BP&P

VERSION 2.1 (April 2011)

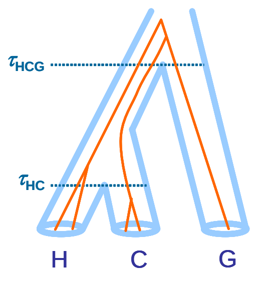


Table of Contents

0. Introduction

BPP is an ANSI C program which implements Bayesian Markov chain Monte Carlo (MCMC) algorithms for analysis of phylogeographic data, that is, sequence alignments from multiple loci and multiple closely-related species. The analysis accommodates the species (population) phylogeny as well as coalescent processes in extant as well as extinct ancestral species. The methods are described in Rannala & Yang (2003) and Yang & Rannala (2010). The models of sequence errors and variable mutation rates among loci are in Burgess & Yang (2008). See also Yang (2006: Chapter 5) for general discussions of Bayesian MCMC algorithms. The parameters in the model include the species divergence times τ, measured by the expected number of mutations per site, and population size parameters $\theta = 4N\mu$, where N is the effective population size and μ is the mutation rate per site per generation so that θ is the average proportion of different sites between two sequences sampled at random from the population. bpp generates samples from the posterior distribution of the τ and θ parameters.

Common model assumptions made in the program include no recombination within a locus, free recombination between loci, no migration (gene flow) between species, and neutral evolution. The JC69 mutation model (Jukes and Cantor, 1969) is assumed to accommodate multiple hits. The sequences are supposed to be close, so that JC69 is deemed adequate.

Example data files used in the method papers are included in the package. We recommend that you use them to duplicate our published results first, before trying to analyze your own data. See the section Example data files for more details.

The same source code can be compiled into a simulation program (MCcoal). See the section *The simulation program (MCcoal)* later in this document for details.

Writing your paper. It has become customary for the programmer to write a paragraph to be copied into the user's paper, so the following is a start. To cite the program, please cite the original papers. You should report the priors in your analysis, while the number of iterations is not essential for others to reproduce your results.
"… analysis was conducted using the program BPP (Rannala and Yang, 2003; Yang and Rannala, 2010). This method accommodates the species phylogeny as well as lineage sorting due to ancestral polymorphism. A gamma prior $G(2, 1000)$ is used on the population size parameters (θs). The age of the root in the species tree (τ_0) is assigned the gamma prior $G(2, 1000)$, while the other divergence time parameters are assigned the Dirichlet prior (Yang and Rannala, 2010: equation 2). Each analysis is run at least twice to confirm consistency between runs."

1. Getting started

Compiling the program

Win32 executables are included. If you use UNIX /linux or Mac osx, remove the .exe files and re-compile the program. You need to do this only once. You can use gcc or any ANSI C-compatible compiler. Some source files are from my PAML package (`paml.h`, `tools.c`, `treesub.c`). Look at the `README.txt` file, or try one of the following:

```
cc -o bpp -fast bpp.c tools.c -lm
cc -o bpp –O4 bpp.c tools.c
gcc -o bpp -O3 bpp.c tools.c -lm
```

```
    cl -O2 -Ot bpp.c tools.c   (MS VC++ 6)
```

The `-o` flag specifies the name of the resulting executable file, `-O3` and `-fast` is for optimizing the code, and `-lm` is to link to the math library. You may have to change some of the optimization flags (`-fast`, `-O4`, `-O3`, etc.), and the `-lm` flag is not needed on some systems.

Trial run

In the bpp/ folder, run the program by typing the following command

```
bpp 5s.bpp.ctl
```

Or move to the `examples/` folder, and run the example analysis

```
   cd examples
   ../bpp yu2001.bpp.ctl
   ../bpp ChenLi2001.bpp.ctl
   ../bpp lizard.bpp.ctl
```

2. File formats

Sequence data file and Imap file

The sequence alignments are in the phylip/paml format, with one alignment following the other, all in one file. Have a look at the sequence files `5stest.txt` and `ChenLi2001.txt`. An alignment is also called a locus. Every locus must have at least 2 sequences. Each sequence name is followed by ^ and an individual ID. The ID is a short string, like MBt03 (for Mallet Basement Topshelf #3). For example in `5stest.txt`, the sequence name `A3^3` means sequence A3 is from individual 3. Sequence names are read and ignored by the program, while the IDs are used.

The IDs are used to map individuals to species through a map file. For example, `5s.Imap.txt` for the `5stest` dataset is shown below, which maps individual 3 to species A.

```
   1 A
   2 A
   3 A
   4 B
   5 C
   6 C
   7 D
   8 E
```

The sequence name and the sequence should be separated by a line break or at least spaces.

By default (`cleandata = 0`), alignment gaps and ambiguity nucleotides are used in the likelihood calculation, with gaps treated as question marks (see Yang, 2006, pp. 107-108). If `cleandata = 1`, all columns with gaps or ambiguity characters are removed before analysis.

Control file

The default control file name is `bpp.ctl`. Lines beginning with an asterisk are comments. The order of the lines is unimportant, I think. Below I will use `ChenLi2001.bpp.ctl` to explain the variables in the control file. This is for analysis of multiple-loci multiple-species data on a fixed species tree. I will then comment on the differences for analysis of data with one species only and for the species-delimitation analysis. Please also read the section on the example datasets later in this document.

