EmpPrior v0.2: using empirical data to inform branch-length priors for Bayesian phylogenetics

January 13th, 2015

John J. Andersen[1]
Bradley J. Nelson[1]
Jeremy M. Brown[1,2]

[1] Department of Biological Sciences, Louisiana State University, Baton Rouge, Louisiana, USA

[2] jembrown@lsu.edu

http://www.phyleauxgenetics.org/

Contents

1. What is EmpPrior?

Prior distributions can have a strong effect on the results of Bayesian analyses. However, no general consensus exists for how priors should be set in all circumstances. Branch-length priors are of particular importance for phylogenetics, because they simultaneously set a prior distribution on all branch-length parameters and biologically relevant inferences have been shown to be sensitive to the chosen prior distribution.

Many researchers use the default prior settings provided by programs like MrBayes (Ronquist et al., 2012), although these default branch-length priors can sometimes lead to tree-length estimates that are incongruent with maximum-likelihood estimates (Brown et al, 2010; Marshall, 2010; Zhang et al., 2012). In other words, researchers can inadvertently specify strongly informative prior information about the lengths of branches in their tree that conflicts with the information provided by the data.

Why does this happen? Recent versions of MrBayes (through v3.2.2), and several other Bayesian phylogenetic programs, have by default placed exponentially distributed priors on the lengths of branches with a mean of 0.1 (λ=10). In addition, these priors are assumed to be independent and identically distributed (i.i.d.) across branches. In this framework, the exponential branch-length priors set an implicit prior on tree length that increases linearly with the number of taxa and can effectively exclude reasonable values of tree length. This problem is worst when the number of branches in the tree is large and the average lengths of these branches is much less than 0.1 – a situation that often occurs in studies with dense intraspecific sampling (i.e., phylogeographic studies). Such datasets have provided clear examples of this phenomenon (see Brown et al., 2010 and Marshall, 2010 for exemplars).

The incongruence between specified priors and maximum-likelihood estimates is concerning, because branch and tree lengths form the basis for many downstream inferences (e.g., comparative methods and divergence time estimation) and choice of branch-length prior has been shown to influence the inferred posterior probabilities of branches and topologies (Yang and Rannala, 2005). To date, the goal of most researchers has been to specify priors that are as uninformative as possible. Clearly, that goal was not met by the default exponential distributions. Recently, Rannala et al. (2012) proposed a new compound Dirichlet prior on tree and branch lengths. This prior seems to have better default behavior than the exponential (Zhang et al., 2012) and should be preferred. The compound Dirichlet has become the default prior in MrBayes v3.2.3.

Regardless of the distribution chosen, default parameterizations are meant to work reasonably well in most circumstances. However, they are not guaranteed to perform well in all circumstances. Beyond that, they do not incorporate any previous information that the researcher (or others) may possess about what constitutes a reasonable branch or tree length for

the phylogeny of interest. We have created EmpPrior to help researchers incorporate the information held in other, relevant datasets to set branch- and tree-length priors for new analyses.

EmpPrior is made up of two components: EmpPrior-search and EmpPrior-fit. EmpPrior-search is a Java application that quickly and efficiently queries the TreeBASE repository and returns nexus files matching specified search criteria (gene name and the number of sequences desired). EmpPrior-search is exceptionally easy to run and will automatically parse out genes of interest from datasets containing multiple loci (when sufficient information was provided by those depositing the data in TreeBASE). EmpPrior-fit is an R script that uses maximum likelihood (ML) to fit branch- and tree-length distributions to trees inferred from datasets harvested by EmpPrior-search. In between these two steps, researchers will need to conduct their own ML phylogeny inference for the outside datasets they wish to use. For that step, we recommend the use of Garli (Zwickl, 2006) or another program that conducts a full, thorough ML tree search.

The goal of EmpPrior is to encourage researchers to think more carefully about their branch- and tree-length priors and to use relevant information, when it is available, to set these priors. The use of outside data also avoids the circularity, and corresponding artificial reduction in uncertainty, inherent to empirical Bayesian analysis.

