Tutorial using BEAST v2.6.0

Introduction to BEAST2

Jūlija Pečerska, Veronika Bošková and Louis du Plessis
Adapted by Joëlle Barido-Sottani

This is a simple introductory tutorial to help you get started with using BEAST2 and its accomplices.

1 Background

Before diving into performing complex analyses with BEAST2 one needs to understand the basic workflow and concepts. While BEAST2 tries to be as user-friendly as possible, the amount of possibilities can be overwhelming.

In this simple tutorial you will get acquainted with the basic workflow of BEAST2 and the software tools most commonly used to interpret the results of analyses. Bear in mind that this tutorial is designed only to help you get started using BEAST2. This tutorial does not discuss all the choices and concepts in detail, as they are discussed in further tutorials. Interspersed throughout the tutorial are topics for discussion. These discussion topics are optional, however if you work through them you will have a better understanding of the concepts discussed in this tutorial. Feel free to skip the discussion topics and come back to them later, while running the analysis file, or after finishing the whole tutorial.

2 Programs used in this Exercise

2.0.1 BEAST2 - Bayesian Evolutionary Analysis Sampling Trees 2

BEAST2 (http://www.beast2.org) is a free software package for Bayesian evolutionary analysis of molecular sequences using MCMC and strictly oriented toward inference using rooted, time-measured phylogenetic trees. This tutorial is written for BEAST v2.5.0 (Bouckaert et al. 2014; Bouckaert et al. 2019).

2.0.2 BEAUti2 - Bayesian Evolutionary Analysis Utility

BEAUti2 is a graphical user interface tool for generating BEAST2 XML configuration files.

Both BEAST2 and BEAUti2 are Java programs, which means that the exact same code runs on all platforms. For us it simply means that the interface will be the same on all platforms. The screenshots used in this tutorial are taken on a Mac OS X computer; however, both programs will have the same layout and functionality on both Windows and Linux. BEAUti2 is provided as a part of the BEAST2 package so you do not need to install it separately.

2.0.3 TreeAnnotator

TreeAnnotator is used to summarise the posterior sample of trees to produce a maximum clade credibility tree. It can also be used to summarise and visualise the posterior estimates of other tree parameters (e.g. node height).

TreeAnnotator is provided as a part of the BEAST2 package so you do not need to install it separately.

2.0.4 Tracer

Tracer (http://beast.community/tracer) is used to summarise the posterior estimates of the various parameters sampled by the Markov Chain. This program can be used for visual inspection and to assess convergence. It helps to quickly view median estimates and 95% highest posterior density intervals of the parameters, and calculates the effective sample sizes (ESS) of parameters. It can also be used to investigate potential parameter correlations. We will be using Tracer v1.7.0

2.0.5 FigTree

FigTree (http://beast.community/figtree) is a program for viewing trees and producing publication-quality figures. It can interpret the node-annotations created on the summary trees by TreeAnnotator, allowing the user to display node-based statistics (e.g. posterior probabilities). We will be using FigTree v1.4.3.

2.0.6 DensiTree

Bayesian analysis using BEAST2 provides an estimate of the uncertainty in tree space. This distribution is represented by a set of trees, which can be rather large and difficult to interpret. DensiTree is a program for qualitative analysis of sets of trees. DensiTree allows to quickly get an impression of properties of the tree set such as well-supported clades, distribution of tree heights and areas of topological uncertainty.

DensiTree is provided as a part of the BEAST2 package so you do not need to install it separately.

3 Practical: Running a simple analysis with BEAST2

This tutorial will guide you through the analysis of an alignment of sequences of SARS-CoV-2. The aim of this tutorial is to co-estimate the phylogeny and the rate of evolution. More generally, this tutorial aims to introduce new users to a basic workflow and point out the steps towards performing a full analysis of sequencing data within a Bayesian framework using BEAST2.

After completing this tutorial you should be able to:

- Set up all the components of a simple BEAST2 analysis in BEAUti2
- Use Tracer, FigTree and DensiTree to check convergence and analyse results

3.1 Creating the Analysis File with BEAUti

To run analyses with BEAST, one needs to prepare a configuration file in XML format that contains everything BEAST2 needs to run an analysis. A BEAST2 XML file contains:

- The data (typically a sequence alignment)
- The model specification
- Initial values and parameter constraints
- Settings of the MCMC algorithm
- Output options

Even though it is possible to create such files from scratch in a text editor, it can be complicated and is not exactly straightforward. BEAUti is a user-friendly program designed to aid you in producing a valid configuration file for BEAST. If necessary, that file can later be edited by hand, but it is recommended to use BEAUti for generating the files (at least for the initial round of analysis).

Begin by starting **BEAUti2**.

3.1.1 Importing the alignment

To give BEAST2 access to the data, the alignment has to be added to the configuration file.

Drag and drop the file into the open BEAUti window (it should be on the **Partitions** tab) and pick the option **Import alignment** in the dropdown menu.

Alternatively, use **File > Import Alignment** or click on the + in the bottom left-hand corner of the window, then locate and click on the alignment file.

Then select **nucleotide** in the dropdown menu and click **OK**.

Once you have done that, the data should appear in the BEAUti window, which will look as shown in Figure 1.

Since we only have one partition there is nothing more we can do in the **Partitions** panel and proceed to specifying the tip dates.

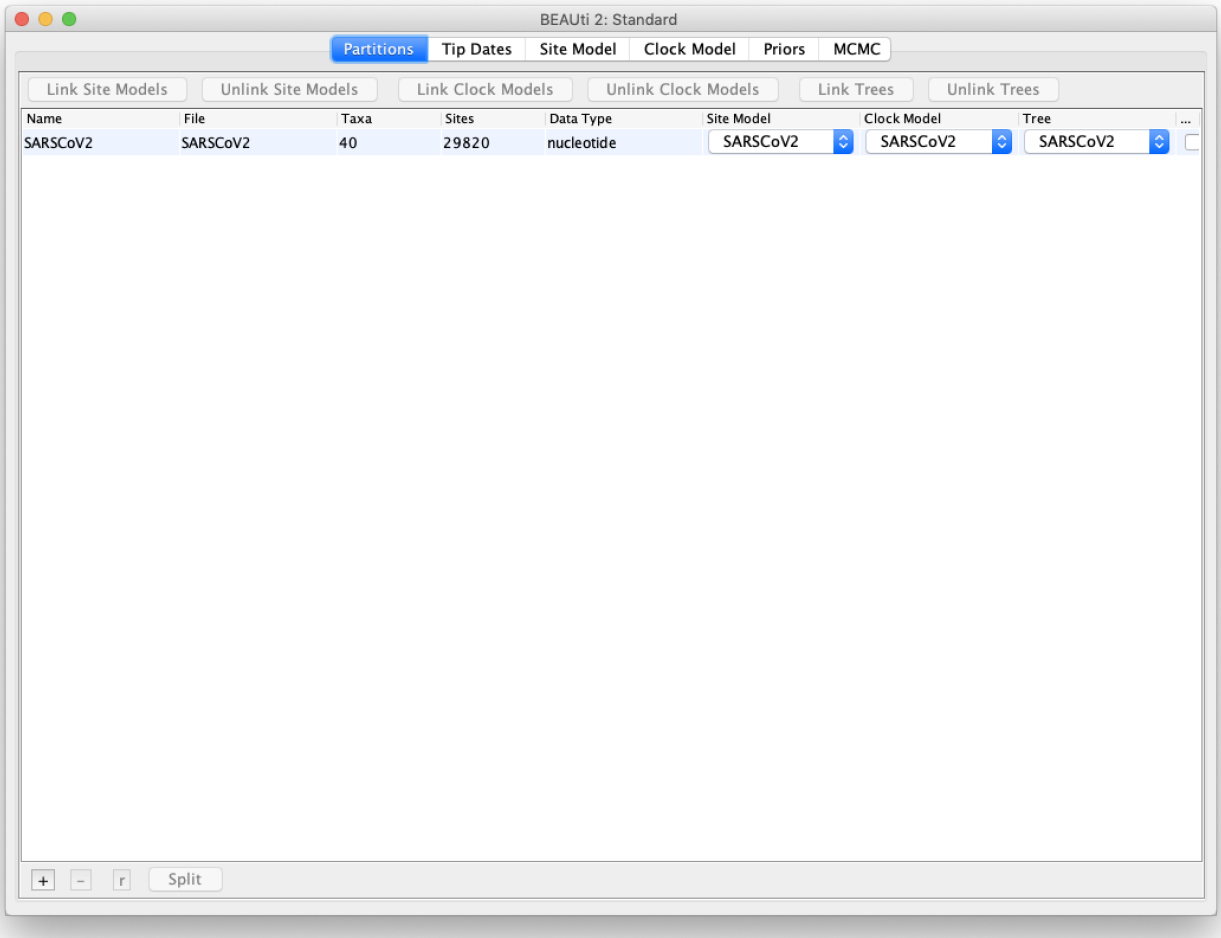

Figure 1: Data imported into BEAUti2.

