a large effect on the mixing process. In the current release of the program we have fixed the maximum number of loci to 200 (this can be changed - see `MAXLOCI` in ima.hpp). The maximum number of populations that can be run is 8 (9 with a ghost), and this cannot easily be increased. With this many populations there are a total of 1,587,600 different topologies (i.e. values of Φ). The maximum number of gene copies per locus is 1000. However the effect on the speed of the analysis of having large numbers of gene copies has not been well explored. In our hands, we rarely have more than 50 or 100 gene copies per locus.

For data sets with multiple loci it is usually desirable to run multiple heated chains simultaneously, and if the number of chains 10 or more, considerable time can be saved by running on multiple processors. Even so, run times can be quite long for large data sets. For example, for phylogeny estimation using a data set of 100 loci it is not uncommon to run on 20 to 40 processors for one or two weeks. When the phylogeny is known then IMa3 takes a similar amount of time as IMa2p, which is roughly the amount of time required of IMa2 divided by the number of processors used. These runs typically proceed more quickly than when estimating the phylogeny.

The relationship between data size (e.g. # of loci) and model size (e.g. # of populations that can be examined with the data) certainly depends on how much the populations have actually diverged, and

how much information is in the data, but otherwise is hard to predict. For data sets that show clear patterns of divergence for each of multiple loci, phylogeny estimation using IMa3 is probably unnecessary as many other methods will return the correct tree. We've had success estimating phylogenies with 7 populations that had a lot of gene flow using simulated data sets with 50 loci, and 5 gene copies per locus.

2. ESTIMATING Φ, THE TOPOLOGY OF THE PHYLOGENETIC TREE

IMa3 estimates the rooted ordered topology, where "rooted" means that the topology is oriented in time, and "ordered" means that the order of the internal nodes (branch points) of the tree are ordered in time. For example, three 4-population trees are shown in figure 1, with all sharing the same splitting times, while differing in topology. Note that the leftmost and rightmost trees have the same branching pattern, in so far as both have groups ((0,1),(2,3)), but they differ in that the tree on the left has (3,2) as the younger pair while the tree on the right has (0,1) as the younger pair.

Figure 1 also introduces the string formatting used for trees by IMa2 and IMa3. 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. For starters, the counting of populations begins with 0, and sampled populations are numbered 0 to `npops-1`, where `npops` is the number of sampled populations. The internal nodes of the tree correspond to ancestral populations, and these are numbered beginning with the number of populations, i.e. `npops` for the most recent ancestral population, and proceeds up to `2×(npops-1)` for the ancestor of all the sampled populations. Ancestral populations are represented by a colon, i.e. ':', followed by their ancestral population number. It is also important to understand that the meaning of an ancestral population that is identified by a particular number will change with the topology, as seen in Figure 1 where population 5 has different descendant populations in each of the trees.

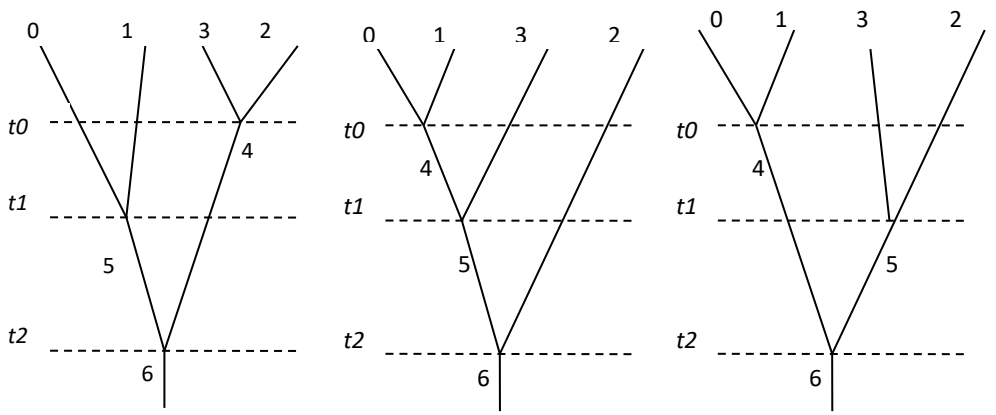

Figure 1

At any point in time during the MCMC simulation run by IMa3, there will be a current tree with a topology (e.g. as represented by the tree string) with splitting time values. The program is designed so that the trees that are generated by the simulation will be drawn from the posterior probability distribution. In other words, if D is the data and Φ is the topology, then the current value of Φ is a random draw from p(Φ|D). Then by recording lots of values of Φ the user can build a frequency distribution that will approximate p(Φ|D).

2.1. Getting good samples of Φ

The challenge, as with all MCMC methods, is obtaining a large number of sampled values that are independent of each other. In IMa2, and when running IMa3 on a fixed topology, the goal of the MCMC is to generate a good sample of genealogies (G) and values of the times of population splitting (τ), and the challenges of doing so are addressed in *Introduction_to_IM_and_IMA.pdf* and the IMa2 documentation. When sampling only Φ values the same issues apply, but they can be quite daunting for large data sets and broad priors. Actually assessing whether or not a run has burned in sufficiently, or that one has a good sample of topologies is difficult, and must usually be addressed by doing repeated runs, or by assessing changes in the sampled distribution during the run.

In order to monitor how well the MCMC simulation is mixing, IMa3 provides output to the screen at intervals, and it can be run in modes that provide output files at intervals.

2.1.1. RUNS WITH ONE CHAIN

For a data set with few loci (e.g. <5) it is worth trying a run with a single MCMC simulation (1 chain). In this case the first thing to watch out for is the number of updates of the topology. If this number rises into the thousands fairly quickly then it is possible that the run will be ok with a single chain.

2.1.2. RUNS WITH MULTIPLE CHAINS

For larger data sets we use multiple heated chains with Metropolis swaps between chains (Geyer 1991). To assess whether the number of chains and the heating values are a good fit to the data set, the user should check the number of updates to the topology in the output to the screen, or an output file. In general the swap rate among the least heated chains and between these and chain 0 (the cold chain) should be greater than 0.9. The scheme used to pick heating values builds a declining series of values that decrease very slowly at first for values near 1, and then more quickly as the most heated value is approached. (See `'-h Heating terms'`)

2.2. Runtime information when sampling Φ

During the run the following information will appear in the command window (or wherever screen output is redirected to):

- The current step number, the number of sampled values of Φ, the current values for the probability of the data, given the current genealogies, 'p (D|G)', and the prior probability of the genealogy 'p (G)'. Both of these probabilities are also of course conditioned on the value of Φ that exists in the simulation at the time they were recorded. They can jump around a lot because Φ is changing constantly. Therefore these probabilities are really only useful as a check, for example to see that they are actual numbers (e.g. non-numerical values indicate a floating point problem and the run should be halted).
- The value of the text string for the current topology (see Figure 1).
- The current value of τ (splitting times) that were sampled along with the current topology.
- A series of tables showing acceptance rates (update rates) for quantities in the MCMC simulation. (These are all for the cold chain, if multiple heated chains are used. Heated chains will usually have higher acceptance rates) :
    - The update rates for τ using an update method called `RannalaYang`, which is based on the method in Rannala and Yang (2003)).

- Update rates for the branches of the phylogeny using a branch sliding update. Rates are shown for changes that affect only branch lengths (i.e. effectively updates to τ), for changes that affect the topology *per se* (i.e. Φ), and changes that affect the time of the root of the phylogenetic tree (the TMRCA, which is the last element of τ).
- Update rates for genealogies, which are also branch slide updates. For these updates, some fraction of these branch slides also affect the topology and/or the genealogy TMRCA and the update rates for these are given as well.
- If the data have multiple loci, then update rates are also available for mutation scalars (μ), and these can be requested for runtime output to the screen using a command line flag (`'-r4'`).

