Manual

DivBayes 1.1

Martin Ryberg[*] and R. Henrik Nilsson
[*]kryberg@utk.edu

Introduction

DivBayes is a program to estimate net diversification rate (netdiv; speciation minus extinction rate) and relative extinction rate (extinction proportion; extprop; extinction rate divided by speciation rate) from stem ages and number of species of a set of clades. It uses Markov chain Monte Carlo (MCMC) methods in a Bayesian context to produce an estimate of the posterior probability distribution for the parameters. DivBays reads a texfile with the data and specific commands and then executes the MCMC. The output is a tab-separated table of the Markov Chain that can be summarized with a consecutive command. DivBayes is available from `http://web.utk.edu/~kryberg/`

Bayesian estimation of posterior distribution

Bayes' theorem states that the posterior probability distribution ($f(\theta|X)$) for the parameters (θ) and the data (X) is:

$$f(\theta \mid X) = \frac{f(X \mid \theta) f(\theta)}{f(X)}$$

Where $f(X|\theta)$ is the likelihood, $f(\theta)$ is the prior probability distribution, and $f(X)$ is the marginal likelihood. The marginal likelihood is a normalizing constant that makes the posterior probability distribution integrate to 1. Analytic solutions are rare for the marginal likelihood except for in very simple cases. However, the posterior distribution can be estimated using MCMC methods without calculating the marginal likelihood. A Markov chain is a system that transfers from one state to another and where the next state is only dependent on the present state, not any previous states. The Monte Carlo property translates into an essentially random walk between states. DivBayes uses the Metropolis-Hastings algorithm for the MCMC (Metropolis et al. 1953; Hastings 1970).

The MCMC algorithm:

1. Set initial parameter values (θ).
2. Suggest new parameter values (θ^*) according to a normal distribution centered at the present value.
3. If a random value between 0 and 1 is less than $\frac{f(X \mid \theta^*)}{f(X \mid \theta)} \times \frac{f(\theta^*)}{f(\theta)}$ then accept the suggested state and set $\theta = \theta^*$ (otherwise keep θ as is).
4. At a regular number of iterations, print θ.
5. Go to step 2.

After repeating steps 2-5 infinitely many times, the chain is guaranteed to converge on the posterior distribution. Of course, in reality steps 2-5 cannot be repeated infinitely many times, and the number of iterations needed to reach convergence depends on how efficient the chain is. Methods to judge if the chain has converged on the target (posterior) distribution are treated below. For a more complete introduction to MCMC in Bayesian analysis see, for example, Gamerman and Lopes (2006).

The Likelihood

The probability of observing n number of species at time t after the split from the sister taxon of a group is:

$$p_n = (1-\alpha)\times(1-\beta)\beta^{n-1}, \qquad n\geq 1,$$
$$p_0 = \alpha,$$

where

$$\alpha = \frac{e^{netdiv\times t}-1}{e^{netdiv\times t}\left(\frac{1-extprop}{extprop}+1\right)-1}, \text{ and}$$

$$\beta = \frac{e^{netdiv\times t}-1}{e^{netdiv\times t}-\left(\frac{1}{\frac{1-extprop}{extprop}+1}\right)}.$$

Since we observe only the lineages that have not gone extinct, we are interested in the probability that we observe n species given that n>0:

$$P_{n|n>0}(t) = (1-\beta)\beta^{n-1}.$$

This gives the likelihood formula:

$$l(netdiv,extprop) = \prod_{i}^{k}(1-\beta_i)\beta_i^{n_i-1},$$

where k is the number of taxa (Bailey 1964).

The prior distributions

A uniform distribution between 0 and 1 is used for the relative extinction rate prior. For the net diversification rate, a gamma-distributed prior is assumed. If the ages of the taxa are not known with certainty, DivBayes offers the option to assign normal distributed priors to these.