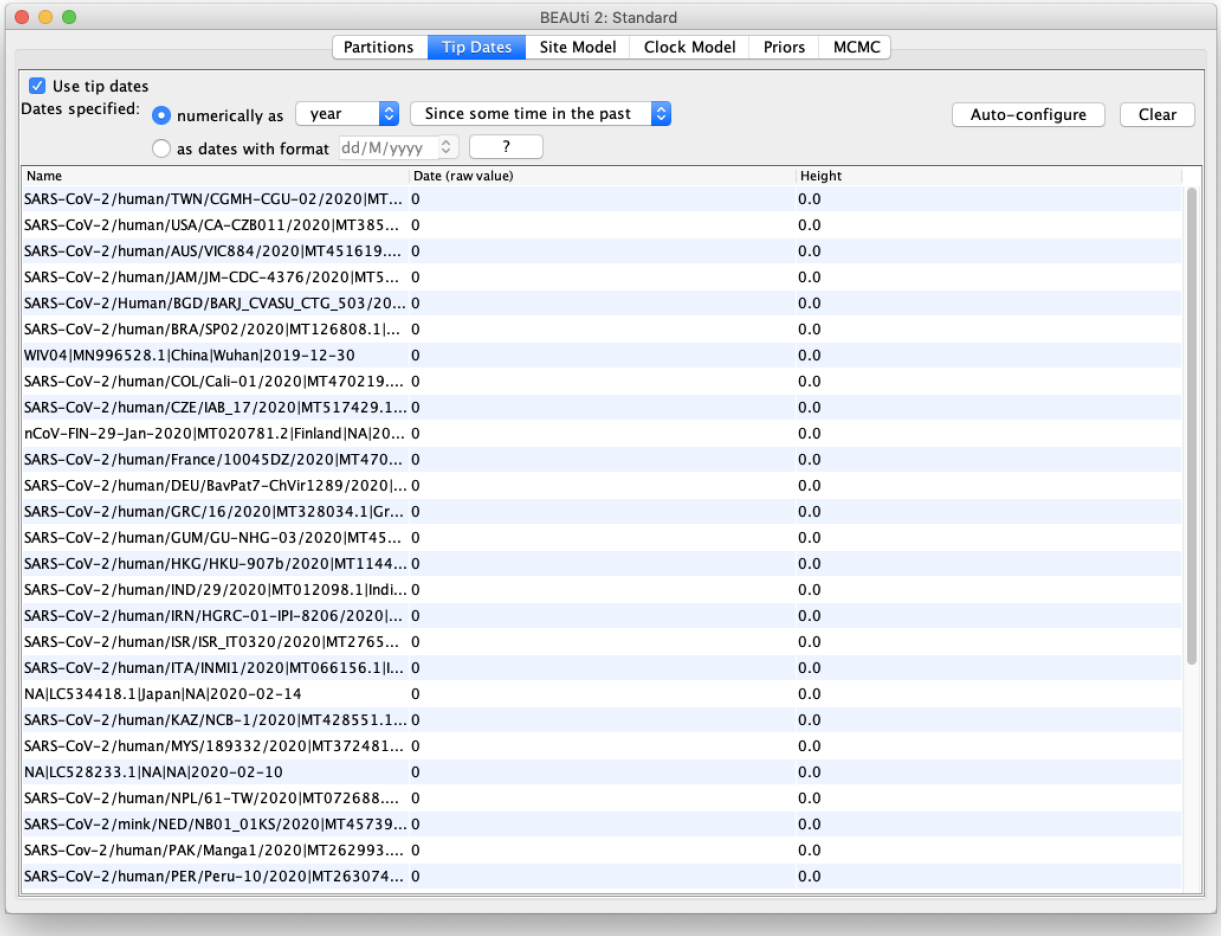

Figure 2: Tip dates panel.

3.1.2 Setting tip dates

The next step is to set the sampling dates for all of our sequences, which is crucial to obtain accurate age estimates on the phylogeny. This is done in the **Tip Dates** tab.

> Select the **Tip Dates** tab.
>
> Select the **Use tip dates** option.

The panel for setting sampling dates will look as shown in Figure 2. By default, all sampling dates are set to $t = 0$, i.e. the present. We *could* input the dates manually, however this is time-consuming, so instead we are going to use the **Auto-configure** option, which allows us to set the dates automatically from the sequence names. The first step is to specify the date format.

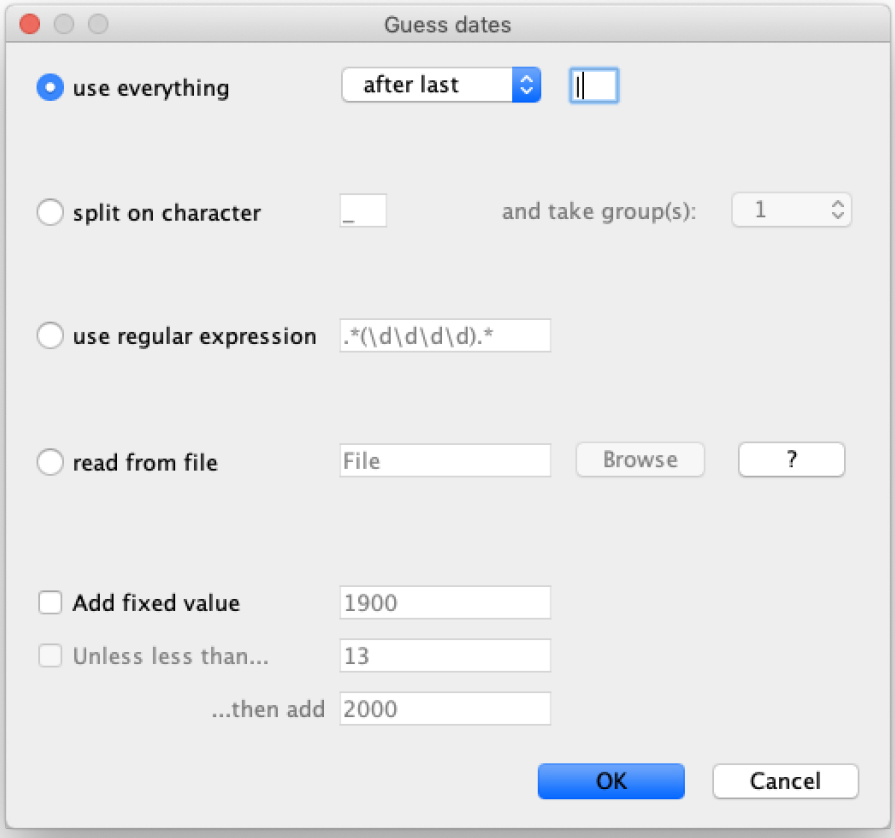

Figure 3: Auto-configure options.

At the top of the panel, select the **as dates with format** option, then select **yyyy-M-dd** as the format in the dropdown menu.

Now we need to tell BEAUti where the date is stored in the sequence name. There are several different options.

Click on the **Auto-configure** button.

In the panel that opens, set the configuration to **use everything**, **after last** and |. The correct configuration is shown in Figure 3.

Click **OK**.

An error message will appear, telling us that not all dates could be configured manually. The problem

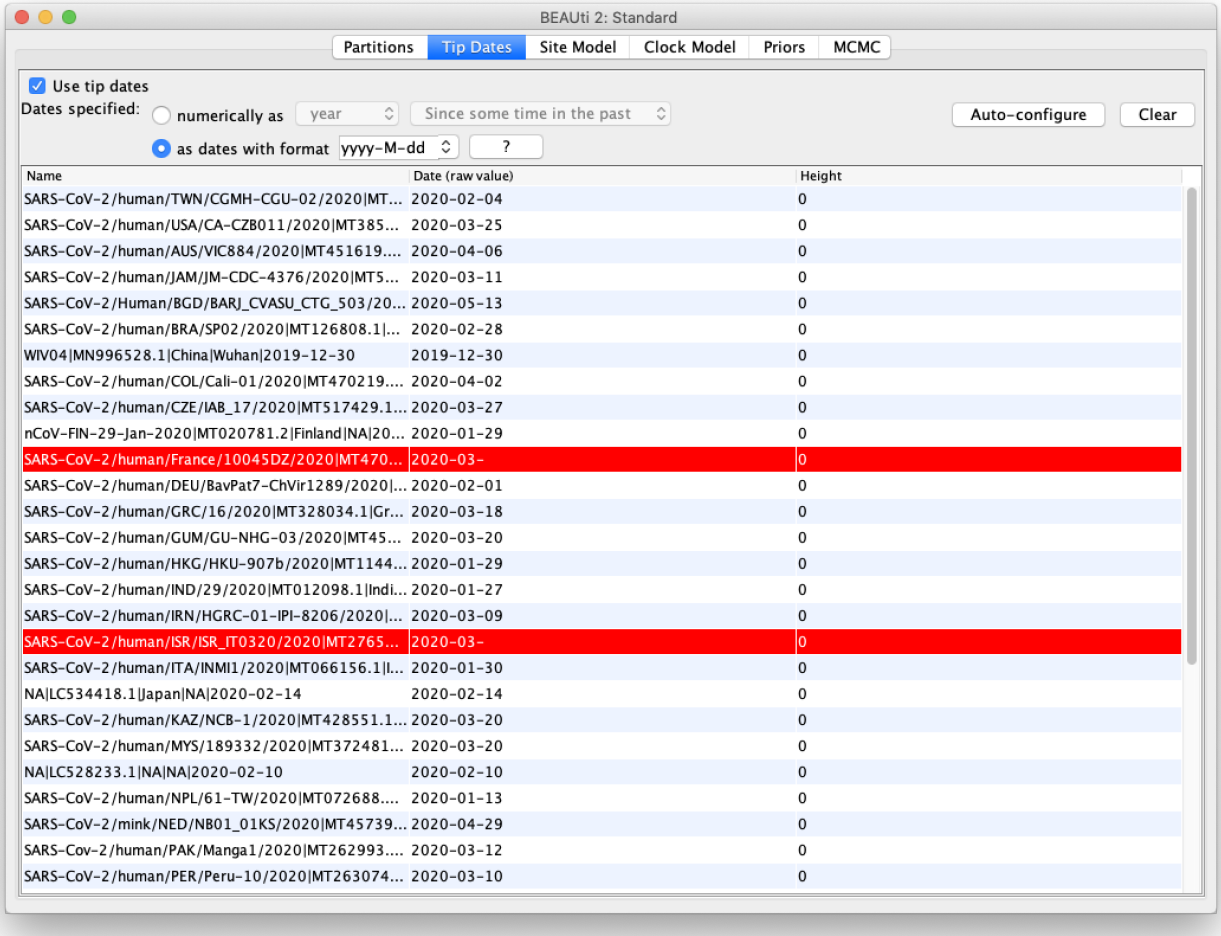

Figure 4: Problem sequences.

sequences are shown in red. As shown in Figure 4, we can see that there are two problem sequences, where the day is missing from the sampling date. We need to edit those sequences manually to set the day, and we choose to set both sequences to have been sampled on the 15th.