- A table of Effective Sample Size (ESS) estimates and autocorrelations for Φ, for the sum of P(D|G) and P(G) (called L[P]), splitting times, and tmrca values of genealogies. This table begins to appear after 100,000 steps or so.
- If Metropolis-coupling has been invoked then an additional table appears with columns:
  - The heating terms for each pair of successive chains
  - The number of accepted swap proposals between successive chains (swaps between chains with non-successive heating terms are not shown).
  - The number of attempts
  - The rate of accepted swapping. This value will depend on the number of chains and the heating values. In general the closer are the heating values for two chains the higher will be the acceptance rate of proposed swaps between them. For large data sets with large numbers of chains, it seems best to have very high swap acceptance rates (e.g. > 0.95) between those chains nearest the cold chain.

2.3. Assessing burnin and mixing when sampling Φ

The MCMC simulation proceeds in two stages, including burnin, in which the simulation runs at least to the point that the current phylogeny is independent of the starting conditions, and sampling, in which values of the phylogeny are recorded at intervals. The success of both of these phases depend on how well the Markov chain is mixing, where 'mixing' is a catchall term referring to the rate at which the

simulation explores the state space. If mixing is poor, then the simulation spends long times in local areas of the set of possibilities, and successive samples are highly correlated.

It is possible for the simulation to appear to be mixing well on the basis of large numbers of phylogeny updates that include a wide variety of different topologies. However mixing of phylogenies is entirely dependent on mixing of the latent genealogies and hidden genealogies.

When estimating the phylogeny with IMa3, and particularly when there are more than 10 loci and more than three populations, the safest route to ensuring a good burnin is too simply observe the distribution of sampled values at intervals, and to wait until that distribution appears to not be changing (see below).

Similarly during the sampling phase, and particularly for larger problems, the safest route to ensuring a good sample is to first ensure a good burnin, and then observe the sampled distribution at intervals to see that it is no longer changing. Even then, it is quite important to repeat the entire process with a different random number seed, at least once, to see that you have obtained the same distribution.

2.3.1. ASSESSING BURNIN FOR Φ

There are two main ways to extend a burn until such time as the user feels that the state space is finally free of the starting conditions. One of these is to record a 'burntrend' file at intervals (filename will end with "…`burntrend.out`") . This option is invoked by using `-b` followed by a floating point value for the time in hours for the interval between the times when the program updates the burntrend file. (e.g. `-b 1.0` for every hour, or `-b24.0` for every 24 hours). The user must also ensure that there is a text file named 'IMburn' in the same folder with the data, and that that file begins with 'yes' (Deletion of the IMburn file, or deletion of the 'yes' in the file, causes the burnin period to end at the close of the current time interval specified by `-b`).

The second way to extend a burn is to restart a previously completed run using a checkpoint file that contains the state space at the end of that previous run. The creation of checkpoint files can be invoked by putting using `-r2` or `-r7` on the command line when starting the run. There will be one checkpoint file for each cpu used in the run. The files themselves end in ".mcf.#" where # is the cpu #.

To load a previously saved mcf file, the user can use `-r3` (together with `-f` to specify the file name) or `-r7` which causes the current output file name to be used as a base for the mcf file and if a file of that name is present at the beginning of the run, it will be loaded. The `-r6` option (together with `-r3` or `-r7`) causes the previously recorded values in the mcf file to be ifnored, and begins the MCMC

simulation fresh from the point at which it left off in the previous run. So by using `-r6` and `-r7` together it is possible to keep repeating a command line, and assessing mixing by looking at the output files, until such time as it seems that the actual sampling can begin (at which point `-r6` is no longer needed).

2.3.2. ASSESSING SAMPLE QUALITY FOR Φ

If the chain is mixing well then there should not be a strong autocorrelation for tree topology over the course of a sampling period. However the ESS values for topology are of limited usefulness, particularly when there are many populations in the model . Because the Φ are discrete and autocorrelations are being done on integers that range from 0 to the number of possible trees, it is possible for the autocorrelation to appear quite low simply because a number of different trees are being sampled. To counteract this the autocorrelations and ESS value for Φ (and the trends plot for Φ in the output file) is based on a Robinson-Foulds distance (Robinson and Foulds 1981) when there are 6 or more populations in the model. The Robinson-Foulds distance for a tree is calculated with respect to an any other arbitrarily picked tree (tree 0 in this case), and use of it is intended to provide a score that varies somewhat continuously. In practice, with these small numbers of populations, it does not work particularly well. When there 5 or fewer populations the autocorrelations are calculated using the tree number integers. The ESS values can be somewhat useful, however it must be kept in mind that a changing value of Φ can create the appearance of low correlation and high ESS values for L[P] and τ.

Another tool for assessing mixing for Φ is to compare the distribution of trees for the first half of the sample with that for the second half of the sample. These values ('freq_set1" and "freq_set2") in the table with sorted topologies and estimated posterior probabilities should be similar to each other and to "freq_ALL" if a large sample of effectively independent values have been obtained.

Users are encouraged to be quite conservative with respect to assessing mixing. Even if a the sample appears to be high quality, based on these criteria, it can be useful to extend the run (e.g. using –r7) and to repeat the run with a different random number seed.

3. ESTIMATING AN IM MODEL GIVEN A PHYLOGENY Φ

Once a phylogenetic topology has been estimated, that phylogeny can be entered into an input file and the user can do a run that does not include phylogeny updating. For that matter a user can run the file using IMa2, the results should be the same as for IMa3 if mixing is good and a large sample of genealogies is taken.

Like its immediate predecessors IMa3 implements two quite different suites of calculations when estimating an IM model, given a phylogenetic topology. One part runs the MCMC simulation that generates samples of genealogies (i.e. samples from the posterior probability density of genealogies, given the data and a value of Φ). The second part does the analyses on the function that is built from these genealogies.

The MCMC part of this must include a burnin period of sufficient length, 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. Typically the analyses are carried out by the program as soon as the sampling phase is complete. However if desired, and if the genealogies from a prior run were saved, the program can be run in a 'Load-Genealogies' mode (using `-r0`) to do additional analyses using the function generated from saved genealogies.

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 (see discussion in *Introduction_to_IM_and_IMA.pdf*). These are fundamentally the same issues that arise when using IMa3 to estimate Φ, however it can be easier to assess mixing when not sampling topologies, that is when running on a given value of Φ and sampling genealogies and splitting times.

3.1. Runtime information when sampling G and τ (but not Φ)

As when sampling Φ values, sampling genealogies and splitting times for a given value of Φ requires a burnin period, during which the state of the system must move to some point independent of the starting values, followed by a sampling period of sufficient duration that are large number of effectively independent values have been recorded. Generally we try to arrange the duration of a run so that we end up with at least 10,000 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 `ima.h`) depending on available memory.

During the run the following information will appear in the command window (or wherever screen output is redirected to, depending on the environment):

- The current step number, the number of sampled genealogies, 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 current value of τ (splitting times).

- The update rates for τ and G. 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 a run sampling values of G, update rates are also available for mutation scalars (μ), and these can be requested for runtime output to the screen using a command line flag ('-r4').

- 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 an additional table appears with columns:

    - The heating terms for each pair of successive chains

    - The number of accepted swap proposals between successive chains (swaps between chains with non-successive heating terms are not shown).

    - The number of attempts

- The rate of accepted swapping. This value will depend on the number of chains and the heating values. In general the closer are the heating values for two chains the higher will be the acceptance rate of proposed swaps between them. For large data sets with large numbers of chains, it seems best to have very high swap acceptance rates (e.g. > 0.95) between those chains nearest the cold chain.

3.2. Assessing mixing and run duration when sampling G and τ (but not Φ)

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 auto-correlated 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 be run to improve mixing over the course of a run. In general, multiple chains are recommended or required when there are five or more loci.
- Multiple independent runs should be conducted using identical priors and number of coupled chains, but different random number seeds. If the separate runs give similar

results, then it is probable that both have provided useful samples, and the genealogies from both sets of runs can be combined in a single Load-Genealogies run for the analysis.