Multiple species on a fixed species tree

Below is a copy of the control file `ChenLi2001.bpp.ctl`, to illustrate the analysis on a fixed species tree using the method of Rannala & Yang (2003) and Burgess & Yang (2008). Note that if there are s species in the species tree, the model will involve the following parameters: $(s - 1)$ species divergence times (τs) and $(s - 1)$ ancestral θs. If a species has at least two sequences at any loci, a θ for that species will be used as well. If there is one species, the model will involve one parameter only, θ for that species. The parameters are ordered as follows: θs for the extant species, θs for the ancestral species ($s - 1$ of them), and the divergence times τs for the ancestral nodes.

```
          seed =  -1

       seqfile = ChenLi2001.txt
      Imapfile = ChenLi2001.Imap.txt
       outfile = out.txt
      mcmcfile = mcmc.txt

speciesdelimitation = 0 * fixed species tree
  species&tree = 4  H  C  G  O
                    1  1  1  1
                 (((H, C), G), O);

       usedata = 1    * 0: no data (prior); 1:seq like
         nloci = 53    * number of datasets in seqfile

     cleandata = 0    * remove sites with ambiguity data (1:yes, 0:no)?

    thetaprior =  2 2000   # gamma(a, b) for theta
      tauprior = 14 1000   # gamma(a, b) for root tau & Dirichlet(a) for other tau's

*      heredity = 0    # (0: No variation, 1: estimate, 2: from file) & a_gamma b_gamma (if 1)
*      heredity = 1 4 4   # (0: No variation, 1: estimate, 2: from file) & a_gamma b_gamma (if 1)
*      heredity = 2 heredity.txt   # (0: No variation, 1: estimate, 2: from file) & a_gamma b_gamma (if 1)

*     locusrate = 0 2.0    # (0: No variation, 1: estimate, 2: from file) & a_Dirichlet (if 1)
* sequenceerror = #  model of sequencing errors has changed, to be described later

       finetune =  0: 0.5 0.0015 0.0006  0.0004 0.06 0.2 1.0  # auto adjustment: finetune steplengths for
GBtj, GBspr, theta, tau, mix, locusrate, seqerr

         print = 1
        burnin = 2000
      sampfreq = 2
       nsample = 20000
```

`seed` is the random number seed. If you use a positive integer, the program will produce identical results in different runs. This is useful for debugging. If you use -1, the program will use the wall clock to generate a seed, and different runs will produce different results. It is recommended that you run the same analysis at least twice using different seeds to confirm that the results are stable across runs.

`seqfile` is the name of the sequence alignment file, while
`Imapfile` is the individual map file. The Imapfile is not needed if the data contain only one species. These two files are input.

`speciesdelimitation = 0` assumes that the species tree given below is fixed.
`speciesdelimitation = 1` is for species delimitation, with the species tree given below treated as

the guide tree. See the section below.

```
species&tree = 4  H  C  G  O
                  1  1  1  1
             (((H, C), G), O);
```

The above block specifies 4 species in the data, which are H (human), C (chimp), G (gorilla), and O (orang). The maximum number of sequences is 1 for H, 1 for C, 1 for G, and 1 for O. These numbers serve two purposes. First, they are used to determine which θ parameters are involved in the model and should be estimated. Second, they specify the maximum number of sequences for all species at a locus. The model always use a θ and a τ (age) for every interior (ancestral) node on the species tree. The model also uses a θ for each extant species if and only if that species has more than one sequence at some locus. If the example here, the parameters are the three ancestral θs and three node ages (τs).

The species tree is fixed if `speciesdelimitation = 0` and is used as the guide tree in the rjMCMC run for species delimitation if `speciesdelimitation` = 1.

`usedata = 0` is for running the MCMC without sequence data to generate the prior, while `usedata = 1` is for generating the posterior.

`nloci` specifies the number of loci (alignments).

`cleandata` = 1 causes the program to remove all columns in the alignment which have gaps or ambiguity characters. `cleandata` = 0 means that those will be used in the likelihood calculation.

`thetaprior = 2 2000` specifies the gamma prior $G(\alpha, \beta)$ for the θ parameters, with the mean to be α/β. In the example, the mean is 2/2000 = 0.001.

`tauprior` specifies the gamma prior $G(\alpha, \beta)$ for τ_0, the divergence time parameter for the root in the species tree. Other divergence times are generated from the Dirichlet distribution (Yang and Rannala, 2010: equation 2).

```
heredity = 0       # (0: No variation)
heredity = 1 4 4   # (1: estimate, & a_gamma b_gamma)
heredity = 2 heredity.txt      # (2: from file)
```

`heredity = 0` is the default and means that θ is the same for all loci. `heredity = 1` or `2` specifies two models that allow θ to vary among loci, which may be useful for combined analysis of data from autosomal, mitochondrial, X and Y loci. With such mixed data, the effective population sizes are different among loci, so that a heredity multiplier (inheritance scalars, Hey and Nielsen, 2004) should be applied. Other factors such as natural selection may also cause θ to deviate from the neutral expectation. bpp implements two options for this, but one option is not working and the other is not well tested. The first option (`heredity = 1`) is to estimate the multipliers from the data, using a gamma prior with parameters α and β specified by the user. In the example above, a gamma prior $G(4, 4)$, with mean 4/4 = 1, is specified for the multiplier for each locus. The MCMC should then generate a posterior for the multiplier for each locus. The print out however has included only the estimates for the first ten loci. The second option (`heredity = 2`) is for the user to specify the multipliers in a file, and the multipliers will then be used as fixed constants in the MCMC run. I have not tested this option carefully.