Double click on the date of the first problem sequence.

Set the date to **2020-03-15** and press **Enter**. The error message will repeat.

Repear for the second problem sequence.

Now that dates have been set correctly for all sequences, BEAUti will convert them to heights which will be used in the tree. The final result should look like Figure 5.

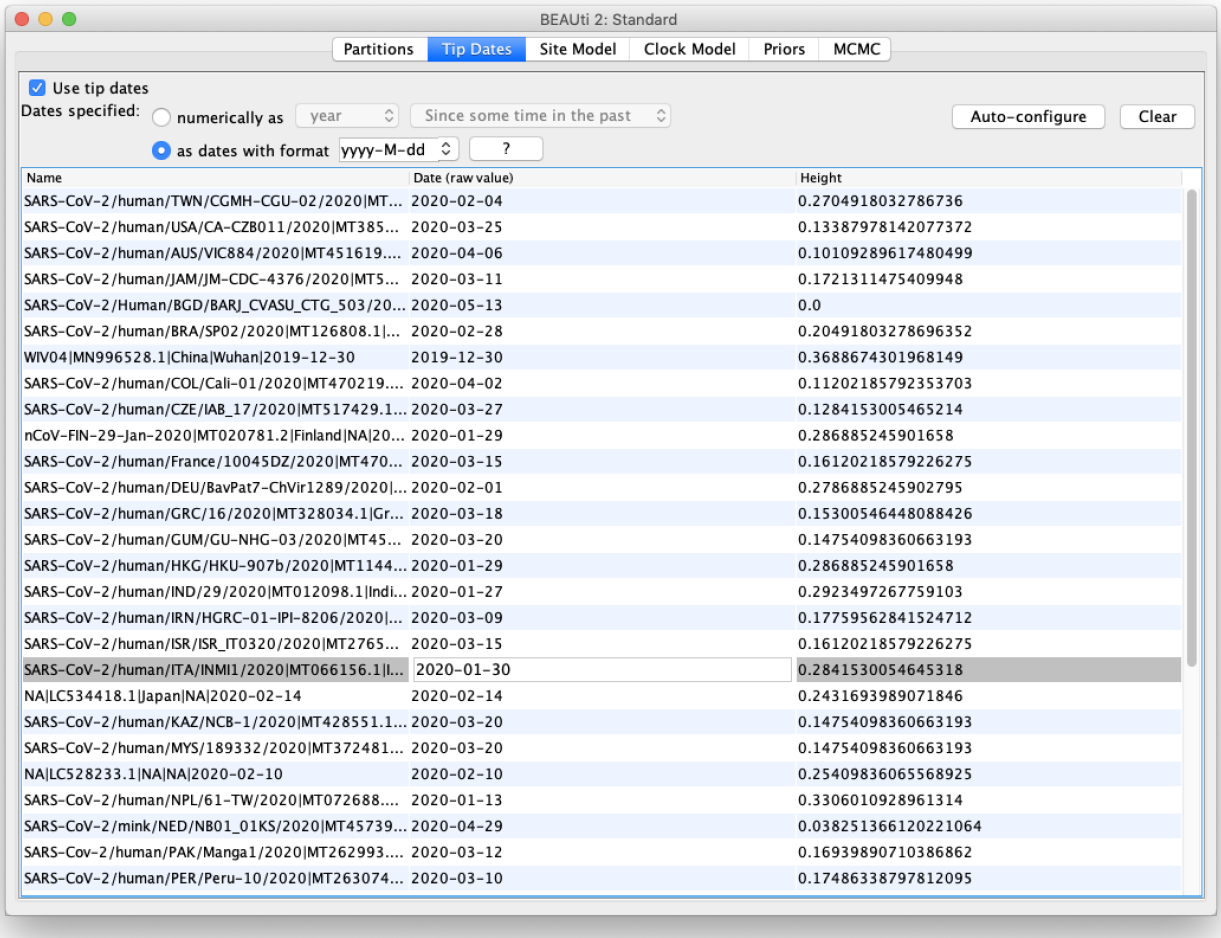

Figure 5: Tip dates panel with final setup.

3.1.3 Setting the substitution model

Next, we need to set up the substitution model in the **Site Model** tab.

Select the **Site Model** tab.

The options available in this panel depend on whether the alignment data are in nucleotides, amino-acids, binary data or general data. The settings available after loading the alignment will contain the default values which we normally want to modify.

Set the **Gamma Category Count** to 4.

Check the **estimate** box for the **Shape** parameter (it should already be checked).

Select **HKY** in the **Subst Model** drop-down menu.

Select **Empirical** from the **Frequencies** drop-down menu.

The panel should look like Figure 6.

We are using an HKY substitution model with empirical frequencies. This will fix the frequencies to the proportions observed in the partition. This approach means that we can get a good fit to the data without explicitly estimating these parameters. To model site-to-site rate variation within each partition we use a discrete Gamma site model with 4 categories.

3.1.4 Setting the clock model

Next, select the **Clock Model** tab at the top of the main window. This is where we set up the molecular clock model. For this exercise we are going to leave the selection at the default value of a strict molecular clock. rate variation among branches to be included in the model.

Click on the **Clock Models** tab and view the setup *(but don't do anything)*.

3.1.5 Setting priors

The **Priors** tab allows prior distributions to be specified for each model parameter. The model selections made in the **Site Model** and **Clock Model** tabs determine which parameters are included in the model. For each of these parameters a prior distribution needs to be specified. It is also possible to specify hyperpriors (and hyper-hyperpriors etc.) for each of the model parameters. We also need to specify a prior for the **Tree**. In this example we will set the tree prior to a Constant Coalescent Population model, which is a simple model appropriate for datasets with serially sampled sequences.

Go to the **Priors** tab and select **Constant Coalescent Population** in the drop-down menu next to **Tree.t:tree**.

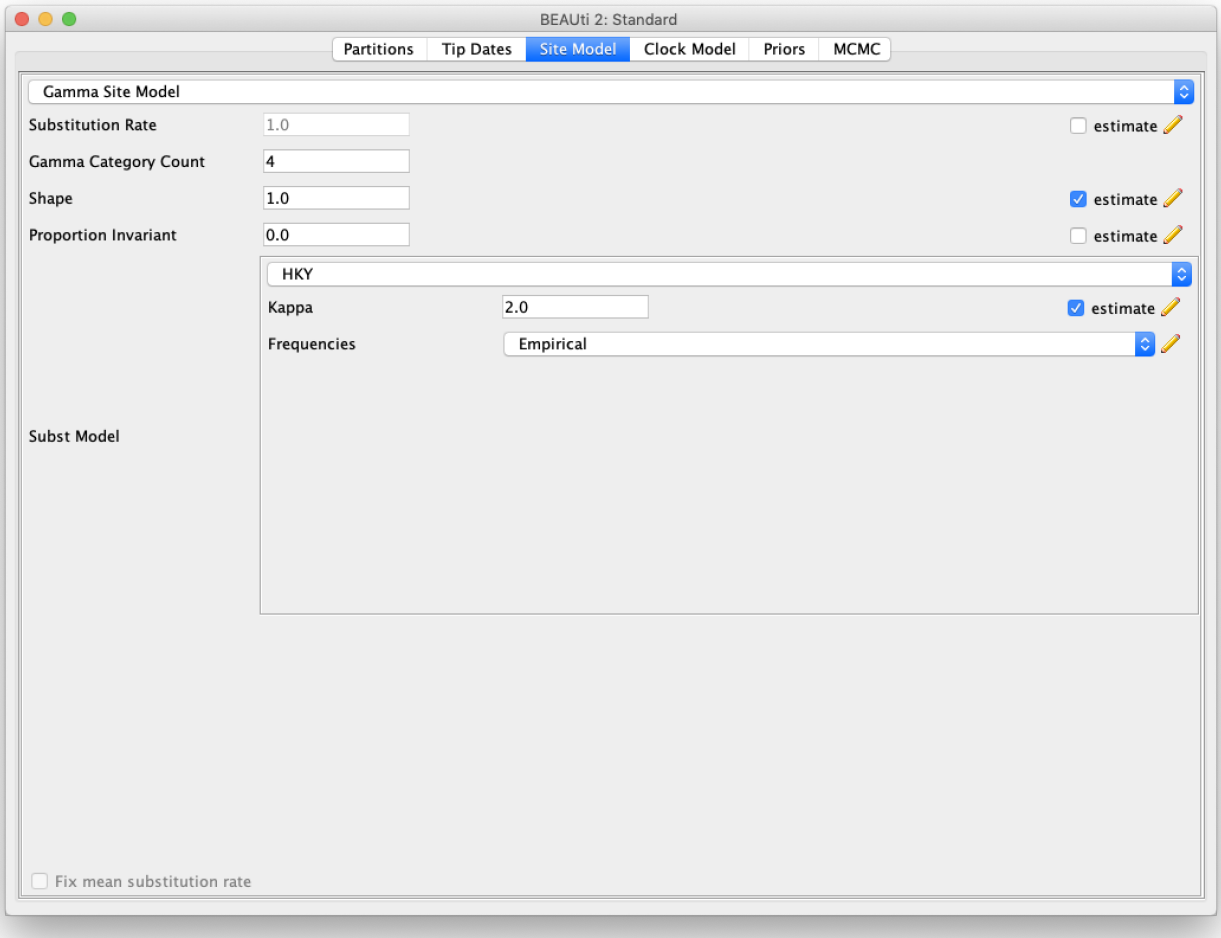

Figure 6: Site model setup.