When starting out 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.

If you are running many chains, remember that the total update rate for any term in chain 0 (cold chain from which samples are taken) includes the rate of swapping with other chains. If splitting times are updating at higher rates in higher numbered chains (as they typically will be), then it may be the case that splitting times for chain 0 have an overall update rate that is acceptable. The trend plots will tell the tale.

3.3. Load-Genealogy mode runs

If the program is run using `-r0` and one or more files of previously saved genealogies are loaded, then the only output to the screen is an occasional indication of which analyses are being done. Load-Genealogy 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 several days or more. The slowest analyses are estimations of joint-posterior densities and likelihood-ratio tests of nested models.

3.4. Parameter Conversions - Obtaining Demographic Parameter Estimates

In most situations users will have provided generation time in years on the command line (`-u`) and absolute mutation rates per year in the input file, and there won't be a need to deal with parameter conversions (which can be confusing). The following text is copied, with only slight modifications, from the *Introduction_to_IM_IMa* document:

> The IM programs can be used to generate estimates of model parameters (θ_1, θ_2, θ_A, m_1, m_2 and t). Typically the peaks of the estimated distributions are taken as the estimates, just as if one was taking a maximum likelihood estimate. Indeed, because the method uses uniform prior distributions, these estimates are maximum likelihood estimates in those cases when the prior distributions are very wide and the tails of the posterior distributions are clearly contained within the range of the prior distributions (i.e. as if the prior really was uniform over the range of zero to infinity).

Once the model parameter estimates are in hand, many investigators will wish to also generate estimates of the demographic quantities (i.e. N_1, N_2, N_A, t, m_1, and m_2). Most of these conversions can be done automatically by the program, provided that mutation rate estimates are included in the input file (see the section on Input File Format and command line flags for the program you are using). Whether this is done by the program or by the investigator, it is important to understand what is being done here, as the subject can be a bit confusing. The next few paragraphs explain how these calculations are made.

First, note that that all of the parameters in the model include the mutation rate u (which is a value for the gene, not per base pair). If you have multiple loci, then u is the geometric mean of the mutation rates of all the loci. The method for converting parameter scales is explained for diploid autosomal loci for the parameters *t*, θ_1, and m_1. The same approach is readily applied to θ_2, θ_A, and m_2 in the same way.

First, gather your model parameter estimates, and an estimate of actual mutation rates (using an outgroup or some other relevant data).

- Let ***A*** be your estimate of θ_1 (i.e. $4N_1u$ where N_1 is the effective size of population 1)
- Let ***B*** be your estimate of *t*, the time parameter (i.e. t u, where t is the time since splitting)
- Let ***C*** be your estimate of m_1 (i.e. m_1/u). It is important to understand that m_1 is the rate per gene per generation from population 1 to population 2, *in the coalescent*. Since the coalescent goes backwards in time, m_1 is more easily thought of as the rate at which genes come into population 1, from population 2, as time moves forward.
- Let ***U*** be an estimate of the mutation rate per year for the gene being studied. (not per base pair, but for the entire gene). This must usually be obtained using some other data, such as distance from an outgroup of known time separation. If you are doing multiple loci, then ***U*** is the geometric mean of the mutation rates (per year) for the loci.
- Let ***V*** be an estimate of the mutation rate per generation for the gene being studied. If ***G*** is the number of years per generation, then ***V*** = ***U G***.

Now generate estimates of demographic quantities:

- To estimate the effective population size, N_1, calculate $\boldsymbol{A}/(4\,\boldsymbol{V})$. This is because N_1 is defined in the coalescent models as being proportional to the inverse of the coalescent rate per generation. Therefore we need to use ***V*** since it is an estimate of the mutation rate on a scale of generations.
- To estimate the time since splitting, t, in units of years, take $\boldsymbol{B}/\boldsymbol{U}$.
- To estimate the time since splitting in generations, take $\boldsymbol{B}/\boldsymbol{V}$.
- To estimate the migration rate per generation, m_1, take $\boldsymbol{C} \times \boldsymbol{V}$.
- To estimate a migration rate per year, take $\boldsymbol{C} \times \boldsymbol{U}$.
- To estimate the population migration rate (the effective rate at which genes come into a population, per generation) for population 1 (i.e. $2\,N_1\,m_1$) you don't even need the estimate of mutation rate, Since $4N_1u \times m_1/u\,/\,2 = 2N_1\,m_1$, all you need to do is take $\boldsymbol{A}\times\boldsymbol{C}/2$.

When multiple loci are studied, the program also allows estimation of mutation rate scalars for each locus. If x_i is the estimate of the scalar for locus i, then an estimate of $4N_1u_i$ can be obtained by taking $\boldsymbol{A} \times x_i$.

If data from multiple loci are used, but per-year mutation rates are only available for a subset of them, then it is still possible to generate estimates of demographic quantities. The trick is to make use of the estimates of the mutation rate scalars for those same loci. Let ***X*** be the geometric mean of the estimates of the mutation rate scalars for just those loci for which per-year mutation rates are also available. Let ***U*** be the geometric mean of the per-year mutation rates *for just those same loci*, and let $\boldsymbol{V} = \boldsymbol{U}\,\boldsymbol{G}$. Then $\boldsymbol{A}\,\boldsymbol{X}\,/(4\,\boldsymbol{V})$ is an estimate of N_1. Similarly an estimate of the number of years since divergence began is obtained with $\boldsymbol{B}\,\boldsymbol{X}/\boldsymbol{U}$. An estimate of the migration rate per generation per gene copy is obtained with $\boldsymbol{C}\,\boldsymbol{V}/\boldsymbol{X}$.

If you are working with multiple loci that have mixed inheritance models, then you will want to be setting the inheritance scalars in the input file (see Input File Format). In this case the method described above can be applied in exactly the same way without changes.

Similarly if you are working with a single locus, and the inheritance scalar in the input file is not one (e.g. if the data is mtDNA and the scalar is set to 0.25), then the effect of that scalar is to yield results that would be comparable to a diploid autosomal locus, and again the method described above can be applied in exactly the same way without changes.

However if you are working with a single locus that does not have diploid autosomal inheritance, but the inheritance scalar is set to 1 in the input file. Then the estimates of θ_1, θ_2, and θ_A are not of $4N_eu$ but rather of the product of $4N_eu$ and whatever the true inheritance scalar actually is for that locus.