Genome	Heredity scalar
Nuclear autosome	1
X chromosome	0.75
Y chromosome	0.25
Mitochondrial	0.25

```
locusrate = 0                        # (0: No variation)
locusrate = 1 2.0                    # (1: estimate, & a_Dirichlet)
locusrate = 2 LocusRateFileName      # (2: from file)
```

`locusrate = 0` (default) means that all loci have the same mutation rate. `locusrate = 1` or `2` specifies two models for variable mutation rates among loci (Burgess and Yang, 2008).
`locusrate = 1` specifies the random-rates model of Burgess & Yang (2008: equation 4). The average rate for all loci is fixed at 1, while the rates among loci are assumed to generated from the Dirichlet distribution $D(\alpha)$. Parameter α is inversely related to rate variation, with a large α meaning similar rates among loci. If all loci are nocoding, the rates are probably similar, so $\alpha = 10$ or 20 may be reasonable, while $\alpha = 2$ or 1 may be too small. The MCMC generates the posterior for rates at loci, but I think only the rate for the first locus is included in the output.
`locusrate = 2 LocusRateFileName` specifies the fixed-rates model of Burgess & Yang (2008). This is the strategy used by Yang (2002), with the relative rates estimated by the distance to an outgroup species. The relative locus rates are listed in the file: there should be as many numbers in the file, separately by spaces or line returns, as the number of loci or `ndata`. The program re-scales those rates so that the average among all loci is 1 and then use those relative rates as fixed constants.

Note 1. The effect of the locus-specific mutation rates and the locus-specific heredity multipliers are different. A locus rate is used to multiply all θs and τs for the locus, while a heredity multiplier is used to multiply all θ parameters for the locus but not the τs. Nevertheless, those parameters are quite likely to be strongly correlated, especially when the species tree is small. Thus you should not be over-excited by those models. If you would like to use them, I suggest that you analyze the loci as separate datasets as well to avoid being totally misled by the complicated models.

Note 2. The variable-rates model implemented here has some differences from a similar model implemented in the IMa program (Hey and Nielsen, 2004). The biggest difference appears to be the parametrization. bpp defines mutation rate on a per-nucleotide basis, so the prior specifies that the expectation of the mutation rate per site is constant among loci. IM defines mutation rate on a per-locus basis, so its prior specifies that the expectation of the mutation rate per locus is the same among loci. If locus one has 100 sites and locus two has 1000 sites, then IM assumes that the per-site rate for locus one is 10 times for locus two, while bpp assumes the same per-site rate. Also IM constrains the geometric mean of rates across loci to be one, while bpp constrains their arithmetic mean to be one. Personally I do not think the IM assumptions are reasonable.

Ziheng note on 3 March 2011.
The model of sequence errors is rewritten, so that the old options, described in this box, is obsolete. The new model will be described later, and this section of the manual updated.

```
sequenceerror = 0                     # (0: default: No error)
sequenceerror = 0 1 0 0 : 0.05 1      # (1: estimate, & a_gamma b_gamma)
sequenceerror = 0 0.002 0 0           # (fixed error rate at 0.2% in chimp)
sequenceerror = 0 0.002 1 0 : 0.05 1  # (fixed error 0.2% in chimp & error in Gorilla)
```

`sequenceerror = 0` means no sequencing errors.

`sequenceerror = 0 1 0 0 : 0.05 1` is used to implement the model of random sequencing errors of Burgess & Yang (2008). The four flags above, `0 1 0 0`, correspond to the four species H, C, G, O, where 0 means no error and 1 means possible errors. In this example the chimpanzee sequences may have errors while the other species do not. The error rate ε is then assigned a gamma prior $G(0.05, 1)$, with mean 0.05. You cannot assume errors in every species so `sequenceerror = 1 1 1 1` is not allowed. The same gamma prior on ε is assigned for all species assumed to have errors. The posteriors of the error rates differ between species.

`sequenceerror = 0 0.002 0 0` means that the error rate in chimp is fixed at 0.2%, while the other species do not have errors.

`sequenceerror = 0 0.002 1 0 : 0.05 1` means that the chimp has a fixed error rate of 0.2% while the gorilla has an error rate $\varepsilon_G \sim G(0.05, 1)$. The program will generate a posterior for ε_G.

The model of sequence errors is motivated by analysis of ape genomic data, in which certain species are understood to have substantial sequencing errors, due to low coverage compared with the human genomic sequence (Burgess and Yang, 2008). Errors are assumed to occur at random and to affect every nucleotide site, at the rate of ε. The model is implemented by extending every terminal branch leading to that species in every gene tree by length ε. If errors tend to cluster (e.g., if different chromosomes are sequenced at different coverages for the same genome), the model will be unrealistic. Also the model relies on the molecular clock to detect and estimate sequencing errors, and is unable to distinguish sequence errors from violation of the clock. Indeed Burgess & Yang (2008) used this model to mimic hominoid slowdown, which is a violation of the clock.

```
finetune = 0: 0.5 0.0015 0.0006 0.0004 0.06 0.2 1.0  # auto (0 or 1): finetune for GBtj,
GBspr, theta, tau, mix, locusrate, seqerr

finetune = 1: 0.01 0.01 0.01 0.01 0.01 0.01 0.01  # auto (0 or 1): finetune for GBtj, GBspr,
theta, tau, mix, locusrate, seqerr
```

This is about the step lengths used in the proposals in the MCMC algorithm. The first value, before the colon, is a switch, with 0 meaning no automatic adjustments by the program and 1 meaning automatic adjustments by the program. Following the colon are the step lengths for the proposals used in the program. If you choose to let the program adjust the step lengths, burnin has to be >200, and then the step lengths specified here will be the initial step lengths, and the program will try to adjust them using the information collected during the burnin step. This option is not yet well tested.