The **clockRate** parameter measures the overall rate of molecular evolution in our dataset. The default prior contains no information, so we are going to change it to a LogNormal distribution.

For **clockRate.c:SARSCoV2** select **LogNormal** from the drop-down menu

- Expand the options for **clockRate.c:SARSCoV2** using the arrow button on the left.
- Set the **M** (mean) parameter to **0.005** and check the **Mean in real space** option.

Note that BEAUti displays a plot of the prior distribution on the right, as well as a few of its quantiles. This is for easy reference and can help us to decide if a prior is appropriate. The LogNormal prior we are using is defined on $(0, \infty)$ and the values included between its 2.5% and 97.5% quantiles are between 2×10^{-4} and 2×10^{-2}.

If we wanted to add a hyperprior on one of the parameters of the Gamma prior we would check the **estimate** box on the right of the parameter. We could also change the initial values or limits of the model parameters by clicking on the boxes next to the drop-down menus. Do **not** do this here, as we are **not** adding any hyperpriors or changing limits in this analysis!

We will leave the rest of the priors on their default values! The BEAUti panel should look as shown in Figure 7.

Please note that in general using default priors is frowned upon as priors are meant to convey your prior knowledge of the parameters. It is important to know what information the priors add to the MCMC analysis and whether this fits your particular situation. In our case the default priors are suitable for this particular analysis, however for further, more complex analyses, we will require a clear idea of what the priors mean. Getting this understanding is difficult and comes with experience. The topic of choosing priors is discussed in more detail in later tutorials.

3.1.6 Setting the MCMC options

Finally, the **MCMC** tab allows to control the length of the MCMC chain and the frequency of stored samples. It also allows one to change the output file names.

Go to the **MCMC** tab.

The **Chain Length** parameter specifies the number of steps the MCMC chain will make before finishing (i.e. the number of accepted proposals). This number depends on the size of the dataset, the complexity of the model and the precision of the answer required. The default value of 10'000'000 is arbitrary and should be adjusted accordingly. For this small dataset we initially set the chain length to 1'000'000 such that this analysis will take only a few minutes on most modern computers (rather than hours). We leave the **Store Every** and **Pre Burnin** fields at their default values.

Set the **Chain Length** to 1'000'000.

Below these general settings you will find the logging settings. Each particular option can be viewed in

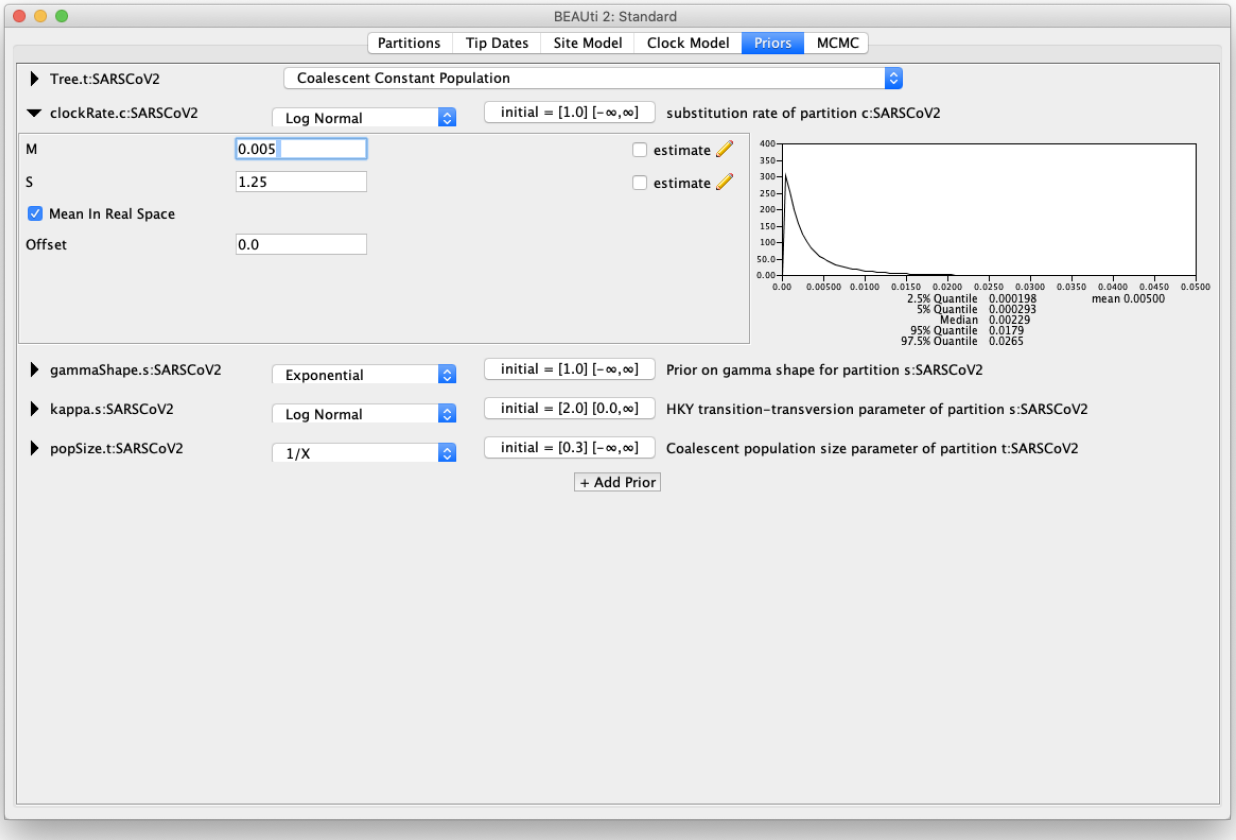

Figure 7: Prior setup.

detail by clicking the arrow to the left of it. You can control the names of the log files and how often values will be stored in each of the files.

Start by expanding the **tracelog** options. This is the log file you will use later to analyse and summarise the results of the run. The **Log Every** parameter for the log file should be set relative to the total length of the chain. Sampling too often will result in very large files with little extra benefit in terms of the accuracy of the analysis. Sampling too sparsely will mean that the log file will not record sufficient information about the distributions of the parameters. We normally want to aim to store no more than 10'000 samples so this should be set to no less than chain length/10'000. For this analysis we will make BEAST2 write to the log file every 200 samples.

Expand the **tracelog** options.

- Set the **Log Every** parameter to **200**.
- Leave the filename as is

Next, expand the **screenlog** options. The screen output is simply for monitoring the program's progress. Since it is not so important, especially if you run your analysis on a remote computer or a computer cluster, the **Log Every** can be set to any value. However, if it is set too small, the sheer quantity of information being displayed on the screen will actually slow the program down. For this analysis we will make BEAST2 log to screen every 1'000 samples, which is the default setting.

Expand the **screenlog** options.

- Leave the **Log Every** parameter at the default value of 1'000.

Finally, we can also change the tree logging frequency by expanding **treelog.t:tree**. For big trees with many taxa each individual tree will already be quite large, thus if you log lots of trees the tree files can easily become extremely large. You will be amazed at how quickly BEAST can fill up even the biggest of drives if the tree logging frequency is too high! For this reason it is often a good idea to set the tree logging frequency lower than the trace log (especially for analyses with many taxa). However, be careful, as the post-processing steps of some models (such as the Bayesian skyline plot) require the trace and tree logging frequencies to be identical!

Expand the **treelog.t:tree** options.

- Set the **File Name** to `SARSCoV2.trees`.
- Leave the **Log Every** parameter at the default value of 1'000.

The final setup should look as in Figure 8.

3.1.7 Generating the XML file

We are now ready to create the BEAST2 XML file. This is the final configuration file BEAST2 can use to execute the analysis.