4. COMMAND LINE OPTIONS

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

```
IMa3 2018 Jody Hey, Rasmus Nielsen, Sang Chul Choi, Vitor Sousa, Janeen Pisciotta, Yujin Chung and
    Arun Sethuraman
This program is run using a command line interface
To execute, type the program name followed by command line flags and options
If compiled with MPI_ENABLED,  IMa3 can be run with multiple cpus
-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 -r0 -w
    3 Get prior distribution terms from file (requires filename given with -g )
-d  Number of steps between sampling genealogies (default 100)
-f  Name of file with saved Markov chain state generated in previous run (use with -r3)
-g  Name of file for parameter priors, default: 'imapriors.txt'
-h  Heating terms (MCMC mode only):
  -hn Number of chains
  -ha First heating parameter (less than, but near 1.  The more chains, the closer to 1)
  -hb Second heating parameter  (the smallest Beta value)
-i  Input file name (no spaces)
-j  Model options:
    0  Population Topology Updating and Estimation (for 3 or more populations)
    1  Add a non-sampled ghost population to the model
    2  Migration prior follows exponential distribution with mean given by -m or priorfile
    3  Use hyperpriors. -q and -m specify hyperprior distributions (if not -j0, priorfile is created)
    4  Migration only between sister populations (do not use with -j0)
    5  One migration parameter for each pair of populations (do not use with -j0)
    6  Migration only between sampled populations (do not use with -j0)
    7  Separate population size and migration parameters in each period (do not use with -j0)
    8  No migration in the model (do not use with -j0)
    9  One single migration parameter for all pairs of populations (do not use with -j0)
    x  One single population size parameter for all populations (do not use with -j0)
-L  Run duration. Duration depends on integer or floating point value and other settings
     if integer value, run duration depends on value and other settings
       If estimating population phylogeny (-j0), # of phylogenies to save, 10 steps per save
       If fixed phylogeny in MCMC mode, # of genealogies to save
         This value times -d value sets the # of steps in the chain after burnin
       If fixed phylogeny,LOAD-GENEALOGY mode (-r0, -v), # genealogies to load from file(s)
        If floating point: value = the time in hours between outputs.
          Run continues until file IMrun is absent from directory, or does not begin with 'y'
-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 splitting times divided by prior (do not use with -j0)
    4 Turn off printing estimates and histograms of population migration rate (2NM)
    5 Print pairwise probabilities that one parameter is greater than another
    6 Print histograms of the number of migration events (do not use with -j0)
    7 Print joint estimate for splitting times (not with -j0, models with 3, 4 or 5 populations)
-q  Maximum for population size parameters (4Nu)
-r  Run options
    0 LOAD-GENEALOGY Mode - load from previous run(s); also requires -v (do not use with -j0)
    1 Do not save genealogies (default saves sampled genealogies, ignored with -j0)
    2 Save the state of the Markov chain in a file - named with extension .mcf (MCMC mode only)
    3 Start by loading *.mcf file; requires -f (only the state space is loaded)
    4 Write all mutation related updates rates to stdout during the run (default is to suppress)
    5 Print burntrend file at end of burnin period
    6 When loading mcf files (-r3,-r7) do not load sampled values (i.e. use previous run as burnin)
```

```
   7 Load *.mcf file if present AND save to *.mcf file.
      No burn done if *.mcf is present. Use to continue a run by repeating the same command line.
-s  Random number seed (default is taken from current time)
-t  Maximum time of population splitting
-u  Generation time in years (default is 1)
      Only needed if demographically scaled histograms are desired in fixed phylogeny runs
-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 (requires use of -r0), invokes –c2
-x  Set phylogeny priors for pairs of sampled populations (requires -j0)
      e.g. -x 0 1 0.5  set relative prior on all trees with 0,1 as sisters to 0.5 (default is 1)
      can have multiple -x terms in command line
-y  Mutation rate scalar for relevant loci
      Only needed if demographically scaled histograms are desired in fixed phylogeny runs
      Only needed in LOAD-GENEALOGY Mode (-r0) and if mutation rates not known for all loci
-z  Number of steps between screen output (default is 10000) (don't use with LOAD-GENEALOGY mode -r0)
```

4.1. 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 the number of migration events (`-p6`), 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. The IMa2 documentation has some additional information about selecting these ranges.

`2 Joint posterior density calculations, for LLR tests of nested models, use with -r0 -w`

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 Load-Genealogy 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 some model options (i.e. with most –j options 4,5,6,7,9,x).

-d Number of steps between sampling genealogies (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 a specific number of sampled genealogies is requested (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 (use with –r3)

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 sampling runs (whether of topologies or genealogies) 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. For chain i (where i goes from 0 – the cold chain – to n-1, where n is the number of chains) , 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.

For runs that sample τ and G (but not Φ) lots of trial and error suggests that a heating scheme should be selected that provides swap rates of between 0.4 and 0.8 between chain 0 and chain 1, and generally between chains i and i+1.

For runs that sample Φ it seems necessary to have a very gentle gradient of heating terms so that swapping rates among low numbered chains (i.e. the cold chain and others with only a little heating) have very high swap rates (preferably > 0.9).

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. The trick is to get sufficient swapping among low numbered chains (particularly with chain 0), and yet also have sufficient heating in high numbered chains, so that the heated chains do really explore the parameter space.

Unlike previous programs, IMa3 provides only a single scheme for specifying heating values based on –ha and –hb values. This scheme (called the 'Geometric model' in IMa2) is specifically intended for large numbers of chains where very small differences are needed between lower numbered chains. Two terms are required. *ha*, given by '-ha' on the command line, which specifies the degree of non-linearity, where 0<*ha*<=*1*. The case of *ha*=1 gives a linear decline and *ha*<1 gives β values that decline slowly for low numbered chains, and fast for high numbered chains. The second term, *hb* 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

$$\beta_i = 1 - \frac{(1-hb) \times i \times ha^{n-1-i}}{n-1}$$

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

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

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

`-hn Number of chains`

The number of chains. This must be at least 2 per cpu. If multiple processors are being used then this must be a multiple of 2 or higher of the number of processors. For example, if 10 processors are being run, then valid –hn values are 10,20, 30 … etc. The maximum value is given in the `ima.h` header file in the source code if a user needs to change it.

`-ha First heating parameter (less than, but near 1. The more chains, the closer to 1)`

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

`-hb Second heating parameter (the smallest Beta value)`

The second heating term specifies the lowest Beta value (see above).

`-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). Note that the numbering is quite different from the same or similar options in IMa2.

`0  Population Topology Updating and Estimation (for 3 or more populations)`

This invokes updating and sampling of Φ (the topology of the population phylogeny). Once the burnin is complete values of Φ are sampled every 10 steps. Topology sampling requires a basic underlying Isolation-with-Migration model in which all sampled and ancestral populations have a population size parameter, and all pairs of populations that co-exist have two migration parameters (one each direction). This means that some IM models with reduced parameter sets (e.g. `–j4` and others) will not run with `–j0`.

`1  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.

`2  Migration prior follows exponential distribution with mean given by -m or priorfile`

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.

`3 Use hyperpriors. -q and -m specify hyperprior distributions (if not -j0, priorfile is created)`

IMa3 has an important feature in that users can work with hierarchical priors for the population size and the migration rate parameters. To do so the user specifies the hyperprior probability density from which the terms of the prior distributions will be sampled. Then the actual priors are included in the state space of the MCMC simulation and are updated using Metropolis-Hastings updates just as are other elements of the state space.

There are two benefits two using hyperpriors. One is that it allows the prior distributions to adjust to the data and to the model, and this can improve mixing and, in our limited experience, higher posterior probabilities for the phylogeny that fits the data best. The second benefit is that by specifying hyperprior densities the investigator is partly freed from having to carefully match the priors to the data (a messy and not easily justified practice). When `-j3` is invoked the `-q` and `-m` terms apply, not to the prior distributions, but to the hyperprior distributions.

For population size terms, with `-j3` the number specified with `-q` becomes the upper bound on uniform distribution (with lower bound 0). Then each individual population size parameter has a uniform prior distribution (lower bound 0), the upper bound of which is drawn from this uniform hyperprior distribution.

For migration parameters, using `-j3` uniform hyperpriors (the default) things work just as they do for population size terms. If exponential priors are specified for use with migration parameters (`-j2`) and hyperpriors are used, then both the hyperprior density and the individual prior densities follow exponential distributions. Specifically, with `-j2` and `-j3`, the value following the `-m` term is the mean of an exponential hyperprior distribution. Each individual migration parameters has an exponential distribution with a mean that is drawn from this hyperprior density.

Invoking `-j3` causes tables with update rates of population size and migration priors to appear in the screen output and in the main output file.

When `-j3` is invoked without using `-j0` (i.e. when sampling genealogies and splitting times, with a fixed topology) then trend plots and asci curves approximating the posterior density of each of the priors for the given topology will appear in the output file. Because there are not set prior distributions the program will not estimate the joint or marginal densities of migration and population size parameters under these circumstances. However IMa3 will generate a priorfile format that contains suggested values for `-q` and `-m` (the spitting time priors are set to the same as specified with `-t`), based on the estimated posterior distribution of priors, to use in a run that does not use either hyperpriors (i.e. without either `-j0` or `-j3`).