2. Installing EmpPrior

EmpPrior can be obtained from http://code.google.com/p/empprior. The download is a .zip file containing a compiled .jar version of EmpPrior-search (`EmpPrior-search.jar`), the Java source code and dependencies for EmpPrior-search, a script to compile EmpPrior-search, and a command-line R script for EmpPrior-fit (`EmpPrior-fit.r`). If you have a recent version of Java installed (the code was compiled and tested on version 1.6.0_65), the downloaded version of `EmpPrior-search.jar` should run as distributed.

If you need to install Java, go here: https://java.com/en/download/manual.jsp.

If you need to recompile EmpPrior-search on a Mac, open up Terminal, change directories to the EmpPrior folder (something like “cd /path/to/EmpPrior.0.2”), then type:

```
> ./compile.sh
```

Running this script should create a new, working version of `EmpPrior-search.jar`. Feel free to delete the .class files after you have re-compiled.

Before running EmpPrior-fit, make sure that a recent version of R is installed on your computer, along with the packages ape and bbmle.

If you need to install R, go here: http://www.r-project.org.

To install the other packages, simply open up R and type:

```
> install.packages("package_name")
```

[NOTE: Be sure to put the quotes around the package name.] You will then be asked to select a CRAN mirror. Pick your favorite location, click OK, and R should do the rest. Run this command once for each of the required packages (ape and bbmle).

3. Running EmpPrior

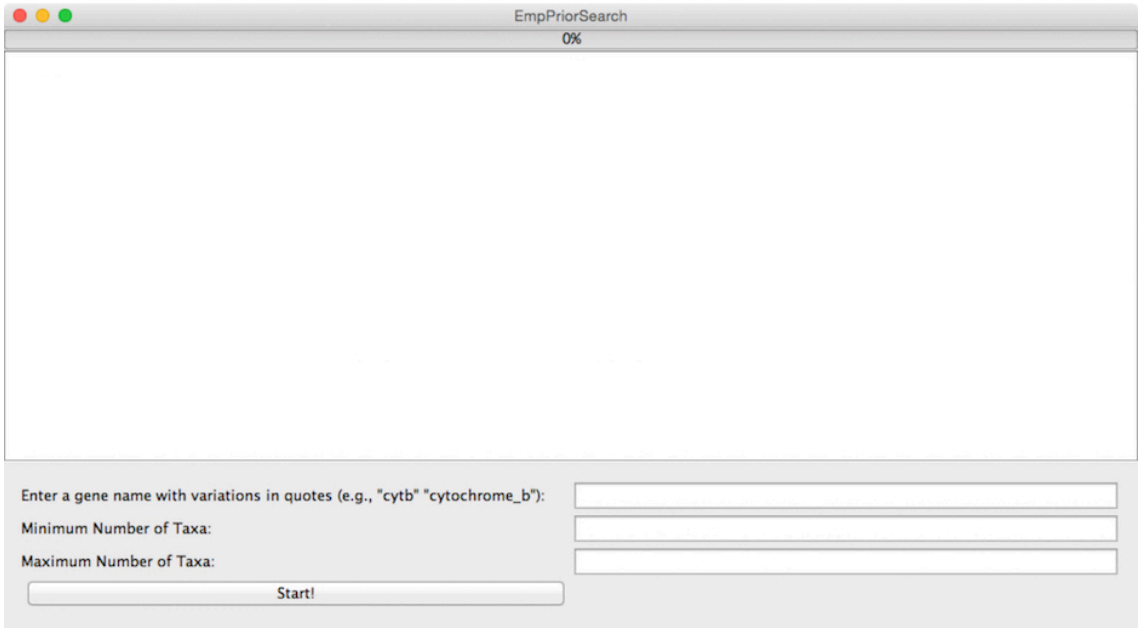
3.1 EmpPrior-search

To launch EmpPrior-search on Mac OS X, you can use either of 2 approaches: (1) You should be able to simply double click the jar file (`EmpPrior-search.jar`). (2) If you want to launch the jar from the command line, navigate to the folder that contains `EmpPrior-search.jar` and type:

```
> java -jar EmpPriorSearch.jar
```

[NOTE: EmpPrior-search relies on code contained in the `commons-lang3-3.1.jar` file. If you move `EmpPrior-search.jar` out of its original folder, you will also need to move `commons-lang3-3.1.jar` with it. Failure to do so will cause EmpPrior-search to throw errors and result in blank output files.]