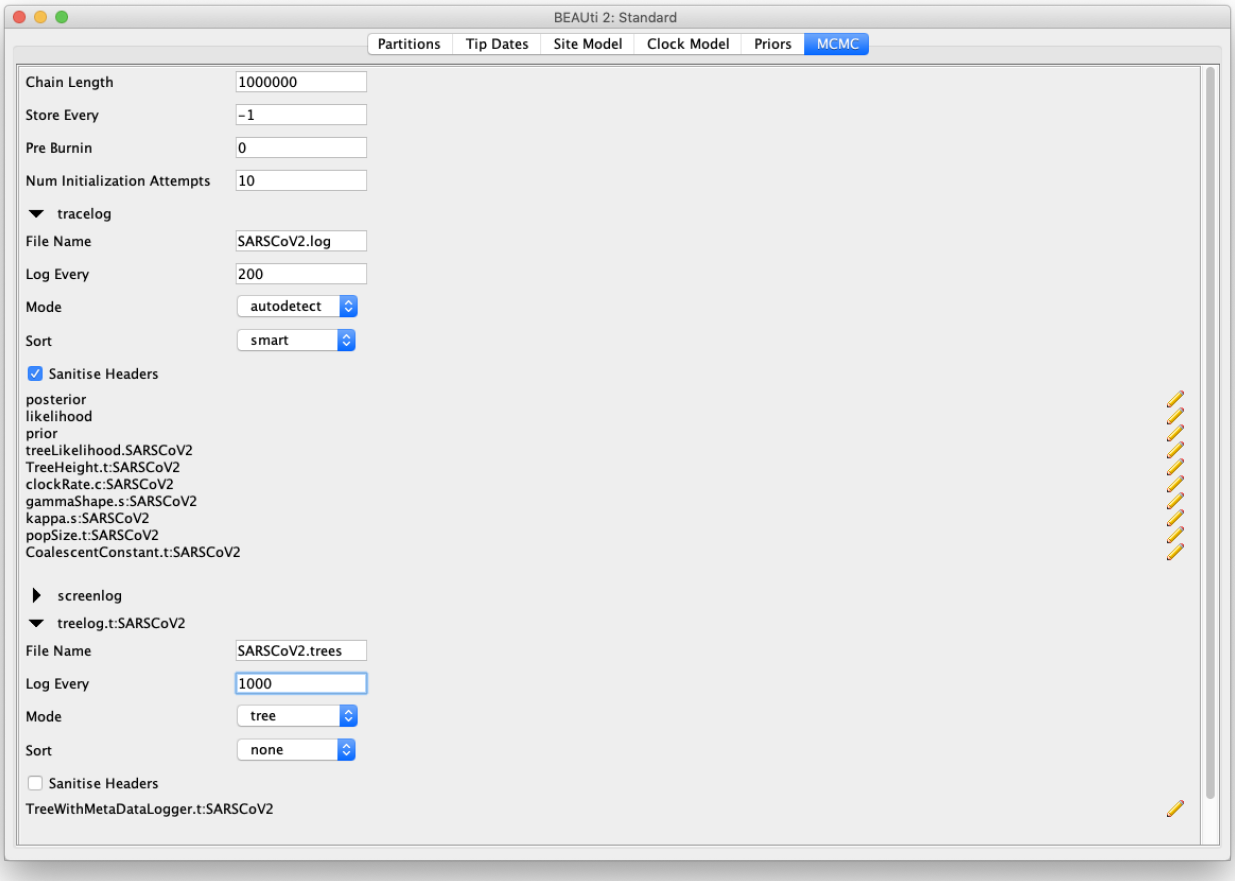

Figure 8: Logging options.

Save the XML file under the name `SARSCoV2.xml` using **File > Save**.

3.2 Running the analysis

Now run BEAST2 and provide your newly created XML file as input. You can also change the **random number seed** for the run. This number is the starting point of a pseudo-random number chain BEAST2 will use to generate the samples. As computers are unable to generate truly random numbers, we have to resort to generating determinate sequences of numbers that only look random, but will be identical when the starting seed is the same. If your MCMC run converges to the true posterior then you will be able to draw the same conclusions regardless of which random seed is provided. However, if you want to exactly reproduce the results of a run you need to start it with the same random number seed.

Run the **BEAST2** program.

- Select `SARSCoV2.xml` as the **Beast XML File**.
- Set the **Random number seed** to **777** (or pick your favourite number).
- Check the **Use BEAGLE library if available** checkbox. If you have previously installed BEAGLE this will make the analysis run faster.

The BEAST2 window should look as shown in Figure 9.

Run **BEAST2** by clicking the `Run` button.

BEAST2 will run until the specified number of steps in the chain is reached. While it is running, it will print the screenlog values to a console and store the tracelog and tree log values to files located in the same folder as the configuration XML file. The screen output will look approximately as shown in Figure 10.

The window will remain open when BEAST2 finished running the analysis. When you try to close it, you may see BEAST2 asking the question: "Do you wish to save?". Note that your log and trees files are always saved, no matter what answer you choose for this question. Thus, the question is only restricted to saving the BEAST2 screen output (which contains some information about the hardware configuration, initial values, operator acceptance rates and running time that are not stored in the other output files).

Topic for discussion: While the analysis is running see if you can identify which parts of the setup in BEAUti are concerned with the data, the model and the MCMC algorithm.

Open the XML file in your favourite text editor. Can you recognize any of the values you set in BEAUti? Can you identify the data, model specification and MCMC settings in the XML file?

Can you find the likelihood, priors and hyperpriors in the XML file?

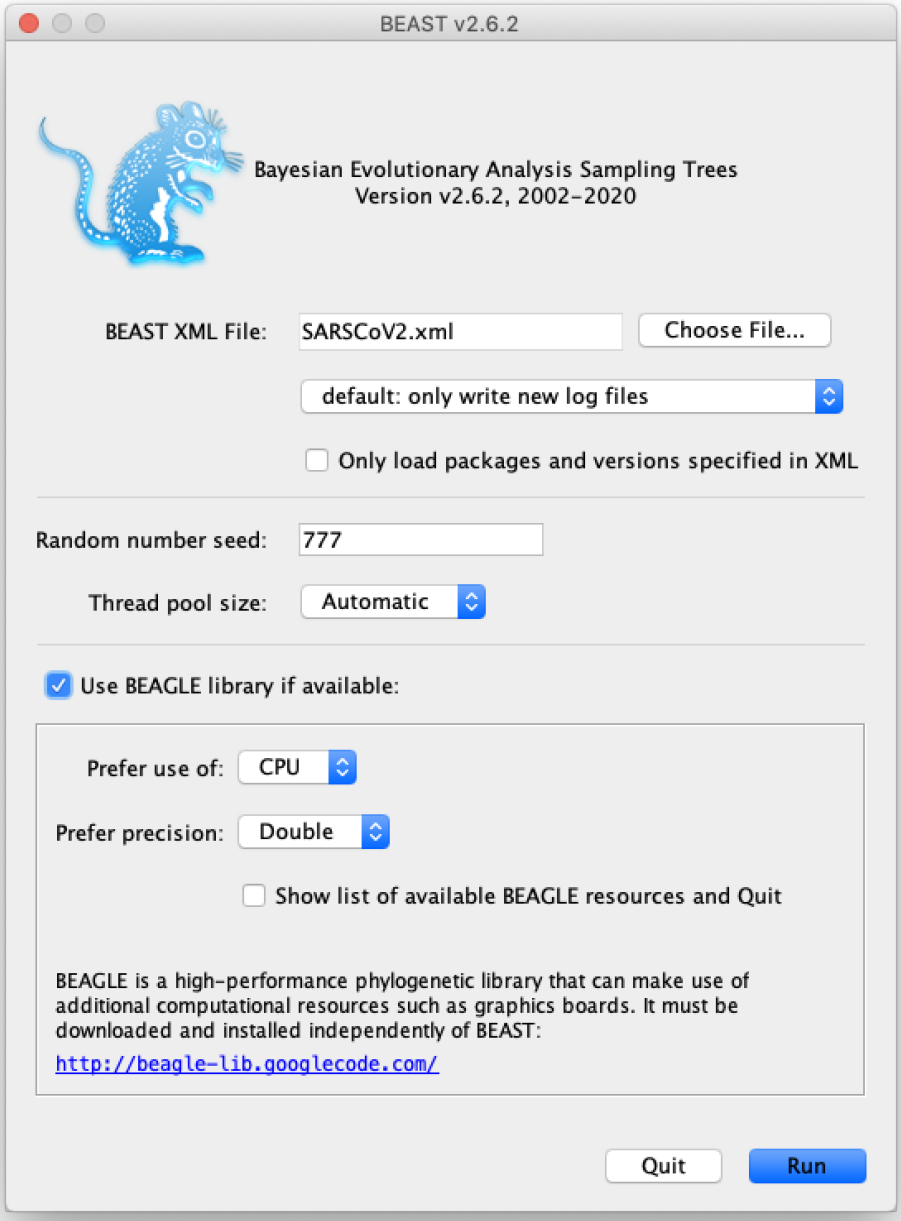

Figure 9: BEAST2 setup for the analysis.

