BAli-Phy Tutorial (for version 3.0-beta)

Benjamin Redelings

Table of Contents

1. Introduction
2. Setting up the `~/alignment_files` directory
3. Command line options
4. Analysis 1: variable alignment
5. Analysis 2: fixed alignment
6. Complex substitution models

    6.1. Defaults
    6.2. Rate variation
    6.3. Codon models
    6.4. Fixing parameter values
    6.5. Priors

7. Generating an HTML Report

    7.1. Inspecting output files
    7.2. Command-line summary of output files
    7.3. Generating an HTML Report

8. Starting and stopping the program
9. Specifying the model for each partition

    9.1. Using different substitution models
    9.2. Disabling alignment estimation for some partitions
    9.3. Sharing model parameters between partitions
    9.4. Sharing substitution rates between partitions

10. Option files
11. Dataset preparation

    11.1. Splitting and Merging Alignments
    11.2. Shrinking the data set
    11.3. Cleaning the data set

1. Introduction

Before you start this tutorial, please download and install bali-phy, following the installation instructions in the manual.

2. Setting up the `~/alignment_files` directory

Go to your home directory:

```
% cd ~
```

Make a directory called alignment_files inside it:

```
% mkdir alignment_files
```

Go into the `alignment_files` directory:

```
% cd alignment_files
```

Download the example alignment files:

```
% wget http://www.bali-phy.org/examples.tgz
```

Alternatively, you can use **curl**

```
% curl -O http://www.bali-phy.org/examples.tgz
```

Extract the compressed archive:

```
% tar -zxf examples.tgz
```

Take a look inside the `examples` directory:

```
% ls examples
```

Take a look at an input file:

```
% less examples/5S-rRNA/5d.fasta
```

Get some information about the alignment:

```
% alignment-info examples/5S-rRNA/5d.fasta
```

3. Command line options

What version of bali-phy are you running? When was it compiled? Which compiler? For what computer type?

```
% bali-phy -v
```

Look at the list of command line options:

```
% bali-phy --help | less
```

Some options have a short form which is a single letter:

```
% bali-phy -h | less
```

You can also show help for advanced options:

```
% bali-phy --help=advanced | less
% bali-phy --help=expert | less
```

You can get help on the command line options:

```
% bali-phy --help=iterations | less
```

You can also get help on models, distributions, and functions:

```
% bali-phy --help=TN | less
% bali-phy --help=Normal | less
% bali-phy --help=quantile
```

4. Analysis 1: variable alignment

Let's do an analysis of intergenic transcribed spacer (ITS) genes from 20 specimens of lichenized fungi (Gaya et al, 2011 analyzes 68). This analysis will estimate the alignment, phylogeny, and evolutionary parameters using MCMC.

This data set is divided into three gene regions, or partitions. It is assumed that all genes evolve on the same tree, but may have different rates and evolutionary parameters. Let's look at the sequences. How long are they?

```
% cd ~/alignment_files/examples/ITS
% alignment-info ITS1.fasta
% alignment-info 5.8S.fasta
% alignment-info ITS2.fasta
```

By default, each gene gets a default substitution model based on whether it contains DNA/RNA or amino acids. By running bali-phy with the `--test` option, we can reveal what substitution models and priors will be used, without actually starting a run.

```
% bali-phy ITS1.fasta 5.8S.fasta ITS2.fasta --test
```

Now start a run. BAli-Phy will create an directory called `ITS1-5.8S-ITS2-1` to hold output files from this analysis.

```
% bali-phy ITS1.fasta 5.8S.fasta ITS2.fasta &
```

Run another copy of the same analysis, which will create a new directory called `ITS-2` for its own output files. This additional run will take advantage of a second processor, and will also help detect when the runs have performed enough iterations.

```
% bali-phy ITS1.fasta 5.8S.fasta ITS2.fasta &
```

You can take a look at your running jobs. There should be two bali-phy jobs running.

```
% jobs
```

After each iteration, one line containing values for numerical parameters (such as the nucleotide frequencies) is written to the files `ITS1-5.8S-ITS2-1/C1.log` and `ITS1-5.8S-ITS2-2/C1.log`. So we can check the number of iterations completed by looking at the number of lines in these files:

```
% wc -l ITS1-5.8S-ITS2-*/C1.log
```