Doing either (1) or (2) should open up a graphical user interface (GUI), that looks like this:

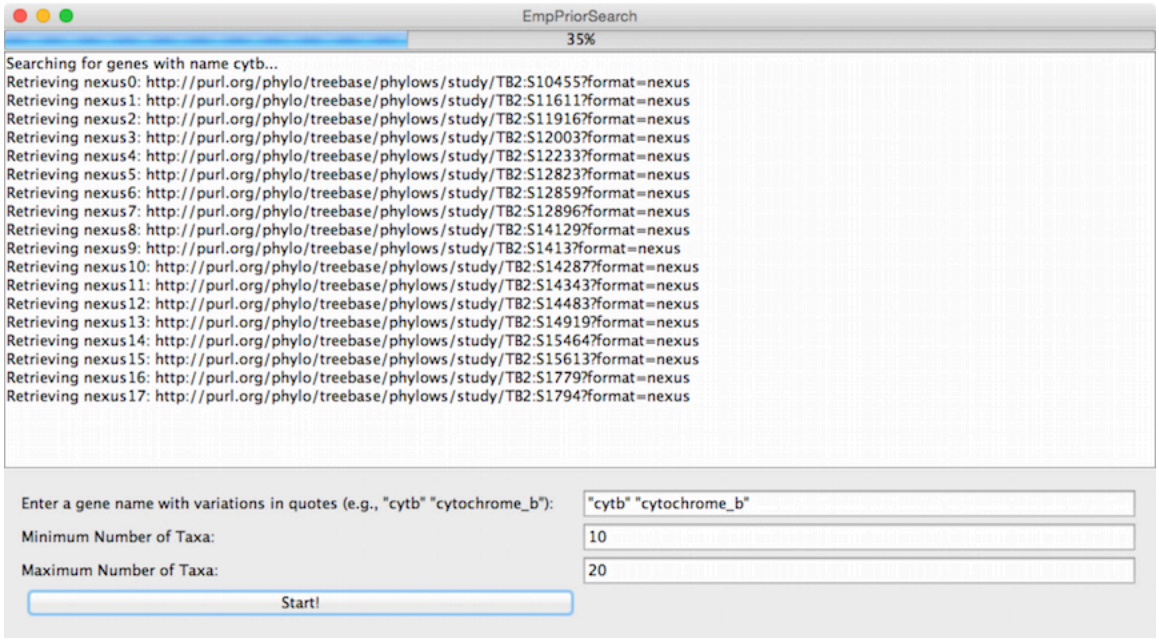


User options can be entered into the text fields at the bottom of the window. The options are pretty self-explanatory. In the first field, enter the name of the locus for which you want to search (in quotations), along with any common variants of the name or abbreviations. For instance, if you are looking for *cytochrome b*, you could enter: “cytb” “cytB” “CytB” “cytochrome_b” “cytochrome b”, etc. EmpPrior-search will delete any duplicate datasets returned by searches using different names, because it assumes that all names entered are synonyms. Therefore, please conduct separate searches for different genes.

The next two text boxes simply provide bounds on the number of sequences that are included in the datasets returned by EmpPrior-search.

When you have entered the appropriate information into these three boxes, click “Start!”. Because EmpPrior-search needs to access TreeBASE, you must be connected to the internet.

As EmpPrior-search runs, the progress bar at the top of the window will give you a rough idea of how much progress the program has made. The large text window in the middle of the screen will provide information about what EmpPrior-search is doing. Here is a snapshot from the middle of a run:

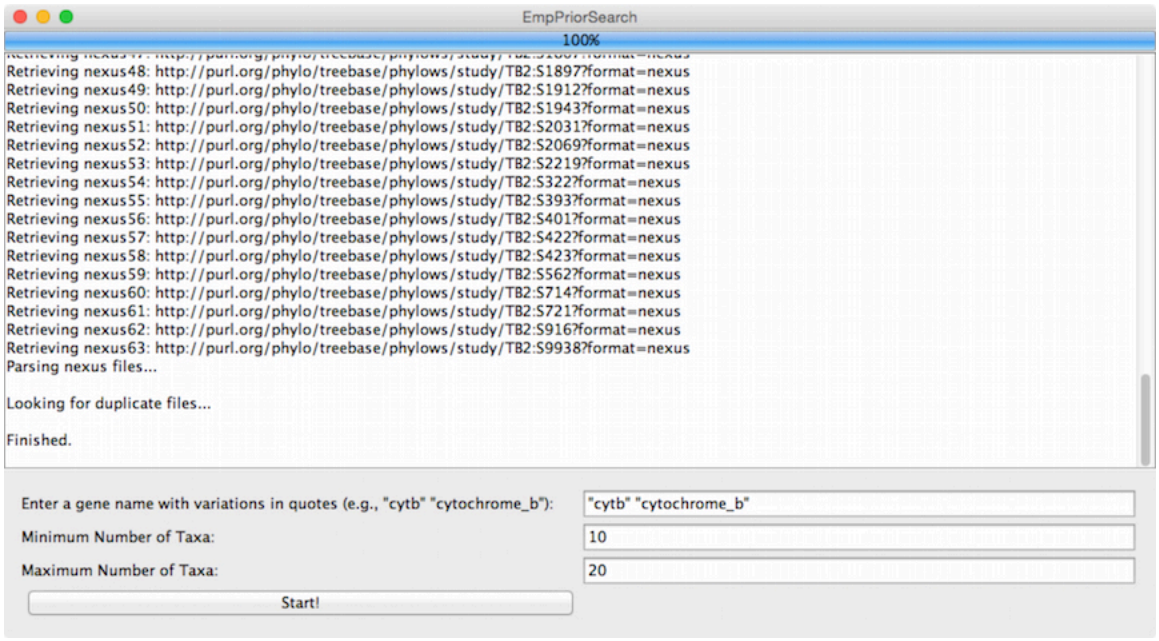


The first line in the output window shows that EmpPrior-search has started its first search using “cytb” as the keyword. Each subsequent line corresponds to a separate dataset identified with this keyword and includes the PURL used to obtain that file. These lines also include the corresponding TreeBASE Study IDs (e.g., S10455). EmpPrior-search quickly assesses whether each of these datasets matches the minimum and maximum taxon number requirements and

deletes those that do not. Those datasets that do match this requirement are stored in the same folder from which EmpPrior-search was launched.

If other genes are included in the datasets returned by these searches, EmpPrior-search will attempt to parse out just the gene of interest. However, doing so requires that those researchers who deposited the original dataset defined the boundaries for different genes. If they did not, EmpPrior-search will return the concatenated matrix and it is up to the user to either extract the relevant gene or delete the file.

Here is a screenshot from the end of this run:



The dataset retrieval lines shown here correspond to the last keyword entered – in this case, “cytochrome_b”. Note that while 63 datasets were found with the keyword “cytochrome_b”, only a handful satisfied the requirement of containing between 10 and 20 sequences.

After searches have been conducted with all keywords, EmpPrior-search looks for and deletes any duplicate datasets. For this search, six files survived all filters. Listing the nexus files in the directory from which EmpPrior-search was run reveals this list:

```
S562_cytochrome_b.nex
S1342_cytochrome_b.nex
S1794_cytb.nex
S1832_cytb.nex
S11611_cytb.nex
S12233_cytb.nex
```

The first part of each filename corresponds to the TreeBASE Study ID. Each of these IDs should be unique for those files that were retained. The second part of each filename corresponds to the keyword used to find that file. Note that some datasets are returned by multiple searches, but only the version found first is retained.

At this point, the user should manually scan the headers of all nexus files to identify those that contain concatenated matrices consisting of multiple genes. Users will also need to exercise judgment when deciding whether the level of divergence of each dataset is sufficiently similar to the focal data as to provide useful prior information. We have used taxonomic rank as a proxy for divergence and found that it works reasonably well, at least at the level of genera in animals (Nelson et al., 2015).

One final note: If a nexus file contains multiple concatenated matrices that all include the gene of interest, it will parse out the gene (when possible) from the first concatenated matrix in the file. This behavior is currently fixed and ignores differences in taxon sampling across concatenated matrices. We hope to make this behavior more flexible in the future.