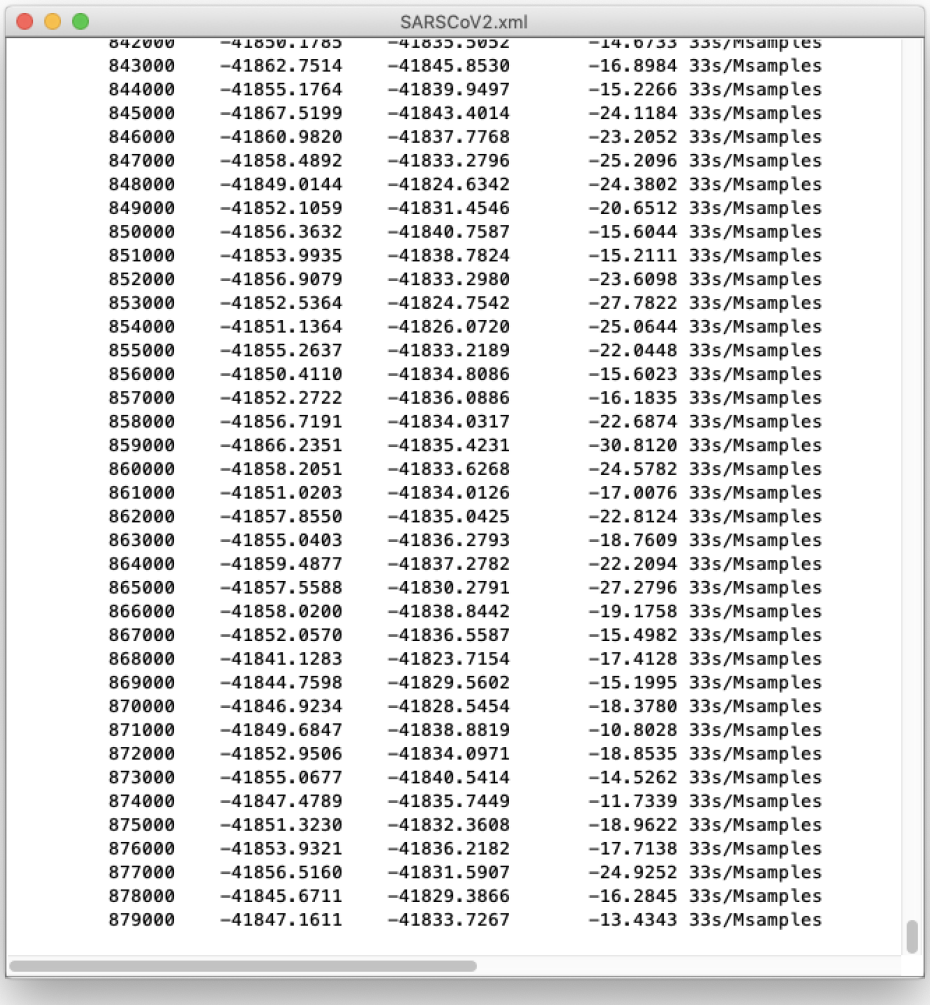

Figure 10: BEAST2 screen output for the analysis.

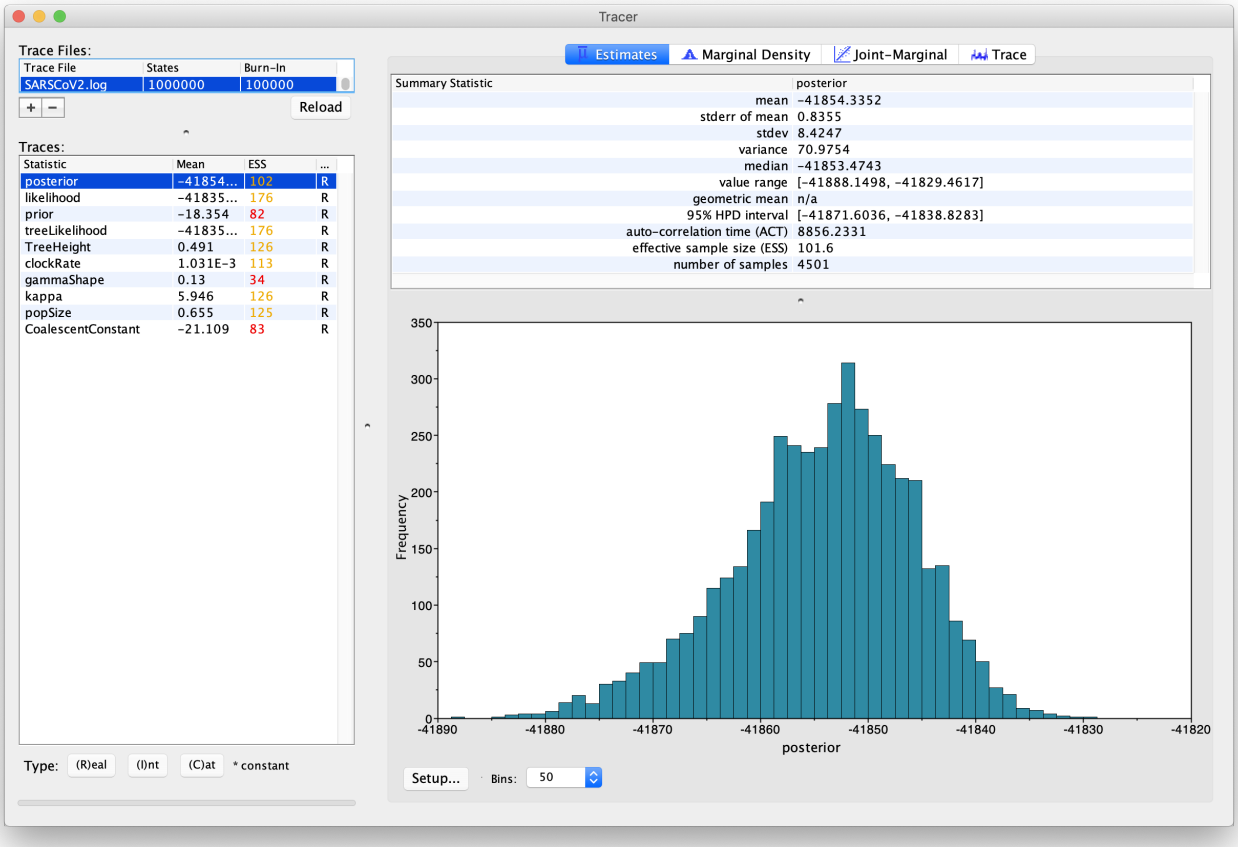

Figure 11: Tracer showing a summary of the BEAST2 run of covid data with MCMC chain length of 1'000'000.

3.3 Analysing the results

Once BEAST2 has finished running, open Tracer to get an overview of BEAST2 output. When the main window has opened, choose `File` > `Import Trace File...` and select the file called `SARSCoV2.log` that BEAST2 has created, or simply drag the file from the file manager window into Tracer.

> Open **Tracer**. Drag and drop the `SARSCoV2.log` file that BEAST2 created into the open Tracer window.
>
> Alternatively, use **File > Import Trace File...** (or press the + button below the **Trace Files** panel) then locate and click on `SARSCoV2.log`.

The Tracer window should look as shown in Figure 11.

Tracer provides a few useful summary statistics on the results of the analysis. On the left side in the top window it provides a list of log files loaded into the program at the moment. The window below shows the list of statistics logged in each file. For each statistic it gives a list of summary values such as the mean, standard error, median, and others it can compute from the data. The summary values are displayed in the

top right window and a histogrom showing the distribution of the statistic is in the bottom right window.

The log file contains traces for the posterior (this is the natural logarithm of the product of the tree likelihood and the prior density), prior, the likelihood, tree likelihoods and other continuous parameters. Selecting a trace on the left brings up the summary statistics for this trace on the right hand side. When first opened, the **posterior** trace is selected and various statistics of this trace are shown under the **Estimates** tab.

For each loaded log file we can specify a **Burn-In**, which is shown in the file list table (top left) in Tracer. The burn-in is intended to give the Markov Chain time to reach its equilibrium distribution, particularly if it has started from a bad starting point. A bad starting point may lead to over-sampling regions of the posterior that actually have very low probability under the equilibrium distribution, before the chain settles into the equilibrium distribution. Burn-in allows us to simply discard the first N samples of a chain and not use them to compute the summary statistics. Determining the number of samples to discard is not a trivial problem and depends on the size of the dataset, the complexity of the model and the length of the chain. A good rule of thumb is to always throw out at least the first 10% of the whole chain length as the burn-in (however, in some cases it may be necessary to discard as much as 50% of the MCMC chain).

Select the **TreeHeight** statistic in the left hand list to look at the tree height estimated jointly for all partitions in the alignment. Tracer plots the (marginal posterior) histogram for the selected statistic and also give you summary statistics such as the mean and median. The 95% HPD stands for *highest posterior density interval* and represents the most compact interval on the selected statistic that contains 95% of the posterior density. It can be loosely thought of as a Bayesian analogue to a confidence interval. The **TreeHeight** statistic gives the marginal posterior distribution of the age of the root of the entire tree (that is, the tMRCA).

Select **TreeHeight** in the bottom left hand list in Tracer and view the different summary statistics on the right.

3.3.1 Analysing the posterior estimate quality

Two very important summary statistics that we should pay attention to are the Auto-Correlation Time (ACT) and the Effective Sample Size (ESS). ACT is the average number of states in the MCMC chain that two samples have to be separated by for them to be uncorrelated, i.e. for them to be independent samples from the posterior. The ACT is estimated from the samples in the trace (excluding the burn-in). The ESS is the number of independent samples that the trace is equivalent to. This is calculated as the chain length (excluding the burn-in) divided by the ACT.

The ESS is in general regarded as a quality-measure of the resulting sample sequence. It is unclear how to determine exactly how large should the ESS be for the analysis to be trustworthy. In general, an ESS of 200 is considered high enough to make the analysis useful. However, this is an arbitrary number and you should always use your own judgment to decide if the analysis has converged or not. As you can see in Figure 11, ESS values below 100 are coloured in red, which means that we should not trust the value of the statistics, and ESS values between 100 and 200 are coloured in yellow.

If a lot of statistics have red or yellow coloured ESS value, we have not sufficiently explored the posterior space. This is most likely a result of the chain not running long enough. Try running the same analysis as before, but with a longer chain.