The mean or the median of these values can be used as an estimate of the parameter. The variance indicates the degree of uncertainty. Let's look at the initial parameter estimates:

```
% statreport ITS1-5.8S-ITS2-1/C1.log ITS1-5.8S-ITS2-2/C1.log | less
% statreport ITS1-5.8S-ITS2-1/C1.log ITS1-5.8S-ITS2-2/C1.log --mean | less
```

The program Tracer graphically displays the posterior probability distribution for each parameter. (If you are using Windows or Mac, first run Tracer, and then press the + button to add the files.)

```
% tracer ITS1-5.8S-ITS2-1/C1.log ITS1-5.8S-ITS2-2/C1.log &
```

How does the evolutionary process for these genes differ in:

1. substitution rate? (Scale<p>)
2. insertion-deletion rate? (I<p>.logLambda)
3. nucleotide frequencies? (S<p>.TN.F.pi.{A,T,G,C})
4. number of indels? (#indels)

5. Analysis 2: fixed alignment

Let's also start a *fixed alignment* analysis. This will estimate the tree and evolutionary parameters, but keep the alignment constant, similar to running MrBayes, BEAST, PhyloBayes, or RevBayes.

```
% bali-phy ITS1.fasta 5.8S.fasta ITS2.fasta --name ITS-fixed -Inone &
```

The `-Inone` is a short form of `--imodel=none`, where *`imodel`* means the insertion-deletion model. When there's no model of insertions and deletions, then the alignment must be kept fixed.

The `--name` option means that output files will be created in the directory `ITS-fixed-1`.

6. Complex substitution models

While those analyses are running, let's look at how to specify more complex substitution models in bali-phy.

6.1. Defaults

When you don't specify values for parameters like *imodel*, bali-phy uses sensible defaults. For example, these two commands are equivalent:

```
% cd ~/alignment_files/examples/
% bali-phy 5S-rRNA/25-muscle.fasta --test
% bali-phy 5S-rRNA/25-muscle.fasta --test --alphabet=RNA --smodel=TN --imodel=RS07
```

You can change the substitution model from the Tamura-Nei model to the General Time-Reversible model:

```
% bali-phy 5S-rRNA/25-muscle.fasta --test --smodel=GTR
```

6.2. Rate variation

You can also allow different sites to evolve at 5 different rates using the gamma[4]+INV model of rate heterogeneity:

```
% bali-phy 5S-rRNA/25-muscle.fasta --test --smodel=GTR+Rates.Gamma[4]+INV
```

You can allow 5 different rates that are all independently estimated:

```
% bali-phy 5S-rRNA/25-muscle.fasta --test --smodel=GTR+DP[n=5]
```

6.3. Codon models

We can also conduct codon-based analyses using the Nielsen and Yang (1998) M0 model of diversifying positive selection (dN/dS):

```
% bali-phy Globins/bglobin.fasta --test --smodel=M0
```

The M0 model takes a nucleotide exchange model as a parameter. This parameter is optional, and the default is HKY, which you could specify as M0[,HKY]. You can change this to be more flexible:

```
% bali-phy Globins/bglobin.fasta --test --smodel=M0[,GTR]
```

You can make the codon frequencies to be generated from a single set of nucleotide frequencies:

```
% bali-phy Globins/bglobin.fasta --test --smodel=M0[,GTR]+MG94
```

The M7 model allows different sites to have different dN/dS values, where the probability of dN/dS values follows a beta distribution:

```
% bali-phy Globins/bglobin.fasta --test --smodel=M7
```

The M7 model has parameters as well. Here are the defaults:

```
% bali-phy Globins/bglobin.fasta --test --smodel=M7[4,HKY,F61]
```

The M3 model allows different sites to have different dN/dS values, but directly estimates what these values are:

```
% bali-phy Globins/bglobin.fasta --test --smodel=M3[n=3]
```

The M8a_Test model allows testing for positive selection in some fraction of the sites:

```
% bali-phy Globins/bglobin.fasta --test --smodel=M8a_Test[4,HKY,F3x4]
```

6.4. Fixing parameter values

We can use the TN+Gamma[4]+INV model without specifying parameters:

```
% bali-phy Globins/bglobin.fasta --test --smodel=TN+Rates.Gamma+INV
```

However, we can also fix parameter values:

```
% bali-phy Globins/bglobin.fasta --test --smodel=TN+Rates.Gamma[n=4,alpha=1]+INV[pInv=0.2]
```

Here we have set the shape parameter for the Gamma distribution to 1, and the fraction of invariant sites to 20%. Since these parameters are fixed, they will not be estimated and their values will not be shown in the log file.