Below are notes about adjusting the step lengths manually. The first five of the fine-tune variables are ε_1, ε_2, ε_3, ε_4, ε_5, described in Rannala & Yang (2003), and are for proposals that (1) change internal node ages in the gene tree, (2) prune and re-graft nodes in the gene tree, (3) update θs, (4) update τs using the rubber-band algorithm, and (5) implements the mixing step. The next two steps are for modifying the locus rate under the model of random mutation rate variation among loci, and for modifying the error rate parameters in the model of sequencing errors.

You have to run the MCMC for a small number of generations and look at the screen output for the acceptance proportions (see the Section on *Running the program – Adjusting the finetune parameters*) for details.

One complication is that if certain parameters do not exist in the model, the acceptance proportion for the proposal that change them will be printed out as 0. For example, if there is only one species, the model will not involve any parameter τ, and the proportion for updating τ will be 0; in this case the step length ε_4 is read and ignored by the program.

```
print = 1
burnin = 2000
sampfreq = 2
nsample = 20000
```

`print = 0` means that no MCMC samples are written into the file, which may be useful if you need the screen output only. `print = 1` generates a file `mcmc.txt` containing the MCMC samples. Sampling starts after the burnin, and in this example, are taken every 2 iterations, with a total of 20,000 samples taken. The total number of MCMC iterations is burnin + output × nsample. The resulting file can be large. If the analysis is conducted on a fixed species tree (`speciesdelimitation = 0`), this file should be readable from Andrew Rambaut's TRACER, I think.

One species

Look at `examples/yu2001.bpp.ctl`, which is for analyzing a sample of 61 human sequences from Yu et al. (2001) to estimate the single parameter $\theta = 4N\mu$ under the standard coalescent model.

Most variables are used in the same way as above. As there is only one species, there is no need for the Imap file. There is neither need to tag the sequence names in the sequence file (`yu2001.txt`): the sequence names are read and then ignored. Multiple loci can be included in the sequence file, but the example includes only one locus. There is no species tree, so the block for specifying species names and species tree looks like this:

```
species&tree = 1  H
                 100  * max number of sequences
```

The printout on the monitor includes the posterior mean of θ, and posterior means of μt_{MRCA} for the loci, calculated up to that point in the MCMC run. If you have many loci, only the first few μt_{MRCA} are printed on the monitor.

The mcmc sample file lists 2 + g columns for g loci: θ, μt_{MRCA} for the g loci, and the log likelihood. The μt_{MRCA} are not parameters, but are sometimes of interest as well.

Species delimitation

Look at the example control file `examples/lizard.bpp.ctl`. Use one of the following two lines to specify a rjMCMC algorithm for species delimitation(Yang and Rannala, 2010).

```
speciesdelimitation = 1 0 5    * speciesdelimitation algorithm0 and finetune(e)
speciesdelimitation = 1 1 2 1  * speciesdelimitation algorithm1 finetune (a m)
```

The first line specifies algorithm 0, with $\varepsilon = 5$ in equations 3 and 4 of the paper, while the second line specifies algorithm 1, with $\alpha = 2$ and $m = 1$ in equation 6 of the paper.

You should run the program a few times, using both algorithms 0 and 1, with different starting species trees and different fine-tune parameters to confirm the stability between runs. With algorithm 0, you may try 2, 5, 10 or 20 for ε. With algorithm 1, you may try $\alpha = 1, 1.5$, or 2, and $m = 0.5, 1, 2$ etc.

```
uniformrootedtrees = 1         * 1 means uniform rooted trees (default)
uniformrootedtrees = 0         * 0 means uniform labeled histories
```

This variable specifies the prior on the species tree models/species delimitations (Yang and Rannala, 2010: page 9265 bottom).

3. Running the program

Screen output

Make the window wider, with 100 or 120 columns before you run the program. (On Windows, you right click the window title bar and choose Properties – Layout and change Window Size Width.) Pay attention to screen outputs, especially at the start of the run, to make sure that the control file and sequence data are read correctly by the program, and that the acceptance proportions are reasonable. Use Ctrl-C to terminate the run after 1 or 0.5 minutes, change the finetune step lengths and rerun the program.

Here is an outline of the steps taken by the program. The sample output is from analyzing the example dataset `ChenLi2001.txt`, on a fixed species tree. The differences for the species delimitation analysis are discussed later. The program first prints out the species tree, as well as a population-population table, which describes the descendant-ancestor relationship between populations and which you may ignore. It then defines the θ and τ parameters involved in the model.

The program then reads and processes the sequence data file.