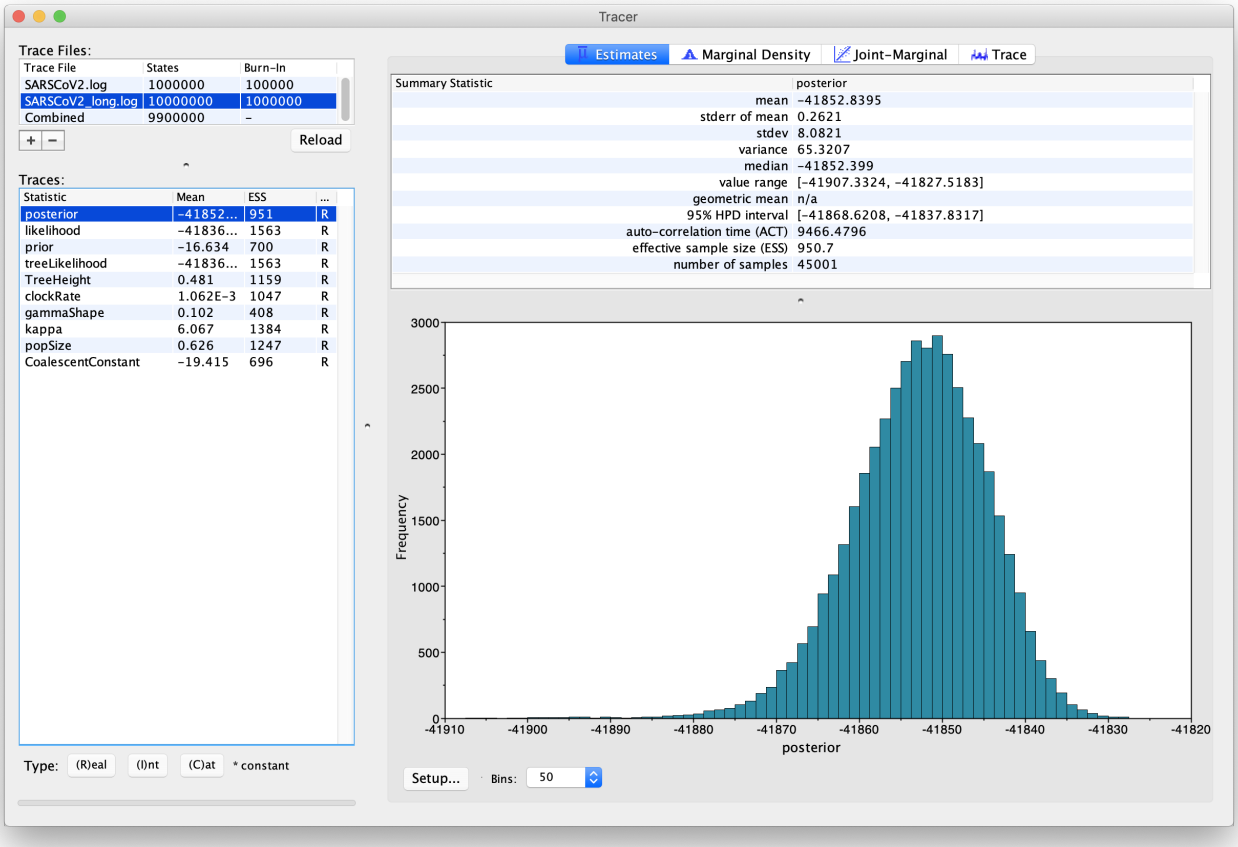

Figure 12: Tracer showing a summary of the BEAST2 run with MCMC chain length of 10'000'000.

First load the XML configuration file into BEAUti again by pressing **File > Load** and select the `SARSCoV2.xml` file. Within BEAUti, change the MCMC chain length parameter to **10'000'000** and change the **tracelog** frequency to **1'000**.

Change the trace and tree log file names in order to not overwrite the results of the previous analysis. You may add something like `_long` behind the name of the file, to obtain `SARSCoV2_long.log` for the log file and `SARSCoV2_long.trees` for trees file.

Run BEAST2 again with the updated configuration file and the same seed of **777**.

This will take a bit more time. Figure 12 shows the estimates from a longer run. In this case all parameters have ESS values larger than 200. Remember that MCMC is a stochastic algorithm, so if you set a different seed the actual numbers will not be exactly the same as those depicted in the figure.

Tracer also allows us to look for correlations between parameters under the **Joint Marginal** tab, as shown in Figure 13. When two parameters are highly correlated this can lead to poor convergence of the MCMC chain (more on this in later tutorials).

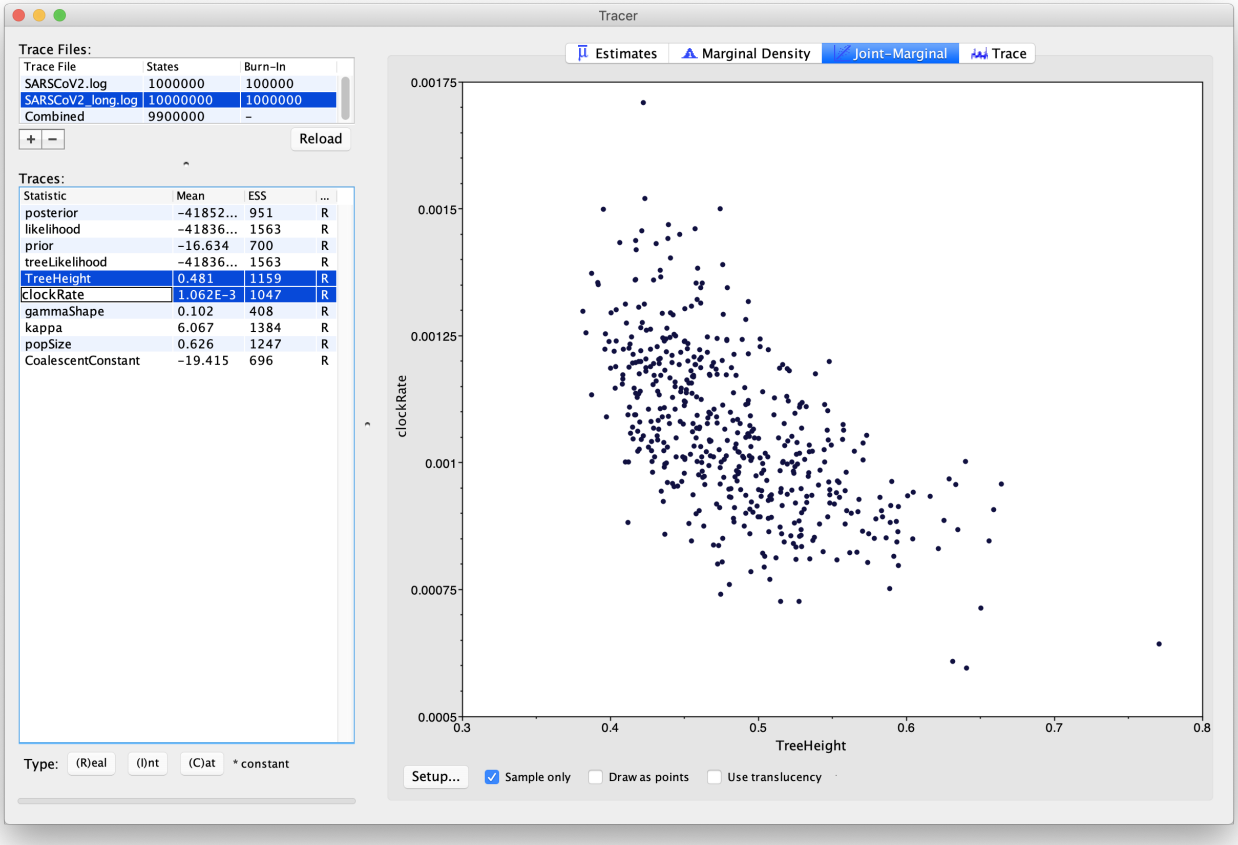

Figure 13: Correlation between the tree height and clock rate estimates.

Topic for discussion: We have not explored the **Trace** tab in Tracer at all!

The **Trace** tab is primarily a diagnostic tool for checking convergence to the posterior, assessing the length of the burn-in and whether or not the chain is mixing well. There is a good argument to be made for this being the *most important* tab in the Tracer program and that it is the first tab users should look at.

Have a look at the individual parameter traces in the **Trace** tab, in both the short and long log files. Can you figure out why ESS values for some parameters are higher than others?

Do you think a burn-in of 10% is sufficient for this analysis?

3.3.2 Analysing tree estimates

Besides producing a sample of parameter estimates, BEAST2 also produces a posterior sample of phylogenetic time-trees. These need to be summarised too before any conclusions about the quality of the posterior estimate can be made.

One way to summarise the trees is by using the program TreeAnnotator. This will take the set of trees and find the best supported tree. It will then annotate this representative summary tree with the mean ages of all the nodes and the corresponding 95% HPD ranges. It will also calculate the posterior clade probability for each node. Such a tree is called the *maximum clade credibility* tree.

Open **TreeAnnotator**.

Set the **Burnin percentage** to **10%** to discard the first 10% of trees in the log file.

The next option, the **Posterior probability limit**, specifies a limit such that if a node is found at less than this frequency in the sample of trees (i.e. has a posterior probability less than this limit), it will not be annotated. For example, setting it to 0.5 means that only nodes seen in the majority (more than 50%) of trees will be annotated. The default value is 0, which we will leave as is, and which means that TreeAnnotator will annotate all nodes.