You can see the parameters for a model by using the `help` command, as in:

```
% bali-phy help Rates.Gamma
```

This will show the default value or default prior for each parameter, if there is one.

6.5. Priors

By default the fraction of invariant sites follows a Uniform[0,1] distribution:

```
% bali-phy help INV
```

However, we can specify an alternative prior:

```
% bali-phy Globins/bglobin.fasta --test -S TN+Rates.Gamma[n=4]+INV[pInv~Uniform[0,0.2]]
```

We can also specify parameters as positional arguments instead of using variable names:

```
% bali-phy Globins/bglobin.fasta --test -S TN+Rates.Gamma[4]+INV[~Uniform[0,0.2]]
```

Here "`~`" indicates a sample from the Uniform distribution instead of the distribution itself.

The insertion-deletion model also has parameters.

```
% bali-phy help RS07
```

Here the default value for meanIndelLength is Add[1,~Exponential[10]]. This indicates a random value that is obtained by sampling an Exponential random variable with mean 10 and then adding 1 to it.

7. Generating an HTML Report

See Section 6: Output of the manual for more information about this section.

7.1. Inspecting output files

Look at the directories that were created to store the output files:

```
% cd ~/alignment_files/examples/ITS/
% ls
```

Here is a shell trick with curly braces {} to avoid typing some things multiple times, illustrated with the `echo` command. Let's see how many more iterations have been completed since we last checked:

```
% echo "hi!"
% echo ITS1-5.8S-ITS2-{1,2}/C1.log
% wc -l ITS1-5.8S-ITS2-{1,2}/C1.log
```

Examine the file containing the sampled trees:

```
% less ITS1-5.8S-ITS2-1/C1.trees
```

Examine the file containing the sampled alignments:

```
% less ITS1-5.8S-ITS2-1/C1.P1.fastas
```

Examine the file containing the successive best alignment/tree pairs visited:

```
% less ITS1-5.8S-ITS2-1/C1.MAP
```

The beginning of the `C1.out` file also contains a lot of information about what command was run, when it was run, and what the process ID (PID) of the running program is.

```
% less ITS1-5.8S-ITS2-1/C1.out
```

7.2. Command-line summary of output files

Try summarizing the sampled numerical parameters (e.g. not trees and alignments).

```
% statreport ITS1-5.8S-ITS2-{1,2}/C1.log --mean --median > Report
% less Report
```

Now lets compute the consensus tree for these two runs. When figtree asks for a name for the branch supports, type `PP` in the box.

```
% trees-consensus ITS1-5.8S-ITS2-{1,2}/C1.trees > c50.PP.tree
% figtree c50.PP.tree &
```

To look at the posterior probability of individual splits in the tree:

```
% trees-consensus ITS1-5.8S-ITS2-{1,2}/C1.trees --report=consensus > c50.PP.tree
% less consensus
```

Now lets see if there is evidence that the two runs have not converged yet:

```
% trees-bootstrap ITS1-5.8S-ITS2-{1,2}/C1.trees > partitions.bs
% grep ASDSF partitions.bs
```

The ASDSF (Average Standard-Deviation of Split Frequences) is a measure of how much the estimated posterior probability of splits differ between the two runs. If it is greater than 0.01 then you should probably accumulate more iterations. The MSDSF (Maximum SDSF) indicates the variation between runs for the worst split.

7.3. Generating an HTML Report

Now lets use the analysis script to run all the summaries and make an HTML report:

```
% bp-analyze ITS1-5.8S-ITS2-{1,2}/
% firefox Results/index.html &
```

This PERL script runs *`statreport`* and *`trees-consensus`* for you. Take a look at what commands were run:

```
% less Results/bp-analyze.log
```

The report should give us an indication of

1. What is the majority consensus tree?
2. What is consensus alignment for each partition?
3. How much alignment uncertainty is there in each partition?
4. How much do the split frequencies differ between runs?
5. What is the effective sample size (Ne) for the Scale[1]? For |A1|?

8. Starting and stopping the program

We didn't specify the number of iterations to run in the section above, so the two analyses will run for 100,000 iterations, or until you stop them yourself. See Section 10: Convergence and Mixing: Is it done yet? of the manual for more information about when to stop an analysis.