Installing DivBayes
DivBayes is distributed as C++ source code in a zip archive. This means that the archive needs to be unzipped and that the source code must be compiled before the software can be used. To compile the source code you need a compiler. Most Unix-like systems come with the GNU compiler g++, but for MacOS X you need to install the developer tool kit (Xcode, found on the System DVD bundled with your computer) to get g++. There are several free compilers for Windows. For example, MinGW includes a compiler that works much like g++. The following instructions are for Unix-type systems (Unix, Linux, MacOS X, and BSD); for instructions on how to compile on Windows, please consult the instructions for the specific compiler you use. The first step is to unzip the archive. Many file browsers can unzip files through a simple double click on the mouse, but if you want to do this in a bash environment (such as the Terminal of MacOS X), the easiest way is to be in the same directory (folder; i.e. have that directory/folder as your present working directory) as the DivBayes1.1.zip file. You can change your directory using the command `cd` followed by the directory you want to go. To see what directory you are in type `pwd`, and to see what files and folders are in the directory type `ls -ltr`. When in the same directory as the file DivBayes1.1.zip type:

```
unzip DivBayes1.1.zip
```

After unzipping the archive you may want to enter the directory where the source code file is by typing:

```
cd DivBayes1.1
```

When in the same directory as the source code, you compile the code by typing:

```
g++ -o DivBayes DivBayes1.1.cpp
```

You should now have an executable file named DivBayes. In the unlikely event that you get any error messages while compiling, it is likely that the source code is not compatible with your version of g++. While still in the same folder as the executable file, you can execute DivBayes by typing:

```
./DivBayes -h
```

This should print the help text, listing the different commands for the program.

(Under rare circumstances, the binary file that is produced by the compiler is not executable by default:

```
[user DivBayes1.1] : ./DivBayes -h
-bash: ./DivBayes: Permission denied
```

This is changed through

```
chmod 755 DivBayes
```

)

Getting started

The commands and data for DivBayes are given in a single text file. It is important to keep in mind that it should be pure text format (.txt) and not any format that includes text formatting (e.g. Microsoft documents (.doc) or rich text format (.rtf)). DivBayes is case sensitive. That is, it matters if you type NTAXA, Ntaxa, or ntaxa. The first thing the program needs to know is the number of taxa, i.e. the number of groups for which there is data on age and number of species. So your file should contain the command `ntaxa` followed by the number of taxa, for example:

```
ntaxa 10
```

This will control how the rest of the file is read, so it is important that this number is correct. The second thing DivBayes needs is your data. The number of species for each group is given by `nspecies` followed by a space-separated vector of the number of species for each taxon, for example:

```
nspecies 281 76 50 30 200 4 2011 500 100 507
```

In this particular case, the first taxon contains 281 species, the second 76, and so on.

The age of each group is given by `ages` followed by a space-separated vector of the stem age of each taxon, for example:

```
ages 61 73 59 66 90 91 75 99 125 62
```

It is important that each position in the `age` vector and in the `nspecies` vector corresponds to the same taxon (e.g., 61 correspond to 281, and 73 to 76, in the example below) and that there are as many entries for each of the vectors as given by `ntaxa`. So a text file for a default run could look like this:

```
ntaxa 10
nspecies 281 76 50 30 200 4 2011 500 100 507
ages 61 73 59 66 90 91 75 99 125 62
```

Although this is sufficient to start an MCMC run, you will probably want to set some other parameters, such as the number of generations of the chain and the sample frequency. These are given by `ngen` and `sampfreq`, respectively. Example:

```
ngen 1000000
sampfreq 1000
```

The default model is the Yule (pure birth, extprop=0) model, but you can switch to the birth-death model by giving:

```
	model d
```

which would give a text file looking like this:

```
	ntaxa 10
	nspecies 281 76 50 30 200 4 2011 500 100 507
	ages 61 73 59 66 90 91 75 99 125 62
	ngen 1000000
	sampfreq 1000
	model d
```

If this file is saved as `infile.txt` it can be executed by typing:

```
	./DivBayes infile.txt > outfile.txt
```

This store the output in the file `outfile.txt`. Longer runs can take some time to complete. It is possible to output the results to file and still monitor the progress:

```
	./DivBayes infile.txt > outfile.txt &
	tail –f outfile.txt
```

This uses the Unix command `tail` in its file mode, which allows viewing of files as rows are appended to them. `Ctrl-C` stops `tail`, but it will not stop the underlying DivBayes run. Both the program and the infile can also be called from any directory using their respective full paths. For a complete account of possible settings see *List of commands* below.