When generating a priorfile, which happens when `-j3` is used without `-j0`, the (highly arbitrary) rules for generating the suggested priors are as follows:

With a uniform hyperprior (either population size or migration rate), the method (that is applied to each individual parameter prior) to get is as follows:

- Let the suggested prior that is to be determined (i.e. the upper bound of the prior distribution for the parameter) be called 'sugpr'
- Identify the upper bound of the hyperprior distribution (specified using `-q` on the command line), call it 'ub'.
- Identify the prior value with the highest estimated posterior probability, call it 'prmax'
- Identify the value of the prior to for which the estimated cumulative probability is 0.8, call it 'prcump8' (i.e. prcump8 is the value x for which it is estimated that prob(prior <= x|Data) = 0.8).

- Then follow this short conditional:
    - If prmax > 0.8×ub : sugpr = ub
    - else:
    - if prmax < prcump8: sugpr = 0.8×ub.
    - else: sugpr = prmax

With an exponential hyperprior (i.e. for migration rates when both `-j2` and `-j3` are invoked)), the method (that is applied to each individual parameter prior) to get is as follows:

- Let the suggested prior that is to be determined (i.e. the upper bound of the prior distribution for the parameter) be called 'sugpr'
- Identify the prior value with the highest estimated posterior probability, call it 'prmax'
- Identify the value of the prior to for which the estimated cumulative probability is 0.1, call it 'prcump1' (i.e. prcump1 is the value x for which it is estimated that prob(prior <= x|Data) = 0.1).
- Then follow this short conditional:
    - If prmax < prcump1: sugpr = prcump1
    - else: sugpr = prmax

Of course users can make their own priorfiles. The histograms for the priors are in the output file and are valid estimates of the posterior probability densities for the priors.

`4 Migration only between sister populations (do not use with -j0)`

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.

`5 One migration parameter for each pair of populations (do not use with -j0)`

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 this also requires turning off estimates of 2NM (-p4).

`6  Migration only between sampled populations (do not use with -j0)`

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.

`7  Separate population size and migration parameters in each period (do not use with -j0)`

An alternative parameterization for a multi-population (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.

`8  No migration in the model (do not use with -j0)`

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

`9  One single migration parameter for all pairs of populations (do not use with -j0)`

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.

`x  One single population size parameter for all populations (do not use with -j0)`

All populations have the same effective population size parameter value

`-L  Run duration. Duration depends on integer or floating point value and other settings`

If the value has a decimal point it is interpreted as a time, in hours, it specifies the duration of the sampling period (after any burnin period) . 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.

If `-L` is followed by a floating point value and topologies are being sampled (`-j0`), then one topology is sampled every 10 steps of the run. If genealogies are being sampled, then the number of steps between sampling of genealogies is given by the `-d` option.

If `-L` is followed by an integer (e.g. `-L 10000`) then this value is the number of times the chain is sampled (topologies or genealogies, depending on the presence of `-j0`) after the burnin is completed.

If the program is run in Load-Genealogies mode (`-r0`) : 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 genealogy sampling 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 Load-Genealogies 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 (-j2) 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 –j8). If the user wishes to set priors individually for each parameter then a priorfile should be used (see –c3).

If hyperpriors are invoked (-j3), then –m sets the upper bound of the hyperprior distribution, or in the case of exponential priors (-j2) it sets the mean of an exponential hyperprior distribution.

```
-o  Output file name (no spaces) default is outfile.txt
```

```
-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). The numbering is different than in IMa2.

```
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. However they do take up space and sometimes it is simpler to not create them.

```
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)
```

This can only be used when sampling genealogies. 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 splitting times divided by prior (do not use
      with –j0)
```

When there are two or more splitting times in the model the marginal prior distributions are not uniform (Hey 2010). 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.

```
4 Turn off printing estimates and histograms of population
      migration rate (2NM)
```

By default IMa3 will provide estimates of the posterior density for the population migration rate associated with each migration parameter. These calculations can take a while (sometimes hours in a big model with lots of genealogies), and so this can be turned off if desire. 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, $2N_iM_{i\rightarrow j} = 4N_iM_{i\rightarrow j}/2 = 4N_iu\times(m_{i\rightarrow j}/u)/2$ (Hey 2010). Considering this forward in time it is the rate at which population *i* receives migrants from population *j*.

```
5 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.

```
6 Print histograms of the number of migration events (do not use
      with –j0)
```

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 (not with –j0, 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. If the user wishes to set priors individually for each parameter then a priorfile should be used (see `–c3`). If hyperpriors are invoked (`-j3`), then `–q` sets the upper bound of the hyperprior distribution.

`-r Run options`

```
0 LOAD-GENEALOGY Mode - load genealogies from previous run(s); also
      requires –v (do not use with –j0)
```

This causes the program to run in Load-Genealogy mode. There is no MCMC simulation, no burnin, and no genealogy sampling. Rather the program loads information on previously sampled genealogies that are contained in .ti files, as specified using `–v`.

```
1 Do not save genealogies (default saves sampled genealogies,
      ignored with –j0)
```

Prevents creation of .ti file(s) with genealogy information. This is ignored with `–j0`.

```
2 Save the state of the Markov chain in a file - named with
      extension .mcf.
```

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. One file is saved for each cpu used. If only one cpu, the file ends in '.mcf.0'.

```
3 Start by loading *.mcf file; requires -f (only state space
      loaded))
```

This option causes the program to begin by loading a previously saved state space file or files. The run should be started with the same number of cpus, the same data file, same priors, same data file, and same heating terms as used in the original run that generated the '.mcf' file. Only the state space of the previous run is loaded, not any sampled values. This is useful for starting a new chain as if the burnin had just been completed or for adding an additional burnin period using –b.

```
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 update information and 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.

```
6 When loading mcf files (-r3,-r7) do not load sampled values (i.e.
      use previous run as burnin)
```

This option causes the program to load a saved mcf file from a previous run, however only the state space at the time the previous run ended will be restored. All of the data structures for recording sampled values will be initialized as if the sampling phase is just beginning. This option allows the user to treat a previous sampling run as a burnin run.

```
7 Load *.mcf file if present AND save to *.mcf file.
```

This option can be used to continue runs using the same command line. It is especially useful for running IMa3 on systems that have a maximum time that a job can be run. In such cases a run can be continued an indefinite number of times using the exact same command line

repeatedly, with the set of sampled values growing with each repeat of the command. Typically `-r7` is to be used after a burn has been completed (`-b` is ignored when `-r7` is used). If a .ti file exists it will be added to, and in general repeated runs with –r7 will accumulate samples from run to run.

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

This is an integer (e.g. '`-s 1731`'). It is useful to specify a value for two reasons: first if you want to repeat an identical run; and second if you are starting multiple separate replicate runs on the same data set, in which case 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.

Importantly, If multiple cpus are being used, then there is no guarantee that reusing a random number seed will generate an identical run.

`-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 - (default is 1)`

Specify the generation time. This is used for demographically scaled estimates of splitting times of population sizes.

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

When running in Load-Genealogy 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 (requires use of -r0), 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).

`-x  Set phylogeny priors for pairs of sampled populations (requires -j0)`

By default IMa3 assumes a uniform prior over all ordered rooted topologies. By using this option users can adjust the topology prior for pairs of populations. For example, if populations 0 and 1 are considered to be half as probable to be sister populations, compared to other populations, then include `–x 0 1 0.5` . If populations 2 and 5 are considered to be twice as probable to be sister populations compared to other pairs, then include `–x 2 5 2.0`. Users can include an additional `–x` and associated terms for every pair of populations considered to have a non-uniform prior of being sisters.

`-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 a Load-Genealogy mode run with a mean value obtained from a previous run in which genealogies were sampled. If all loci in the data file have a mutation rate provided in that file, then use `–y1`.

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

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).

5. INPUT FILE FORMATS

IMa3 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.

5.1. Data File Format (Unchanged from IMa2)

The format for data files for IMa3 is the same as used for IMa2:

- 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 – a string of text representing the phylogenetic tree for the populations. If not estimating the phylogeny, then it is very important to get this right. If phylogenetic topology is being estimated then this line is ignored.

    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 (see Figure 1). 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. 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`.

If estimating the phylogenetic topology, the tree string specified in the input file will be ignored. And if desired the user can simply not include the string in the input file for a `-j0` run (i.e. delete the complete line where the tree string would be, do not leave an empty line).

- 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)
    - `n0` thru `nnpops-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+n0+n1+... +nnpops-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 IMa2 or IMa3
# 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
```