3.2 Inferring ML trees from relevant datasets

In order to find informed estimates of branch-length distributions, users will need to infer phylogenetic trees and estimate branch lengths from the outside, relevant datasets returned by EmpPrior-search. There are a number of programs that already exist for this purpose, so we do not attempt to provide this functionality here. We recommend Garli (Zwickl, 2006; https://code.google.com/p/garli/), but others are available. We do strongly recommend that users conduct a thorough tree search with accurate branch-length optimization. We have found that algorithmic methods (e.g., neighbor joining) and less thorough ML searches often result in branch-length distribution parameter estimates that do not perform as well in subsequent Bayesian analyses.

Please see the documentation for your chosen ML tree search program to infer phylogenies. Nexus files containing inferred trees with branch lengths will be required to run EmpPrior-fit.

3.3 EmpPrior-fit

The executable file `EmpPrior-fit.r` contains code for the R statistical computing language (http://www.r-project.org) that takes phylogenetic trees with branch lengths as input and fits two branch- and tree-length distributions: the i.i.d. exponential and the compound Dirichlet (Rannala et al., 2012). For the exponential, only one parameter is estimated – the rate, λ. For the compound Dirichlet, up to 3 parameters can be estimated in various combinations – the rate (β_T) of the gamma distribution on tree length, the concentration (α) of the Dirichlet distribution

partitioning the total tree length among branches, and the mean ratio of internal to external branch lengths (c). EmpPrior-fit leaves the shape (α_T) of the gamma distribrution on tree length fixed at 1, because we have found that joint estimation of α_T and β_T can be unreliable. For the sake of compactness, and because the exponential is a special case of the gamma, EmpPrior-fit reports the rate (β_T) of the gamma on tree length for the compound Dirichlet in the same column as the exponential rate (λ). This column is labeled "betaT", but note that it applies to the rates of the i.i.d. exponential distributions for the row labeled "fit.expon".

To execute EmpPrior-fit, navigate in Terminal to the folder containing `EmpPrior-fit.r`. If you have a single tree to fit and it's in the same folder as EmpPrior-fit, simply type:

```
> ./EmpPrior-fit.r myTreeFile.tre outfile=myOutputFile.out
```

The user will need to substitute appropriate tree and output file names in place of "`myTreeFile.tre`" and "`myOutputFile.out`". If you receive a permissions error when you try to run the script, you may need to add execute permissions to the script. This can be done by typing:

```
> chmod +x EmpPrior-fit.r
```

Your tree can be stored in another folder, if you provide the path to the tree file as part of the command-line argument:

```
> ./EmpPrior-fit.r /path/to/file/myTreeFile.tre outfile=myOutputFile.out
```

You can also simultaneously fit distributions to multiple trees, with the output directed to the same file:

```
> ./EmpPrior-fit.r treeFile1.tre treeFile2.tre treeFile3.tre outfile=outFile.out
```

You can even have EmpPrior-fit simultaneously fit all tree files in a folder by simply providing the path to the folder. However, the folder must contain only tree files.

```
>> ./EmpPrior-fit.r folder=/path/to/treeFolder outfile=myOutputFile.out
```

By default, output from EmpPrior-fit is printed to the Terminal. If an outfile name is specified (as in the lines above), this file will be created and the screen output will be mirrored in the file.

4. An example and the interpretation of EmpPrior output

Suppose we were performing a phylogeographic assessment of two frog species found within the genus *Acris* (Gamble et al., 2008). Our data set contains 66 sequences of *cytochrome b*: 43 sequences from *A. crepitans* and 23 sequences from *A. gryllus*. To generate informed parameterizations for our priors we first need to target our gene of interest and brainstorm variations of the gene name. For *cytochrome b*, many researchers will abbreviate the gene to *cyt b*. Since whitespace can be an issue, we can add other potential variations, such as: *cytb* and *cytochrome_b*. We should include all of the names within the gene name text box at the bottom of the EmpPrior-search window, with each variant in a set of double quotes. Next, we can set a minimum and maximum number of sequences, in order to keep the size of the dataset and, hopefully, the distribution of branch lengths similar to our focal data. The bounds on number of sequences used for any particular search will require the user to exercise some judgement. How many previous studies are likely to have used this gene? At what scale? One strategy may be to start with a narrow window and then expand outward.

The next step is to carefully look through each nexus file that has been saved. We have employed a 4-step procedure for each file to find acceptable matches:

1) Does the file contain only sequences relevant to the gene you are targeting? Some files may have been submitted incorrectly by not providing the correct coordinates for EmpPrior to parse out relevant information. Be careful to look for nexus files containing genes that are not of interest.