The output
DivBayes will produce text-formatted output. The program starts by printing the command it was called with, followed by the settings used for the run and then starts the Markov chain. The settings common to all taxa and the settings for the Markov chain are given first, followed by a tab-separated table with the data and settings for each taxon. The taxon-specific table has the headings:

Taxa gives the taxa in order;
Number_of_species gives the number of species for each taxon (`nspecies`);
Age_mean gives the age of the taxa or, if a prior distribution is given for the age, the mean of the distribution (`ages`);
Age_sd gives the standard distribution of the prior age distribution (equals to zero if absolute values of age are given) (`agessd`);
Net_div_rate_cat gives the rate category of the taxa, where taxa with the same value have the same net diversification rate (`equalND`);
Rel_ext_rate_cat gives the relative extinction rate category of the taxa, where taxa with the same value have the same relative extinction rate (`equalEP`);
Net_div_prior_mean gives the mean of the prior distribution for the net diversification rate (`meanGamma`);

Net_div_prior_shape gives the shape of the gamma distribution for the net diversification rate prior (`shapeGamma`);
Net_div_start gives the start values for the net diversification rates, but if rate categories have been defined, the actual starting rate will be set to the value of the first taxon in that rate category (`netdivvalues`);
Rel_ext_rate_start gives the start values for the relative extinction rate, but if rate categories have been defined, the actual starting rate will be set to the value of the first taxon in that rate category (`extinctpropvalues`).

When the data and settings have been printed, the program initiates the MCMC process. The Markov chain is printed as a tab-separated table with the headings:

Generation specifies what iteration the Markov chain is at;
Log_Lh gives the log likelihood;
Prior_P gives the log of the prior probability for the state;
Post_P gives the log of the posterior probability (LogLh + Prior_P);
Netdiv _n gives the net diversification rate for taxon n;
Rel_ext_rate_n gives the relative extinction rate for taxon n;
Age_n gives the age for taxon n;
Acceptance_rate gives the proportion of the suggested states that has been accepted. This value is dependent on the window width, and as a rule of thumb an acceptance rate between 0.2-0.4 will give the best mixing.

The first row of the actual chain represents the starting values (Generation 0). DivBayes will summarize the output if given the switch `-s` followed by the file name, and the burn-in (see below) can be given by the `-b` switch, for example:

```
      ./DivBayes -s outfile.txt -b 1000
```

The summary statistics include mean, median, and 95% credibility intervals for net diversification rate and relative extinction rate. Alternatively, the output can be read into spreadsheet programs where the appropriate summary statistics can be calculated. If the rows before the heading of the Markov chain are deleted (in a text editor), the output can be read into R for analysis.

Convergence of the chain and effective sampling size

The first iterations of the Markov chain will be dependent on the starting point, and it will take some time for the chain to start sampling from the posterior distribution as unbiased by the starting point. The first iterations must therefore be excluded as a burn-in period. There are several techniques to try to determine if the chain has converged on the posterior distribution. Many of these are implemented in the R package CODA (Plummer et al. 2006). One of the more basic ways is to plot the (log)likelihood scores for each sampling point of the chain and see when they start to fluctuate around some mean instead of reflecting a general increasing trend (Figure 1).

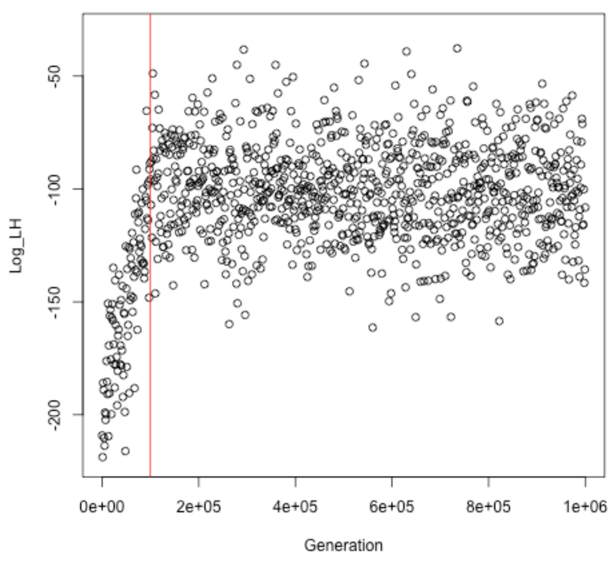

Figure 1. Log likelihood values for an MCMC run. The red line marks the point where the chain levels off and, presumably, has reached the posterior distribution.