It then generates the initial values for parameters θs and τs by using the gamma priors and the initial gene trees and coalescent times by sampling from the prior. The program prints out the initial θs and τs, as well as the initial log likelihood lnL0, and starts the MCMC.

```
Starting MCMC...
prior theta ~ G(2.000, 2000.000)
prior tau   ~ G(14.000, 1000.000)

Initial parameters, np = 6 (gene trees generated from the prior):
  0.00091  0.00111  0.00100  0.01374  0.00306  0.00036
lnL0 =  -43347.191
  0%  0.28 0.42 0.37 0.38 0.38  0.00192 0.00344 0.00240  0.01370 0.00593 0.00477 -42852.55  0:05
  5%  0.28 0.39 0.35 0.37 0.36  0.00198 0.00342 0.00182  0.01403 0.00602 0.00483 -42845.57  0:22
 10%  0.28 0.38 0.34 0.37 0.36  0.00186 0.00349 0.00174  0.01408 0.00596 0.00486 -42837.31  0:35
```

Then on the same line, it prints out a percentage progress indicator (with negative values indicating burnin), followed by the acceptance proportions for the MCMC moves (highlighted in red in the sample output), and by the posterior means of the parameters (in this example there are three θ parameters and three τ parameters). The last number before the time used is the current log likelihood. You have to adjust the finetune variables in the control file so that the acceptance proportions lie in the interval (0.15, 0.7). See below.

The program will then read and process the file `mcmc.txt` to calculate the mean, min, max, median, and percentiles, and histogram information. This may take quite some time if many samples (say, 1M) are collected in the file. The included small program ds can also read the sample file to generate those summary statistics. (use `ds mcmc.txt`).

Adjusting finetune variables for MCMC moves (fixed species tree)

The finetune parameters have to be adjusted to achieve reasonable acceptance proportions for all proposals used in the MCMC. Otherwise the results may be meaningless.

The following notes are about manual adjustment of step lengths (finetune parameters) for MCMC moves in the program. You can use the automatic adjustment as well, but please check the acceptance proportions achieved. See the description of the control file.

There is a line like the following in the control file `ChenLi2001.bpp.ctl`:

```
finetune =  0: 0.5 0.005 0.0006  0.0004 0.06 0.2 1.0  # finetune for GBtj, GBspr, theta, tau, mix, locusrate, seqerr
```

These are the step lengths used in the MCMC proposals. The program does not adjust them automatically and you have to adjust them yourself, by looking at the screen output, like the following.

```
    lnL0 =  -42908.975
      0%  0.28 0.13 0.34 0.37 0.37  0.00189 0.00356 0.00176  0.01408 0.00589 0.00478 -42857.29  0:06
      5%  0.28 0.15 0.39 0.36 0.38  0.00184 0.00347 0.00282  0.01349 0.00594 0.00483 -42828.80  0:18
```

There are seven finetune parameters in the control file. They are in a fixed order and always read by the program even if the concerned proposal is not used. If the model assumes the same rate for all loci and does not use heredity multipliers, the 6th proposal step is not used. If the model assumes no sequencing errors, the 7th step is not used. The acceptance proportions for the first five proposals are always printed out on the screen, but those for the 6th and 7th are printed out only if the concerned proposal is used in the model. In the example above, only the first five proposals are used in the algorithm and only their acceptance proportions are printed out on the monitor.

The optimal acceptance proportions are around 0.3, and you should try to make them fall in the interval (0.15, 0.7). If the acceptance proportion is too small (say, <0.10)), decrease the corresponding finetune parameter. If the acceptance proportion is too large (say, >0.80), increase the finetune parameter. In the above example, the second acceptance proportion, at 0.13 or 0.15, is somewhat too small, which means that the corresponding finetune parameter (0.005 above) is too large. Terminate the program (Ctrl-C) and decrease the value in the control file. Then run the program again (use the up ↑ and down ↓ arrow keys to retrieve past commands). Repeat this process a few times until every acceptance proportion is neither too small nor too large. In this example, changing 0.005 to 0.002 brings the acceptance proportion to 34%, which is good.

Note the different effects of the finetune parameters and of the priors. The priors affect the results of an analysis while the finetune parameters affect the efficiency of the MCMC or how fast one can obtain those results. If the finetune parameters are poorly chosen, it will take a very long time for the chain to converge and mix.

The rjMCMC algorithm for species delimitation

Our MCMC algorithms on a fixed species tree appear to be stable and efficient. They have been used to analyze genomic datasets with >50,000 loci (Burgess and Yang, 2008). In contrast, the rjMCMC algorithms for species delimitation are found to be quite problematic. The rjMCMC moves between models of different dimensions, and this is clearly the source of the difficulty. For example, the one-species model has one single parameter (θ for that species), but a two-species model may have 2-4 parameters (1-3 θ parameters plus one τ).

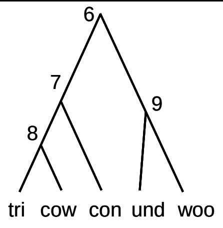
As θ for the root appears in all species-tree models, its posterior mean is printed out on the monitor, but other parameters are not printed. The rjMCMC algorithm applied to the `lizard` data generates screen outputs be like the following. Check the output to make sure that the data files are read correctly.

```
Starting MCMC...

Starting species tree = 1101
node  1 theta = -1.000000  tau =  0.000000
node  2 theta = -1.000000  tau =  0.000000
node  3 theta =  0.001790  tau =  0.000000
node  4 theta =  0.001867  tau =  0.000000
node  5 theta = -1.000000  tau =  0.000000
node  6 theta =  0.002170  tau =  0.001945
node  7 theta =  0.001940  tau =  0.000726
node  8 theta =  0.001957  tau =  0.000000
node  9 theta =  0.001775  tau =  0.000345

prior theta ~ G(2.000, 1000.000)
prior tau   ~ G(2.000, 1000.000)

Initial parameters, np = 9 (gene trees generated from the prior):
 0.00179  0.00187  0.00196  0.00194  0.00177  0.00217  0.00195  0.00073  0.00035
lnL0 =    -1203.653
 -5%  0.55 0.22 0.46 0.33 0.19   6 0.0758 1001  0.00340 -1033.39  0:21
  0%  0.55 0.22 0.46 0.33 0.19   6 0.0806 1001  0.00330 -1032.25  0:43
  4%  0.55 0.21 0.46 0.33 0.19   9 0.0822 1101  0.00336 -1034.77
```