5.2. Parameter Prior File Format (Changed from IMa2)

We call a file containing the terms for the prior distributions of model parameters a 'priorfile'. A priorfile is used for tailoring the prior distributions for specific parameters (i.e. to allow specific values for individual parameters as opposed to using `-q`, `-m` and `-t` which apply to all parameters in the model). A priorfile is not used when sampling topologies (`-j0`). 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.
- There are four keywords: `tree`, `theta`, `migration` and `time`. Each must appear exactly once in a priorfile as the only word on a line. The first non-comment line of the file must have `tree` on it. After the line with `tree`, the order of lines with keywords does not matter. Following a keyword line, the next line(s) has prior information (see below).
- The first non-comment line following the line with `tree` 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 first non-comment line following the line with `time` 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 first non-comment line following the line with `theta` 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

- Following the line with `migration`, 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 $m_{0\to1}$, $m_{1\to0}$, $m_{0\to2}$, $m_{2\to0}$, $m_{0\to3}$, $m_{3\to0}$, $m_{1\to2}$, $m_{2\to1}$, $m_{1\to3}$, $m_{3\to1}$, $m_{2\to1}$, $m_{3\to2}$, $m_{0\to4}$, $m_{4\to0}$, $m_{1\to4}$, $m_{4\to1}$, $m_{4\to5}$, $m_{5\to4}$. However there can be no parameter $m_{3\to4}$ (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.

IMa3 will generate a priorfile when topologies are not being sampled and when hyperpriors are used (see `-j3`).

5.3. Nested Model File Format (Unchanged from IMa2)

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.

6. OUTPUT FILES

The program generates up five main types of output files, including: the main results file, genealogy files (ending in .ti), Markov chain state file (ending in .mcf extension); migration histogram files (ending in .mpt extension); and burntend files for update rates and sampling trends during the burnin period. In addition, if using hyperpriors without sampling topologies, the program will generate a priorfile (see `-j3`).

6.1. Main Results File

The output of the main result file is somewhat similar to that generated by IMa2, with sections summarizing the options, the data, the mixing, parameter estimates, trend and density plots, and histograms. Depending on the command line options the sections that are included will vary:

`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.

`SAMPLING SUMMARIES`

This is a short section that gives information on the duration of the run, in terms of numbers of steps, length of burnin, and numbers of samples taken. If sampling topologies (`-j0`) this also gives the numbers of updates to the topology in the cold chain. This number needs to least be in the thousands to have any hope that the simulation is mixing.

`MCMC INFORMATION`

This section includes tables of MCMC update rates for the cold chain. In each of these tables is listed the quantity being updated, the type of update, the number of attempted updates ('#Tries'), the number of accepted updates ('#Accp'), and the acceptance rate ('%'). If multiple chains have been used there will be a table of swapping rates between successive chains (i.e. adjacent heating values). 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**.** This section is absent in Load-Genealogy mode.

ESTIMATED POSTERIOR PROBABILITIES OF POPULATION TREE TOPOLOGIES

This section appears only if topologies are being sampled (`-j0`) . It lists the sampled topologies sorted by sampled frequency for all trees that were sampled at least once. Also shown are sampled frequencies in the first half of the sample and the second half (set1 and set2). Comparison of these two values can be very useful for assessing mixing. Also listed is the prior probability for each tree (which will be 1 unless –x is used on the command line) and the product of clade posterior probabilities for each tree (ppcp). If the number of possible trees is not too large, then following this table will be a similar one of unsorted trees, including those that were not sampled.

MEANS, VARIANCES AND CORRELATIONS OF PARAMETERS ('$' R > 0.4 '*' R > 0.75)

This section appears only with sampling genealogies with fixed priors. It does not appear when using hyperpriors (`-j3`) or when sampling topologies. It provides a table of the mean and standard deviation of the marginal posterior probability distribution for each of the population size parameters and migration rate parameters (denoted by an m followed by the source population, an arrow >, and the target population). Population size parameters are denoted by a q followed by the population number. Migration rate parameters are denoted by an `m` followed by the source population, an arrow, >, and the target population number (the arrow direction refers to the migration path backwards in time). Also provided is a table of correlations between all pairs of population size and migration rate parameters. 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 is that if two parameters are very strongly correlated, then there is in effect only one parameter.

`ESTIMATED POSTERIOR PROBABILITIES OF POPULATION TREE TOPOLOGIES`

This section appears only when sampling topologies (`-j0`). It lists the populations in the model, the topology prior, a summary of sampling information, including the most frequently sampled topology and the topology with the highest product of posterior clade probabilities (ppcp). Also provided are a list of topologies, ppcp values, and counts sorted by counts, and an unsorted list of all topologies with counts.

`Marginal Peak Locations and Probabilities`

This section provides estimates of population size and migration rate parameters. It appears only with sampling genealogies with fixed priors. It does not appear when using hyperpriors (`-j3`) or when sampling topologies.

A set of tables are provided that give the parameter estimates 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, migration rate parameters, and population migration rates (i.e. *2NM* terms). For each parameter set in each period, the parameters are listed as a table, with each parameter heading a column. 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:

- 'Parameter' 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 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.
- 'Pmax' is the highest posterior marginal posterior density observed for that parameter.
- 'LR95%Lo' is the 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 (2010) . Statistical significance is indicated by asterisks, or lack thereof by 'ns'. These tests are prone to false positives when actual divergence is quite low and the amount of data is not large (Hey, et al. 2015). For *2NM* values, the reported significance level for the LLRtest results is actually copied from the LLRtest for the corresponding migration parameter. In other words, a *2NM* value is considered to be significantly different from zero if the migration component of that *2NM* value was significantly different from zero.
- '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

If the option to get joint estimates and do LLR tests of nested models (`-c3` together with `-r0`) is invoked then IMa3 will generate a joint parameter estimate. This can only be used for two or three population models. This table will 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 generating figures if desired. 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 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 Load-Genealogy mode, trends are provided only for the Log(P) values and the splitting time values that are saved in the '.ti' files.

If topologies are sampled (`-j0`) a trend plot will appear showing topology number on the y axis. Because this is a discrete quantity, and two topologies with adjacent numbers can be very different, this plot is of limited usefulness with more than 4 populations (5 populations have 180 topologies). When there are 6 or more populations, topology numbers are replaced by the Robinson-Foulds distance from topology 0.

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 Load-Genealogy 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 Load-Genealogy mode run will generate ASCII plots

using the splitting time values that correspond to the genealogies that were used from the .ti files.

6.2. Genealogy (.ti) file

If genealogies are being sampled, 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 a Load-Genealogy 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.

6.3. Markov chain state file (.mcf)

If the state of the Markov chain is being saved at the end of a run, so as to allow restarting a new run from that same point, then there will be a file with a '.mcf.#' extension (where # is the cpu number, i.e. 0 if only one cpu). This file can be loaded at the start of another run so long as the alsl the basic run settings are the same (i.e. the data file, the number of cpus, the number of chains, and the priors). 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.

6.4. Migration count histogram file (.mpt)

If histograms are generated for the number of migration events (`-p6`), then these results are written to a file with a '`.mpt`' extension.