It should be noted that not all cases are as clear as the one in Figure 1 and that the likelihood scores can start to increase again after a period of deceptive stability around the same mean. After the burn-in has been determined and removed, you need to make sure that you have enough samples left for a sufficient representation of the posterior distribution. The samples of the Markov chain are not independent since each iteration depend on the previous one. Your samples do not therefore fully qualify as independent samples. It is possible to calculate how many independent samples your chain represents (effective sample size), and CODA has a function to do so. How many samples fewer than your observed sample number this will be depends on how well your chain was mixing (something that can be manipulated by the `windowidth` setting, see below) and how often you sampled the chain. Too high sample frequency will give large output files that may require a lot of memory/disk space and may be slow to work with, but will not increase the effective number of samples that much. How many samples you need depends on what precision you need in your estimation of the posterior distribution. For small simple datasets, sampling every 1,000 generations of a one-million-generation run will often be enough, but for larger and more complex datasets more than ten million generations – with every 10,000 generations sampled - may be needed. The parameter space explored by DivBayes is generally not that complex so the probability of getting stuck in local optima is not high, but to make sure that this is not the case several separate MCMCs can be performed and then compared.

Comparing models using Bayes factors

In Bayesian statistics, different hypotheses (eg. H_1 and H_2) can be compared by posterior odds (Kass and Raftery 1995). If the prior distributions $f(H_2)=1-f(H_1)$ then Bayes' theorem (see above) gives us

$$f(H_{1and2} \mid X) = \frac{f(X \mid H_{1and2}) f(H_{1and2})}{f(X \mid H_1) f(H_1) + f(X \mid H_2) f(H_2)},$$

such that

$$\frac{f(H_1 \mid X)}{f(H_2 \mid X)} = \frac{f(X \mid H_1)}{f(X \mid H_2)} \frac{f(H_1)}{f(H_2)},$$

where $\frac{f(X \mid H_1)}{f(X \mid H_2)}$ = Bayes factors, such that the posterior odds = Bayes factors × prior odds. If the prior probability for each hypothesis is the same, the posterior odds depend only on the Bayes factors. Bayes factors can be estimated from the harmonic mean of the log likelihood from the MCMC. A difference of more than two in two times the difference in harmonic mean of the log likelihood scores (2 * Δ harmonic mean of Log Lh) is taken as *positive* support, more than six as *strong* support, and more than ten as *very strong* support (for the hypothesis with highest mean log likelihood score). The harmonic mean estimate of BFs is not fully robust so it is recommended that several separate MCMCs are performed for each model to validate the results.

List of commands
DivBayes offers a number of options to tweak the chain to be more efficient, relax the assumption that all groups have the same rates, and/or account for uncertainty in the age estimates. The default values are set in a way that assumes that ages are given in million years (Ma) and that net diversification rates are in the order of 0.1 speciations/species/Ma. If your data deviates from this, you should consider setting your own prior and start values. All commands that require a vector to define values for different taxa must have values defined for each taxon, and the value for each taxon must be given in the same order for all commands.

ages gives the mean ages for each group. If no standard deviation is given with `agessd` then `ages` will give the absolute age of the group. Example:

```
    ages 61 73 59 66 90 91 75 99 125 62
```

agessd If the ages are not known with certainty, it is possible to estimate them simultaneously with the diversification rates by treating them as parameters with a prior distribution. DivBayes implements a normal distributed prior with mean given by `ages` and standard deviation given by `agesd`. Example:

```
    agessd 11 13 11 15 11 18 15 12 25 10
```

equalEP (or equalRER) defines what taxa, if any, that have the same relative extinction rate. Three alternatives are available: `yes` (the default), all taxa have the same relative extinction rate; `no`, all taxa have independent relative extinction rate; `string` followed by a space-separated vector, the taxa given the same integer number will have the same relative extinction rate. Example 1:

```
      equalEP yes
```

Example 2:

```
      equalEP no
```

Example 3:

```
      equalEP string 1 2 3 4 5 5 6 6 7 7
```

where taxon 5 and 6 have the same relative extinction rate, as do taxa 7 and 8, and 9 and 10, respectively.

equalND specifies the taxa for which the net diversification rate is the same. Three alternatives are available: `yes` (the default), all taxa have the same net diversification rate; `no`, all taxa have independent rates; `string` followed by a space-separated vector, the taxa given the same integer number will have the same rate. Example 1:

```
      equalND yes
```

Example 2:

```
      equalND no
```

Example 3:

```
      equalND string 1 1 2 2 3 3 4 5 6 7
```

where taxon 1 and 2 have the same rate, as do taxon 3 and 4, and 5 and 6, respectively.