2) Does the nexus file target sequences that have similar taxonomic depth? For example, we are interested in genus-level files because we have two different species in our *Acris* data set. Therefore, we can remove files that are below the species level or above the family level.

3) Is the total number of sequences similar to the focal data? Again, some judgement is required here and the stringency may depend on how many total data sets have passed the first two filters.

4) Are the number of sequences per clade/species similar to your own data set? This last check will probably take the most time. Fortunately, the number of files to examine will have drastically decreased if you employed the first three filters. Continuing with our *Acris* example data set, we would ideally like to find files that roughly match the number of sequences for each species. Therefore, we should look for files that contain roughly 40 sequences for one species, and 20 sequences for the other. Note that the sampling of sequences can have a meaningful impact on the distribution of branch lengths in the tree,

even for data sets with an identical number of sequences. For instance, if the focal data set has randomly sampled intraspecific variation in one species with single exemplar sequences from several other species included as outgroups, but the outside data set has attempted to sample the most divergent lineages across all species in the genus, the distribution of branch lengths will be different and the parameter estimates fitted to the outside data will become less useful for analysis of the focal data.

Hopefully, some data sets survive all four of these filters. If so, we can move them to a new folder called "`acrisData`" and run ML tree searches for each data set. See the documentation for your favorite ML program like Garli for more information. Once these trees have been generated, we can place them in another folder called "`acrisTrees`" and run EmpPrior-fit:

```
> ./EmpPrior-fit.r folder=./acrisTrees/ outfile=./EmpPrior_acrisTrees.out
```

The EmpPrior-fit output will include a table for each of the relevant trees that looks something like this:

	file	negLogL	df	dAICc	weight
fit.compDirichlet.ac	AcrisTest1.best.tre	-480.362	2	0	0.731
fit.compDirichlet.Tac	AcrisTest1.best.tre	-480.438	3	2.009	0.268
fit.gamma	AcrisTest1.best.tre	-473.771	2	13.1830	0.001
fit.compDirichlet.a	AcrisTest1.best.tre	-472.267	1	14.0856	0.001
fit.compDirichlet.Ta	AcrisTest1.best.tre	-472.343	2	16.0391	0
fit.compDirichlet.c	AcrisTest1.best.tre	-304.248	1	350.1233	0
fit.expon	AcrisTest1.best.tre	-303.813	1	350.9938	0
fit.compDirichlet.Tc	AcrisTest1.best.tre	-304.324	2	352.0768	0
fit.compDirichlet.T	AcrisTest1.best.tre	-301.542	1	355.5358	0

TL.mean	alphaT	betaT	concentration	bl.ratio
1	1	1	0.1498	2.6516
0.6599	1	1.515	0.1498	2.6524
0.6598	15.0122	22.7539	1	1
1	1	1	0.1863	1
0.6605	1	1.5140	0.1862	1
1	1	1	1	0.6887
0.6599	80	121.2270	1	1
0.6599	1	1.5153	1	0.6887
0.6599	1	1.5153	1	1

Here, we've reduced the number of digits reported to clean up the table. The first column gives the name of the model to which the branch lengths are being fitted and the second column ("file") gives the name of the tree file providing the branch-length data. The third column ("negLogL") gives the **negative** log-likelihood of the model after parameter optimization, so lower values indicate better fit. The fourth column ("df") indicates the degrees of freedom of that model, determined by the number of free parameters that are estimated. The fifth column ("dAICc") gives delta AICc values for each model, or the difference in AICc scores between that model and the best-fitting model. The sixth column ("weight") uses these dAICc values to calculate model weights, which sum to 1 across all models.

A note about model fit. Bear in mind that the likelihoods and AIC values reported here indicate the fit of each model to the outside dataset, but better fits to outside data do not necessarily indicate more desirable properties of that distribution as a prior during analysis of the focal data. For instance, the exponential distribution might fit branch lengths from some outside datasets well, but it is often highly informative when employed as a prior and may place too much prior weight on values preferred by that particular outside data set. In such cases, the distributions preferred by different outside data sets may have little or no overlap.