The species tree is represented using four 0-1flags for the four interior nodes 6, 7, 8, 9 in the guide tree, with 0 for 'collapsed' and 1 for 'resolved'. The starting species tree is generated by choosing an interior node for collapsing. In this run, the starting tree is 1101, which means that node 8 in the guide tree is collapsed so that there are 4 species and 9 parameters in the model.

Note that the tips in the guide tree are numbered 1, 2, …, s for s potential species, while the interior (ancestral) nodes are numbered s + 1, s + 2, …, 2s – 1, with s + 1 to be the root of the guide tree. The numbering is through a tree-traversal algorithm, fixed by the program. This same order is used to specify the divergence time parameters (τs), so you can work out the order by looking at the list of nodes in the screen output (look at the "population by population table", "# species divergence times in the order:", etc.).

After the chain has started, the five ratios after the % sign are the acceptance proportions for the conventional MCMC moves discussed above. Adjust the values in the `finetune` line in `lizard.bpp.ctl` so that those proportions are close to 0.3-0.4, as discussed above.

There is however one complication. The one-species model (0000 for the lizard example) does not have any τ parameter, so the acceptance proportion for the proposal which changes τ is shown as 0. When the chain moves between the one-species model and other models with two or more species, the acceptance proportion for updating τ is calculated by simple average and may look too low even if the chain is moving adequately. This is because I have not bothered to calculate the acceptance proportion correctly, and no proposals are treated like rejected proposals. The acceptance proportions for θs are correct.

After the acceptance proportions for the MCMC moves, there are three numbers related to the rjMCMC move, highlighted in red above. The rest of the line shows the posterior mean for θ for the root (which is a parameter shared by all species-tree models) and the current log likelihood lnL.

The three numbers related to the rjMCMC move, "`6 0.0758 1001`" in the example, mean 6 parameters in the current species tree, which is 1101, and the rjMCMC move has the acceptance proportion 0.0758. Unfortunately there is no optimal acceptance proportion for the rjMCMC move, and a value close to 0 may not necessarily mean a poor algorithm. If one model has posterior probability close to 1, the acceptance proportion should be near 0 as well. Thus both poor mixing of the rjMCMC algorithm and extreme posterior model probabilities can cause the acceptance proportion for the rjMCMC to be close to 0. It has been noted that if the rjMCMC algorithm is suffering from poor mixing, use of different starting species trees will generate different results.

After the MCMC is finished, the program will print out the posterior probabilities for the species-tree models.

The MCMC sample file `mcmc.txt`. As the number of parameters changes when the rjMCMC moves between models, the mcmc sample file may not be very useful, so you can ignore it. Right now the header line is generated using the starting species tree and should be ignored. After the header line, each line of output has the following numbers, separated by TABs: iteration number, the number of parameters, the tree, the sampled parameter values, and lnL.

```
20 6 1001 0.000538 0.006272 0.013239 0.005056 0.001706 0.000075 -1027.982
```

For example, this above sample is taken when the chain in species model 1001, with 6 parameters. If you know the unix command grep, you can retrieve the lines for the same tree model to summarize the posterior for parameters in that model.

```
grep "6 1001" mcmc.txt > result.Tree1001.txt
```

This should give you the same results as if you run the analysis with the species tree fixed at tree 1001. Of course you can edit the necessary sequence file, and Imap file to run the analysis using the fixed species tree.

Below are some further notes about running the rjMCMC algorithms.

- We have come across one case in which the chain is stuck at the one-tree model (0000), while the correct posterior probability should be ~1.0 for the fully resolved tree (1111). There are more than 10 nuclear loci in the data. There seems to be no problem if a different starting species tree is used, such as 0001, when the main moves quickly to 1111. The problem does not seem to occur in small datasets with one or two loci. If you have a similar problem, you may start the analysis with 1 locus, then 2 loci, etc. to observe the change in the results (using `nloci` in the control file as there is no need to change the sequence file).
- Observe the acceptance proportion for the reversible jump (rj) proposal. If this is >0 and the results are stable when you change the rj algorithm and/or step lengths, the results are probably fine. However when the acceptance proportion is 0, it can mean a mixing

problem (since the chain is visiting one tree only), but it can also mean that the correct posterior probability for that tree is nearly 1, so that the chain should correctly stay in that tree.

- Use sensible priors for tau and on theta. If you don't have much info you can use a = 1.5 or 2. However you need to use sensible prior means. See the note about the gamma distribution later in this document.
- If you encounter similar mixing problems, you may try to run the program with the species tree fixed, and look at the posterior estimates of parameters.