Let's stop the bali-phy runs now. In order to stop a running job, you need to kill it. We can use the PID mentioned in C1.out to kill a specific process. Let's kill the fixed-alignment analysis first.

```
% less ITS-fixed-1/C1.out
% kill PID
% jobs
```

We can also kill all running bali-phy processes:

```
% killall bali-phy
```

However, beware: if you are running multiple analyses, this will terminate all of them.

9. Specifying the model for each partition

For analyses with multiple partitions, we might want to use different models for different partitions. When two partitions have the same model, we might also want them to have the same parameters. This is described in more detail in section 4.3 of the manual.

9.1. Using different substitution models

Now lets specify different substitution models for different partitions.

```
% cd ~/alignment_files/examples/ITS
% bali-phy {ITS1,5.8S,ITS2}.fasta --smodel=1:GTR --smodel=2:HKY --smodel=3:TN --test
```

9.2. Disabling alignment estimation for some partitions

We can also disable alignment estimation for some, but not all, partitions:

```
% bali-phy {ITS1,5.8S,ITS2}.fasta --imodel=1:RS07 --imodel=2:none --imodel=3:RS07 --test
```

Specifying `--imodel=none` removes the insertion-deletion model and parameters for partition 2 and also disables alignment estimation for that partition.

Note that there is no longer an I3 indel model. Partition #3 now has the I2 indel model.

9.3. Sharing model parameters between partitions

We can also specify that some partitions with the same model also share the same parameters for that model:

```
% bali-phy {ITS1,5.8S,ITS2}.fasta --smodel=1,3:GTR --imodel=1,3:RS07 --smodel=2:TN --imodel=2:none --test
```

This means that the information is *pooled* between the partitions to better estimate the shared parameters.

Take a look at the model parameters, and the parentheticals after the model descriptions. You should see that there is no longer an S3 substitution model or an I3 indel model. Instead, partitions #1 and #3 share the S1 substitution model and the I1 indel model.

9.4. Sharing substitution rates between partitions

We can also specify that some partitions share the same scaling factor for branch lengths:

```
% bali-phy {ITS1,5.8S,ITS2}.fasta --smodel=1,3:GTR --imodel=1,3:RS07 --smodel=2:TN --imodel=2:none --scale=1,3: --
test
```

This means that the branch lengths for partitions 1 and 3 are the same, instead of being independently estimated.

Take a look at the model parameters. There is no longer a Scale[3] parameter. Instead, partitions 1 and 3 share Scale[1].

10. Option files

You can collect command line options into a file for later use. Make a text file called `analysis1.script`:

```
align = ITS1.fasta
align = 5.8S.fasta
align = ITS2.fasta
smodel = 1,3:TN+DP[n=3]
smodel = 2:TN
imodel = 2:none
scale = 1,3:
```

You can test the analysis like this:

```
% bali-phy -c analysis1.script --test
```

You can run it like this:

```
% bali-phy -c analysis1.script &
```

11. Dataset preparation

11.1. Splitting and Merging Alignments

BAli-Phy generally wants you to split concatenated gene regions in order to analyze them.

```
% cd ~/alignment_files/examples/ITS-many/
% alignment-cat -c1-223 ITS-region.fasta > 1.fasta
% alignment-cat -c224-379 ITS-region.fasta > 2.fasta
% alignment-cat -c378-551 ITS-region.fasta > 3.fasta
```

Later you might want to put them back together again:

```
% alignment-cat 1.fasta 2.fasta 3.fasta > 123.fasta
```

11.2. Shrinking the data set

You might want to reduce the number of taxa while attempting to preserve phylogenetic diversity:

```
% alignment-thin --down-to=30 ITS-region.fasta > ITS-region-thinned.fasta
```

You can specify that certain sequences should not be removed:

```
% alignment-thin --down-to=30 --keep=Csaxicola420 ITS-region.fasta > ITS-region-thinned.fasta
```

11.3. Cleaning the data set

Keep only sequences that are not too short:

```
% alignment-thin --longer-than=250 ITS-region.fasta > ITS-region-long.fasta
```

Remove 10 sequences with the smallest number of conserved residues:

```
% alignment-thin --remove-crazy=10 ITS-region.fasta > ITS-region-sane.fasta
```

Keep only columns with a minimum number of residues:

```
% alignment-thin --min-letters=5 ITS-region.fasta > ITS-region-censored.fasta
```