6.5. Burnin tend plotfile

If trend plots are generated during at the end of the burnin phase (`-r5`), these are saved in a file that ends in '`.burntrend.out`'. This file contains update summaries and trend plots. In the case of sampling topologies (`-j0`) a listing of observed topology counts during the burnin is also provided, and if a sequence of burntrend recordings is done (i.e. using `-b` followed by a floating point value and in the presence of an `IMburn` file) a listing of observed counts for each burnin interval (up to 10 intervals) will be listed and these can be used to assess mixing.

7. GETTING STARTED, AND SUGGESTIONS FOR RUNNING IMA3

Just as when running previous IM programs running IMa3 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.

7.1. 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) Decide what overall model you want to apply to the data (e.g. sample topologies? Use hyperpriors?); (3) Decide upon priors or hyperpriors; (4) Do preliminary runs to assess mixing and set heating terms and numbers of cores; and (5) 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). These numbers of essential for getting splitting time estimates in years and effective population size estimates.

f. If you don't intend to estimate the phylogeny, and 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. Deciding on the overall initial model requires asking two main questions: will you first estimate the phylogenetic topology or do you know it already?, and will you use hyperpriors or priors? Answering 'yes' to the first obviously means doing topology sampling runs (`-j0`). Answering 'yes' to the second questions means using hyperpriors (`-j3`). In both cases the user can anticipate having to run multiple models. For example, an initial topology sampling run with hyperpriors (`-j03`) may need to be followed by a run fixed on the estimate topology but still using hyperpriors (`-j3`), followed finally by a genealogy sampling run with priors set to values that resulted from the previous hyperprior run.

3. The next step is to figure out what values to use for the upper bounds on your prior distributions or hyperprior 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.

The prior distributions can have a very large impact on the analysis, both in a conventional Bayesian sense of shaping the posterior distribution, but also (and usually more importantly) in affecting the speed of the analysis. If the upper

bounds of the priors are too low, then the parameter estimates might be strongly affected by the prior in a way the user did not intend. On the other hand the higher the upper bounds of the priors the larger is the space of genealogies that have high prior probabilities, which (unless the data strongly dominates over the prior) effectively expands the state space of the MCMC simulation. This means slower mixing and a longer run. This issue can be particularly acute with migration. If the user sets a migration upper bound that is too large, it is possible for the simulation to spend a lot of time on genealogies with very large numbers of migration events in them.

Hyperpriors should probably always be used with sampling topologies. This is because the simulation will explore many configurations of ancestral populations and pairs of populations, each of which will draw upon corresponding prior distributions for the internal calculations. However it is possible that there are data sets where not using hyperpriors works better.

Hyperpriors offer the great advantage that the user does not need to be as careful optimizing or carefully selecting prior distributions, and so it may also be useful to use hyperpriors for a fixed topology. The downside is that, when using hyperpriors on a fixed topology, that the genealogies are actually not sampled (splitting times and prior values are sampled) and the population sizes and migration rates cannot be estimated without an additional run using priors selected on the basis of a previous hyperprior run.

In general users should follow the instructions (below) for picking priors, and then, because these values are intended to be somewhat liberal, if they wish to do a hyperprior run, they can use the same values to specify the hyperprior distributions.

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.

4. Once the data, model and priors are set it is useful to do a very short 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 –b100 –L100 –s123.` If `–j0` is included topologies will be sampled, in which case `–L100` specifies that 100 topologies will be sampled. If the topology is fixed and genealogies are sampled then `–L100` specifies that 100 genealogies will be sampled, which means that following the burning the run will proceed for 10000 steps (i.e. the default number of steps between saving

genealogies is 100). 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.

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 makes sense to try 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).

It is important to observe the swapping rate between chains. In general the rate between chains 0 and 1 should be > 0.9 and the rates between the most heated chains should be >0.1. Making the –ha value closer to 1 will flatten the curve of heating values (i.e. heating value is closer to be a linear function of chain number). The –hb value is the heating term for the most heated chain.

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 are sampling topologies, then you should increase these numbers, and if you have microsatellite data, then you will probably need a great deal more heating than suggested here:

- Very small data sets (< 4 loci < 20 individuals per locus) : data sets of this size might do ok without multiple chains.
- Small data set (e.g. < 15 loci, < 20 individuals per locus), low heating:

  `-hn20 -ha0.97 -hb0.85`
- Small data set, medium heating:

  `-hn40 -ha0.98 -hb0.75`
- Small data set, high heating:

  `-hn80 -ha0.999 -hb0.3`
- Medium data set (e.g. 20 -40 loci), low heating:

  `-hn40 -ha0.96 -hb0.9`
- Medium data set, medium heating:

  `-hn100 -ha0.99 -hb0.75`

- Medium data set, high heating:

  ```
  -hfg  -hn150 -ha0.995 -hb0.4
  ```

- Larger data (100 loci) :

  ```
  -hfg  -hn200 -ha0.997 -hb0.4
  ```

- Larger data (150 loci)

  ```
  -hfg  -hn400 -ha0.999 -hb0.4
  ```

- Larger data (200 loci)

  ```
  -hfg  -hn500 -ha0.999 -hb0.4
  ```

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 10,000 saved topologies or genealogies from at least two well mixed runs. Whether or not 10,000 saved values actually constitute a good sample from the target density depends entirely on how well the simulation was mixing.

If you are sampling genealogies in order to estimate joint posteriors and do likelihood-ratio tests you will need at least 100,000 genealogies. One way to do this is to do multiple 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 use `-p4`), and then after all these runs are done to do a Load-Genealogy mode run that uses all the .ti files and that invokes all the analyses that you want. When you do an Load-Genealogy mode run you can specify how many genealogies to load.

If it is desired to join the results of multiple genealogy sampling runs using a single Load-Genealogy mode run, it is necessary to keep in mind 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.

7.2. How Many Populations can be in the Model?

IMa3 can accommodate anywhere from one to 8 sampled populations (9 with a ghost). Whether or not phylogeny is being estimated, 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 multiple populations for some kinds of history might be quite large. The relationship between the amount of data that is needed, and the quality of estimates under a phylogenetic Isolation-with-Migration model is poorly known, particularly for models with larger numbers of populations. It is possible, for instance, that the amount of data needed to estimate phylogenetic topology is not as great as is needed to obtain good estimates of the Isolation-with-Migration model given the true topology. Secondly, when sampling topologies, it is useful to keep in mind that the program is essentially integrating over all possible Isolation-with-Migration models. The number of parameters over which these integrations occur can become quite large (e.g. with 7 populations there are 13 population size parameters, 72 migration rate parameters, and 6 splitting times). The possible challenges that can arise for different models and data sets are not easily foreseen, but at the very least it is quite possible that some large data sets will present profound mixing problems, even with carefully chosen priors.

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.

7.3. Extending Runs

If there are no restrictions on runtime then the simplest thing to do is use `-B` and `-L` with floating point values (times) in conjunction with `IMburn` and `IMrun` files. Then the user can assess burnin and mixing after results or burntrend results are created.

However if the user is limited to a particular run duration (as is the typically the case on shared servers) then checkpoint (mcf) files should be used. Here are listed the results of various `-r` options in different contexts:

<u>If no mcf files are present:</u>

`-r2` or `-r7` will save an mcf with the given name

If mcf file(s) is present from a previous run:

`-r7` loads an mcf that has the output file name (`-o`) including state space and recorded values, save a new mcf using the output (`-o`) name; adds to preexisting .ti file; adds to preexisting sampled values, no additional burn

`-r67` loads an mcf that has the output file name (`-o`) including state space but not the recorded values, save a new mcf using the output (`-o`) name; overwrites preexisting .ti file; no additional burn (but because the run starts by ignoring all previous samples, it effectively treats previous run(s) as the burnin).

`-r3` will load the the state space, but not recorded values, from the mcf file with name given by `-f`. A run with `-r3` can include an additional burn using `-b`.

`-r36` - same as `-r3`

`-r37` load the mcf, ignore saved values, makes a new .ti file with new name, can do an additional burn with `-b`, makes new mcf file with the output file name (`-o`).