4. Example data files

Three datasets are included in the package to duplicate previous results concerning (a) estimation of θ for one single species, (b) estimation of θs and τs on a fixed species tree, and (c) species delimitation using a guide tree. In addition, a few example datasets are included in the likelihood program `3s`, which were analyzed using bpp to generate results reported by Burgess & Yang (2008) and Yang (2010).

Example dataset 1: One species

Use the Yu et al. (2001) dataset (`yu2001.bpp.ctl` and `yu2001.txt`) to estimate θ for a locus from human individuals. The results are in table 3 of Rannala & Yang (2003), with $\hat{\theta}_H$ = 0.00035, the μt_{MRCA} = 0.00031.

```
        cd examples
        ../bpp yu2001.bpp.ctl
```

Example dataset 2: Fixed species tree

The hominoid data of Chen & Li (2001) includes one sequence from each of human, chimpanzee, gorilla, and orangutan, at 53 loci. The data are used to estimate three θ parameters (θ_{HC}, θ_{HCG}, θ_{HCO}) and three species divergence time parameters (τ_{HC}, τ_{HCG}, τ_{HCGO}) on the species tree (((H, C), G), O).

```
        cd examples
        ../bpp ChenLi2001.bpp.ctl
```

The results should be compared with those in the column "Posterior (53 loci)" in table 2 of Rannala & Yang (2003). However, the speciation times are now 'parametrized' differently. Rannala & Yang used $\tau_{\text{HCGO}} - \tau_{\text{HCG}}$, $\tau_{\text{HCG}} - \tau_{\text{HC}}$, and τ_{HC} while bpp now uses τ_{HCGO}, τ_{HCG}, and τ_{HC} in both prior specification and output. Furthermore, the specification of priors has changed as well. Rannala & Yang used a separate gamma prior for each θ parameter, while bpp now uses the same gamma prior for all θs. Also bpp assigns a gamma prior for the root age (τ for the root node) and a Dirichlet distribution prior to generate the other node ages (τs). For comparison, the results are listed below in table 1.

TABLE 1 Prior and posterior distributions of parameters in the Bayesian analysis of the 53 loci of Chen and Li (2001)

	Gamma prior	Prior	Posterior (53 loci)
Parameter	(α, β)	Mean (95% interval)	Mean (95% interval)

θ_{HC}	(2, 2000)	0.001 (0.00012, 0.00279)	0.00187 (0.00051, 0.00392)
θ_{HCG}		0.001 (0.00012, 0.00279)	0.00349 (0.00216, 0.00500)
θ_{HCGO}		0.001 (0.00012, 0.00279)	0.00199 (0.00026, 0.00450)
τ_{HCGO}	(14, 1000)	0.01415 (0.00758, 0.02227)	0.01395 (0.01230, 0.01541)
τ_{HCG}		0.00956 (0.00246, 0.01852)	0.00593 (0.00515, 0.00681)
τ_{HC}		0.00471 (0.00018, 0.01340)	0.00483 (0.00395, 0.00562)

NOTE .— Both τ and θ are measured as the expected number of mutations per site.

Example dataset 3: species delimitation

The dataset of North American fence lizards *Sceloporus* (Leache, 2009) is included as files (`lizard.txt`, `lizard.Imap.txt`, `lizard.bpp.ctl`). This is one of the datasets analyzed by Yang & Rannala (2010). To duplicate the results, type

```
cd examples
../bpp lizard.bpp.ctl
```

The results for one locus are shown below in table 2. In the paper we used cleandata = 1. You can change the rjMCMC algorithm as well as the fine-tune parameters to see how the algorithm behaves.

TABLE 2 Posterior probabilities for different tree models for the lizard data (1 locus)

Tree	cleandata = 1	cleandata = 0
0000	0.006	0.006
1000	0.020	0.086
1001	0.088	0.413
1100	0.039	0.026
1101	0.168	0.100
1110	0.129	0.084
1111	0.549	0.284

5. A note about the gamma prior

The gamma distribution is used in bpp to specify priors for parameters, partly because all parameters involved in the model are strictly positive. The gamma $G(\alpha, \beta)$ has mean $m = \alpha/\beta$ and variance $s^2 = \alpha/\beta^2$. (Note that in the literature another commonly used parametrization of the gamma gives the mean as $\alpha\beta$.) It may be easier to think of mean m and standard deviation s when constructing the prior, and then get α and β as $\alpha = (m/s)^2$ and $\beta = m/s^2$. For example, as the divergence time between humans and chimps is ~5MY, and the mutation rate is ~10^{-9} per site per year, the mean of τ_{HC} should be ~0.005. To use a fairly informative prior, we may set $\alpha = 25$ and then $\beta = \alpha/m = 5000$.

If little information is available about the parameter, one may want to use a diffuse prior. One can fix m first, and then adjust s or α, with a large s or small α meaning a less informative prior. In a normal distribution, the 95% interval is given roughly as $m \pm 2s$. This rule does not apply well to the gamma distribution if α is small (say, $\alpha < 1$), but may still be used as a guide. With this reasoning, $s = m$ or $\alpha = 1$ represent a very diffuse prior while $s = m/2$ or $\alpha = 4$ is still quite diffuse. Thus values like 1.5 or 2 for α are fairly diffuse, and you can then choose β to get the mean roughly right. Try to avoid very small α (such as 0.1 or 0.01), as they may cause numerical problems.

6. History

This part is moved into the file bppHistory.txt file.