Leave the **Posterior probability limit** at the default value of **0**.

For the **Target tree type** option you can either choose a specific tree from a file or ask TreeAnnotator to find a tree in your sample. The default option which we will leave, **Maximum clade credibility tree**, finds the tree with the highest product of the posterior probability of all its nodes.

Leave the **Target tree type** at the default value of **Maximum clade credibility tree**.

Next, select **Mean heights** for the **Node heights**. This sets the heights (ages) of each node in the tree to the mean height across the entire sample of trees for that clade.

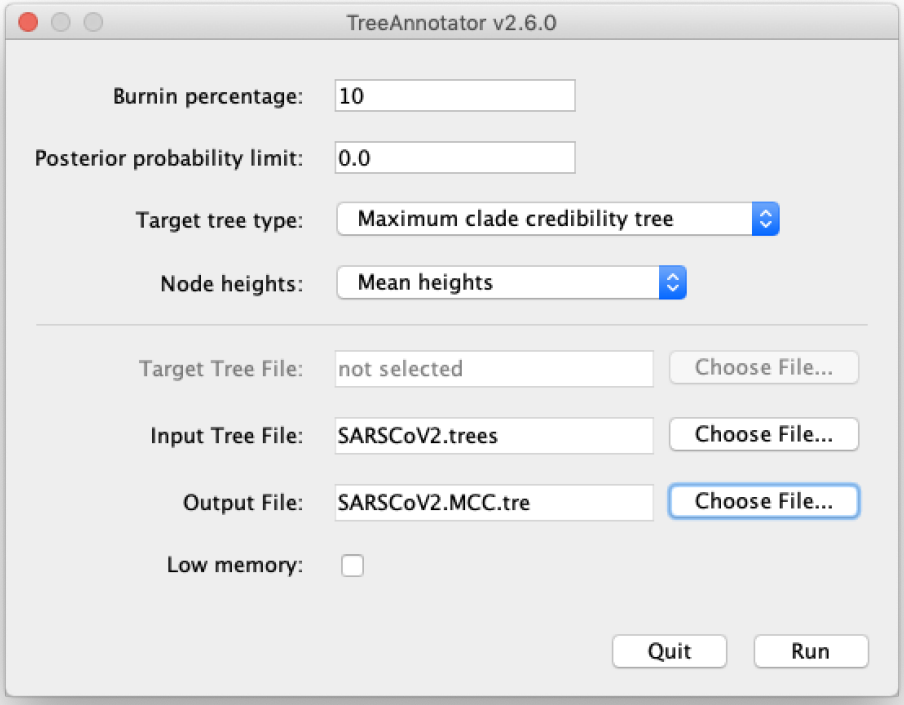

Figure 14: TreeAnnotator setup

Select **Mean heights** in the **Node heights** drop-down menu.

Finally, we have to select the input tree log file and set an output file.

Click **Choose File** next to **Input Tree File** and choose `SARSCoV2.trees`.

Set the **Output File** to `SARSCoV2.MCC.tree`.

The setup should look as shown in Figure 14. You can now run the program.

3.3.3 Visualising the tree estimate

Finally, we can visualize the tree with one of the available pieces of software, such as FigTree.

Open **FigTree**. Use **File** > **Open** then locate and click on `SARSCoV2.MCC.tree`.

- Expand **Trees** options, check **Order nodes** and select **decreasing** from the drop-down menu.

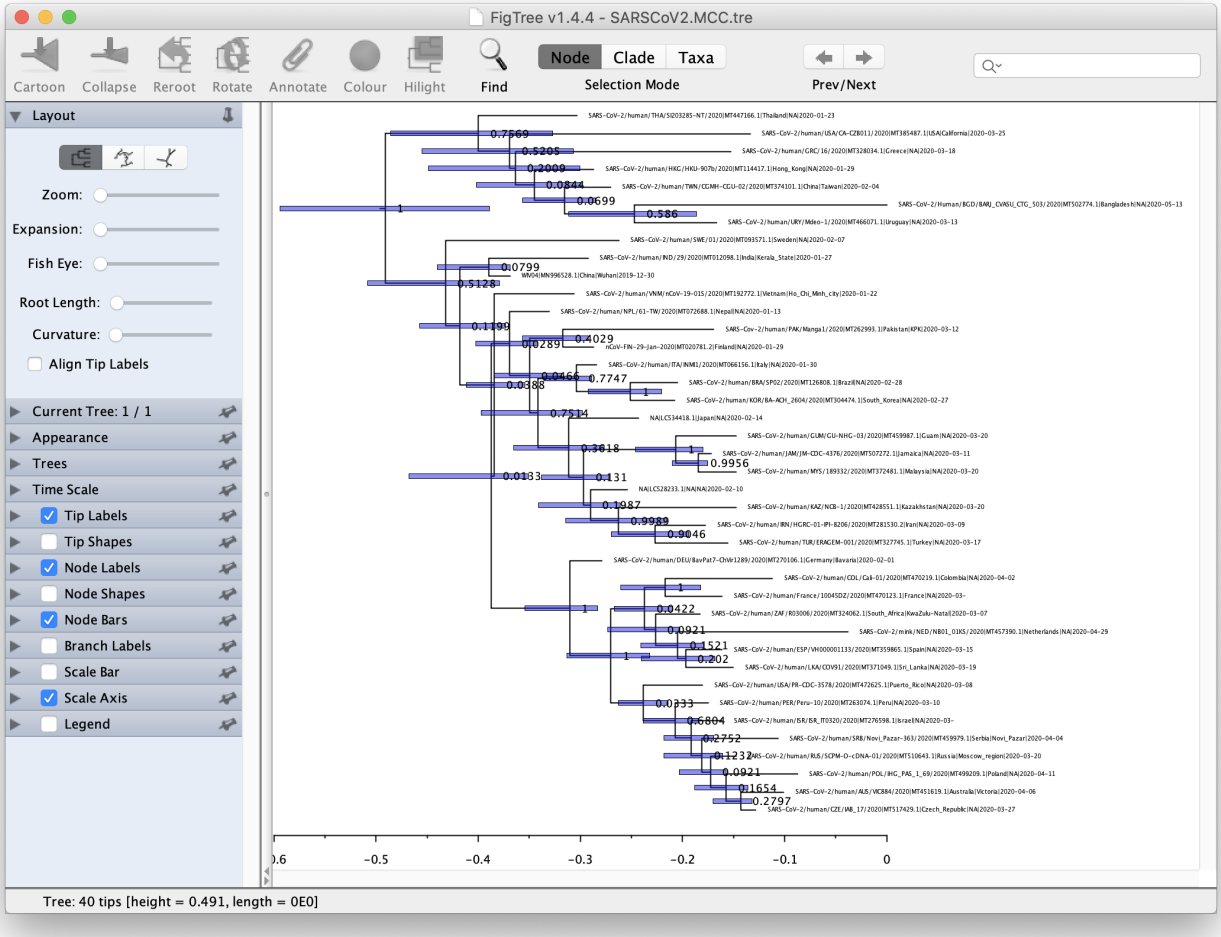

Figure 15: FigTree visualisation of the estimated tree.

- Expand the **Tip Labels** options and decrease the **Font Size** so the tree is visible.
- Check the **Node Bars** checkbox, expand the options and select `height_95%_HPD` from the **Display** drop-down menu.
- Check the **Node Labels** checkbox, expand the options and select `posterior` from the **Display** drop-down menu.
- Uncheck the **Scale Bar** checkbox.
- Check the **Scale Axis** checkbox, expand the options, check **Reverse axis** and increase the **Font Size**.

Your tree should now look something like Figure 15. We first ordered the tree nodes. Because there are many ways to draw the same tree ordering nodes makes it easier for us to compare different trees to each other. The scale bars we added represent the 95% HPD interval for the age of each node in the tree, as estimated by the BEAST2 analysis. The node labels we added gives the posterior probability for a node in the posterior set of trees (that is, the trees logged in the tree log file, after discarding the burn-in). We

can also use FigTree to display other statistics, such as the branch lengths, the 95% HPD interval of a node etc. The exact statistics available will depend on the model used.

Topics for discussion: The posterior probabilities tell us which clades are highly supported and the scale bars tell us how confident we are about their divergence times.

- Are all clades well-supported? How about their ages?
- What is the estimated origin time of this tree? Note that the scale is in units of year.

4 Acknowledgments

The content of this tutorial is based on the Divergence Dating Tutorial with BEAST 2.0 tutorial by Drummond, Rambaut, and Bouckaert.

5 Useful Links

- Bayesian Evolutionary Analysis with BEAST 2 (Drummond and Bouckaert 2014)
- BEAST 2 website and documentation: `http://www.beast2.org/`
- BEAST 1 website and documentation: http://beast.bio.ed.ac.uk
- Join the BEAST user discussion: `http://groups.google.com/group/beast-users`

Version dated: July 16, 2020

Relevant References

Bouckaert, R, J Heled, D Kühnert, T Vaughan, CH Wu, D Xie, MA Suchard, A Rambaut, and AJ Drummond. 2014. Beast 2: a software platform for bayesian evolutionary analysis. *PLoS computational biology* 10: e1003537.

Bouckaert, R et al. 2019. Beast 2.5: an advanced software platform for bayesian evolutionary analysis. *PLOS Computational Biology* 15:

Drummond, AJ and RR Bouckaert. 2014. *Bayesian evolutionary analysis with BEAST 2*. Cambridge University Press,