extinctpropvalues (or rel_extinct_rates) gives the starting values for the relative extinction rate. Any such starting value must be greater than 0 but smaller than 1. The command should be followed by a vector specifying the value for each taxon. If two or more taxa have been set to have the same relative extinction rate (with `equalEP`) the actual starting value will be set to the value for the taxon among these that appears first in the vector. Example:

```
      extinctpropvalues 0.1 0.2 0.3 0.1 0.2 0.3 0.1 0.2 0.3 0.1
```

meanGamma gives the mean value for the net diversification rate prior for each taxon. Different taxa may have different prior mean even if they have been defined as having the same net diversification rate using `equalND`. Appropriate values may be inferred from closely related groups with similar age and number of species Example:

```
meanGamma 0.2 0.3 0.1 0.2 0.3 0.1 0.2 0.3 0.1 0.2
```

model defines which of the two models will be used: Yule (relative extinction rate = 0), or Birth-Death. To apply the Yule model give:

```
model y
```

To apply the Birth-Death model give:

```
model d
```

netdivvalues gives the starting values for the net diversification rates. The command should be followed by a vector defining the value for each taxon. If two or more taxa have been set to have the same net diversification rates (with `equalND`), the actual starting value will be set to the value of the taxon among these that appears first in the vector. Example:

```
netdivvalues 0.2 0.3 0.1 0.2 0.3 0.1 0.2 0.3 0.1 0.2
```

ngen gives the number of generations of the MCMC. Example:

```
ngen 100000000
```

nspecies gives the number of species for each taxon. Example:

```
nspecies 281 76 50 30 200 4 2011 500 100 507
```

ntaxa gives the number of taxa. Example:

```
ntaxa 10
```

sampfreq gives the frequency with which to sample the MCMC. Too dense sample frequency will give large output files and potentially highly autocorrelated sample points, whereas too diluted sample frequency may lead to too few sample points and an overly coarse estimation of the posterior probability. The optimal number will depend on how efficient the chain is. Example:

```
sampfreq 10000
```

shapeGamma gives the shape of the gamma distribution for the net diversification rate prior. It only accepts integer values (Erlang distribution). If the shape is set to 1 it is in fact an exponential distribution. `shapeGamma` requires a space-separated vector with a value for each taxon. Different taxa may have differently shaped priors even if they have been defined as having the same net diversification rate using `equalND`. Example:

```
shapeGamma 1 2 3 4 5 1 2 3 4 5
```

windowidth (or windowwidth) controls how different the proposed state in the Markov chain is from the present state by multiplying the proposed change with the given value. For the net diversification rate and the relative extinction rate, the unmodified change is drawn from a standard normal distribution. If the ages are estimated (`agessd` defined) the proposed change for each age is drawn from a standard normal distribution and multiplied with the standard deviation defined for the age of that taxon and then multiplied by the value defined by `windowidth`. The larger the value given for `windowidth` the more of the parameter space is available in one jump but the chance that the proposed value will be accepted decreases. Example:

```
windowidth 0.01
```

Miscellaneous

Both Excel and OpenOffice Calc sometimes insist on trying to interpret numbers as dates or otherwise reformat them in unexpected ways; solutions are found at http://www.webhostingtalk.com/showthread.php?t=601434 (Excel) and http://user.services.openoffice.org/en/forum/viewtopic.php?f=9&t=32979&start=0 (OpenOffice Calc).

DivBayes will ignore lines given within square brackets `[such as this text]` to give the user the opportunity to enter comments in the infile. The square brackets must be balanced for the program to run.

References

Bailey NTJ. 1964. The elements of stochastic processes with applications to the natural sciences. John Wiley and Sons, New York.

Gamerman D, Lopes HF. 2006. Markov chain Monte Carlo – Stochastic simulations for Bayesian Inference. Chapman & Hall/CRC, Boca Raton.

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

Kass RE, Raftery AE. 1995. Bayes factors. Journal of the American Statistical Association 90:773-795.

Plummer M, Best N, Cowls K, Vines K. 2006. CODA: Convergence Diagnosis and Output Analysis for MCMC. R news 7-11.

Metropolis N, Rosenbluth A, Rosenbluth M, Teller A, Teller E. 1953. Equations of state calculations by fast computing machines. The Journal of Chemical Physics 21:1087–1092.