7. The simulation program (MCcoal)

A simulation program MCcoal is included to simulate sequence alignments on the fixed species tree, that is, under the model of Rannala & Yang (2003). In addition, the simulation model allows migration between species/populations, even though the inference program bpp does not allow migration. Look at the README.txt file for compiling the program. To run the program, type one of the following. The default control file name is `MCcoal.ctl`. Always look at the screen output to confirm that the program reads the control file correctly.

```
MCcoal
MCcoal MCcoalMigration.ctl
```

Simulating without migration

```
          seed = 12345

       seqfile = MySeq.txt   * comment out this line if you don't want seqs
      treefile = MyTree.tre  * comment out this line if you don't want trees
      Imapfile = MyImap.txt

  species&tree = 4  A  B  C  D
                    3  2  1  1

  ((A #0.1, B #0.2) : 0.5 #.12, (C, D) :0.8 #0.34) : 1.0 #.1234;

   loci&length = 100 1000 * number of loci & number of sites at each locus
```

The example above (`MCcoal.ctl`) is for simulating 100 loci, each of 1000 sites, on the species tree ((A, B), (C, D)), with 3, 2, 1, 1 sequences for A, B, C, D, so that there are 7 sequences at each locus. The divergence time parameters (τs) are after ':' in the tree, while the population size parameters (θs) are after '#'. Thus we have $\theta_A = 0.1$, $\theta_B = 0.2$, $\theta_{AB} = 0.12$, $\theta_{CD} = 0.34$, $\theta_{ABCD} = 0.1234$, $\tau_{ABCD} = 1.0$, $\tau_{AB} = 0.5$, and $\tau_{CD} = 0.8$. We need θ_A and θ_B because 2 or more sequences are sampled from species A and B. Parameters θ_C and θ_D are unnecessary: if you specify them, they will be ignored by the program. (Those parameter values are too big. Replace them with more realistic values.)

Simulating with migration

```
      migration = 7   * number of pops (order fixed by program)

               A      B      C      D     ABCD    AB    CD
       A       0      1.1    1.2    1.3    0      0     -1
       B       0.1    0      1.4    1.5    0      0     -1
       C       0.2    0.4    0      1.6    0      1.7    0
       D       0.3    0.5    0.6    0      0      1.8    0
       ABCD    0      0      0      0      0      0      0
       AB      0      0      0.7    0.8    0      0      1.9
       CD     -1     -1      0      0      0      0.9    0
```

The control file for simulating migration as well as coalescence includes a block as above (see the file `MCcoalMigration.ctl`). The line `migration = 7`, where 7 is the number of populations on the species tree, tells the program to simulate migrations. This number is fixed by the species tree and is used here for error checking. This is then followed by a migration matrix, of size 7 × 7. The names of the populations are read and ignored by the program, and the order of the populations is fixed. You should run the program without the migration matrix first and then use the screen output to use the correct order of populations to specify the migration matrix.

In the migration matrix, values 0 and −1 mean that migration is either impossible (for example, if the two populations did not live at the same time) or not allowed (if the two populations were contemporary but no migration between them is assumed to occur). Use 0 for both cases here. Positive values are scaled migration rates, with the element on the *i*th row *j*th column to be $M_{ij} = N_j m_{ij}$, where m_{ij} is the migration rate per generation in population *j* from population *i*, or the proportion of individuals in population *j* that are immigrants from population *i*, and where μ is the mutation rate per site per generation. In the example above, $M_{\mathrm{A}\to\mathrm{B}} = 1.1$, which means that on average 1.1 gene sequences are immigrants from population A to population B.

Note that the θ parameter for a modern species has to be specified in the tree file even if only one sequence is sampled from that species but migrations into that species are allowed by the migration matrix.

8. References

Burgess, R., and Z. Yang. 2008. Estimation of hominoid ancestral population sizes under Bayesian coalescent models incorporating mutation rate variation and sequencing errors. Mol. Biol. Evol. 25:1979-1994.

Chen, F.-C., and W.-H. Li. 2001. Genomic divergences between humans and other Hominoids and the effective population size of the common ancestor of humans and chimpanzees. Am. J. Hum. Genet. 68:444-456.

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

Jukes, T. H., and C. R. Cantor. 1969. Evolution of protein molecules. Pages 21-123 *in* Mammalian Protein Metabolism (H. N. Munro, ed.) Academic Press, New York.

Leache, A. D. 2009. Species tree discordance traces to phylogeographic clade boundaries in North American fence lezards (sceloporus). Syst. Biol. 58:547-559.

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

Yang, Z. 2002. Likelihood and Bayes estimation of ancestral population sizes in Hominoids using data from multiple loci. Genetics 162:1811-1823.

Yang, Z. 2006. *Computational Molecular Evolution*. Oxford University Press, Oxford, England.

Yang, Z. 2010. A likelihood ratio test of speciation with gene flow using genomic sequence data. Genom. Biol. Evol. 2:200-211.

Yang, Z., and B. Rannala. 2010. Bayesian species delimitation using multilocus sequence data. Proc. Natl. Acad. Sci. U.S.A. 107:9264-9269.

Yu, N., Z. Zhao, Y. X. Fu, N. Sambuughin, M. Ramsay, T. Jenkins, E. Leskinen, L. Patthy, L. B. Jorde, T. Kuromori, and W. H. Li. 2001. Global patterns of human DNA sequence variation in a 10-kb region on chromosome 1. Mol. Biol. Evol. 18:214-222.