7.4. How to Analyze Large Data Sets

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. Running with multiple chains can help the mixing, and running with multiple cpus can reduce the waiting time. When running on multiple processors, the program requires that the number of chains be a multiple of the number of processors (at least 2 chains per processor). 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. Run on multiple processors, with saving of checkpoint files and writing of a burntrend file at intervals (e.g. with `-b12.0` in the presence of an IMburn file so that a burntrend file is generated every 12 hours). Continue the run, while checking the burntrend file to assess mixing. Once convinced that the burn is sufficient, the IMburn can be deleted or its name changed, and after the next interval the sampling period will begin. Since checkpoint files are being saved, the actual `-L` value does not really matter for these runs, as new sampling runs can be restarted from the mcf files.

3. Much as when assessing the burnin period, sampling runs can be done by writing results at intervals (using `-L` followed by a floating point value in the presence of an IMrun file). Once these chains have run sufficiently long that various indicators suggest a good sample has been taken (e.g. ESS values are high, there are no perceivable trends in the trendplots, etc) they can be stopped. Simply remove or change the name of the IMrun file and the sampling period will end after the current period.

4. If desired a new set of runs for genealogy sampling can be started by reloading the .mcf files. These runs can have very short burnin periods (e.g. `-b0`). Also it is possible to start more runs for sampling genealogies than were used for the burnin. Each set of .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.

   If mcf files are loaded with `-r6` then the previously sampled values will not be loaded and the run begins anew at the current value of the state space that is given in the mcf file. This can be used to treat a previous run as a burnin run.

   If mcf files are read and written with `-r7` then the same command line can be used repeatedly.

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.

7.5. Avoiding Confusion about Migration Parameters and the Direction of Migration

One persistent source of confusion for users of the IM programs 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.

7.6. 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 gene copy 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 small then it may not be having much affect because the total amount of migration is small. Conversely the same rate of migration, per gene copy, will have a larger effect on a population with a larger effective size. 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 $M_{0\rightarrow 1}/u$. Then the population migration rate, at which the genes of population 0 are supplanted by genes from population 1, is $2N_0M_{0\rightarrow 1} = 4N_0u \times (M_{0\rightarrow 1}/u)/2$.

7.7. Cautions

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. 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.

8. EXECUTION AND COMPILATION

8.1. Windows

A Windows executable is available (IMa3.exe) though is not guaranteed to run under all installations. It has been compiled under Microsoft Virtual Studio with MPI_ENABLED and can be run with a single core or multiple cores. To run with a single core simply type '`IMa3`' at the prompt in a cmd or powershell window.

To run this executable under multiple processors on windows, the user will need to have installed the Microsoft version of MPI (called Microsoft MPI, or sometimes MSMPI). After installation the user will need to find the location of mpiexec.exe and put this location in their PATH variable (or include the full path when running mpiexec). Users should be aware of the number of logical cores on their machine and not try to use more than that number, as mpiexec will run (slowly) with a larger number of virtual cores than logical cores. Then. for example. an IMa3 run with 20 chains on 4 cores might look like:

```
mpiexec –n 4  IMa3 –i mydata  -o mydata.out –s123 –b 100000 –L 10000 –q1 –m1
–t1 –hn 20 –ha0.97 –hb 0.5
```

8.2. Linux and related OSs (including MacOS)

IMa3 has been compiled under several different Linux versions, and so long as MPI is available and an MPI-aware C++ compiler is available then it should not be hard to compile. Such compilers on linux systems typically go by names mpiCC, mpicxx, or mpic++; but there may be others out there. Included in the distribution is a `makefile`, and users can try simply running `make`. If it works it will generate an executable called `IMa3`. Alternatively you can just try compiling at the command line in the same directory as all of the program code:

```
mpicxx *.cpp -D MPI_ENABLED -D NDEBUG -O3 -o IMa3
```

Running at the command line in Linux is typically done using `mpirun`, similar to the way `mpiexec` is used the windows example above.

9. COPYRIGHT

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.

10. 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 IMa3 program. The IMfig documation explains how to run the 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 2 was generated using IMfig for a five population model for common chimpanzees.

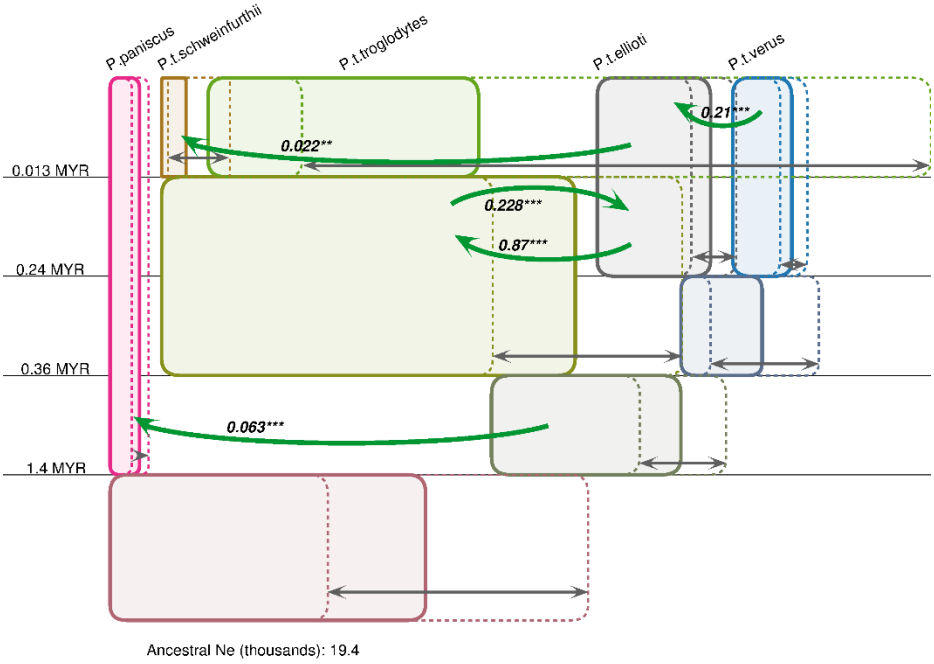

Figure 2

11. 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 CJ. 1991. Markov chain Monte Carlo maximum likelihood. Computing Science and Statistics, Proceedings of the 23rd Symposium on the Interface:156-163.

Hasegawa M, Kishino H, Yano T. (1684 co-authors). 1985. Dating of the human-ape splitting by a molecular clock of mitochondrial DNA. Journal of Molecular Evolution 22:160-174.

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

Hey J, Chung Y, Sethuraman A. 2015. On the occurrence of false positives in tests of migration under an isolation-with-migration model. Molecular Ecology 24 5078-5083.

Hey J, Chung Y, Sethuraman A, Wang Y. 2017. Phylogeny estimation by integration over isolation with migration models. Manuscript.

Hey J, Nielsen R. 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.

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

Hoelzel AR, Hey J, Dahlheim ME, Nicholson C, Burkanov V, Black N. 2007. Evolution of Population Structure in a Highly Social Top Predator, the Killer Whale. Molecular Biology and Evolution 24:1407-1415.

Kimura M. (740 co-authors). 1969. The number of heterozygous nucleotide sites maintained in a finite population due to steady flux of mutations. Genetics 61:893-903.

Knoblauch J, Sethuraman A, Hey J. 2017. IMGui—A Desktop GUI Application for Isolation with Migration Analyses. Molecular Biology and Evolution 34:500-504.

Nielsen R, Wakeley J. (21297244 co-authors). 2001. Distinguishing migration from isolation. A Markov chain Monte Carlo approach. Genetics 158:885-896.

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

Robinson DF, Foulds LR. 1981. Comparison of phylogenetic trees. Mathematical Biosciences 53:131-147.

Sethuraman A, Hey J. 2015. IMa2p – parallel MCMC and inference of ancient demography under the Isolation with migration (IM) model. Molecular Ecology Resources 16:206-215.