The next five columns ("TL.mean" through "bl.ratio") give estimated or assumed model parameter values for each model. "TL.mean" gives the estimated or assumed total tree length. The compound Dirichlet assumes a mean tree length of 1 by default (or anytime α_T and β_T are 1), but this value varies for those models that estimate β_T. Mean tree length is always implicitly estimated for the gamma and exponential models. Values in the "alphaT" column have different meanings. For the compound Dirichlet, these values are the estimated or assumed values for the α_T parameter of the compound Dirichlet. Due to complications with optimization, EmpPrior-fit never attempts to optimize α_T for the compound Dirichlet models. For the exponential, the value shown corresponds to the number of branch lengths in the input tree. Setting α_T of a compound Dirichlet equal to the number of branch lengths and β_T equal to the rate (λ) of the corresponding exponential reveals an equivalence between the exponential and compound Dirichlet distributions. A compound Dirichlet with these parameter values is equivalent to an exponential with $\lambda = \beta_T$. For the i.i.d. branch-length gamma, alphaT is the number of branches in the tree multiplied by the shape parameter for the gamma branch-length distribution. "betaT" gives the estimated or assumed values of β_T for the compound Dirichlet and gamma distributions, but gives the rate (λ) for the exponential. The "concentration" and "bl.ratio" columns give the assumed or estimated values of α and c, respectively, for the compound Dirichlet. Both are set at 1 for the gamma and exponential models, as these assume a single i.i.d. prior across all branches in the tree.

The estimated values from these tables can easily be used to set informed priors in MrBayes. For the exponential, simply include this line among your MrBayes commands:

```
prset brlenspr = unconstrained:exponential(λ)
```

Substitute the rate of the exponential for λ. To reiterate, this value is given in the column labeled "betaT" for the row "fit.expon".

To set an informed compound Dirichlet prior in MrBayes, use the following command:

prset brlenspr = unconstrained:gammadir(α_T,β_T,α,c)

Substitute informed values of β_T, α, c, or any combination thereof, depending on which informed compound Dirichlet distribution you'd like to use. EmpPrior-fit does not provide informed values of α_T.

Note that the i.i.d. gamma distribution on branch lengths is included for comparative purposes, but it is not an available branch-length prior in MrBayes. It is **not** the same as the compound Dirichlet with a gamma prior on tree length.

References

Brown JM, Hedtke SM, Lemmon AR, and Moriarty Lemmon E. 2010. When trees grow too long: investigating the causes of highly inaccurate Bayesian branch-length estimates. *Syst. Biol.* 59: 145-161.

Gamble T., Berendzen P.B., Shaffer B., Starkey D.E., Simons A.M. 2008. Species limits and phylogeography of North American cricket frogs (Acris: Hylidae). *Mol. Phylogenet. Evol.* 48:112–125.

Marshall D.C. 2010. Cryptic failure of partitioned Bayesian phylogenetic analyses: lost in the land of long trees. *Syst. Biol.* 59:108–117.

Nelson B.J., Andersen J.J., Brown J.M. 2015. Deflating trees: improving Bayesian branch-length estimates using informed priors. *Syst. Biol.* In press.

Rannala B., Zhu T., Yang Z. 2012. Tail paradox, partial identifiability, and influential priors in Bayesian branch length inference. *Mol. Biol. Evol.* 29:325–335.

Ronquist F and Huelsenbeck JP. 2003. MrBayes 3: Bayesian phylogenetic inference under mixed models. *Bioinformatics*. 19: 1572-1574.

Ronquist F., Teslenko M., van der Mark P., Ayres D.L., Darling A., Höhna S., Larget B., Liu L., Suchard M.A., Huelsenbeck J.P. 2012. MrBayes 3.2: efficient Bayesian phylogenetic inference and model choice across a large model space. *Syst. Biol.*, 61:539-542

Yang Z., Rannala B. 2005. Branch-length prior influences Bayesian posterior probability of phylogeny. *Syst. Biol.*, 54:455–470.

Zhang C., Rannala B., Yang Z. 2012. Robustness of compound Dirichlet priors for Bayesian inference of branch lengths. *Syst. Biol.* 61:779–784.

Zwickl D.J. 2006. Genetic algorithm approaches for the phylogenetic analysis of large biological sequence datasets under the maximum likelihood criterion. Ph.D. dissertation, The University of Texas at Austin.