πBUSS: BEAST/BEAGLE Utility for Sequence Simulation

Authors

- Filip Bielejec (filip.bielejec(sorry_spybots)rega.kuleuven.be)
- Philippe Lemey (philippe.lemey(sorry_spybots)rega.kuleuven.be)
- Luiz Max Carvalho (luizepidemiologia(sorry_spybots)gmail.com)
- Guy Baele (guy.baele(sorry_spybots)rega.kuleuven.be)
- Andrew Rambaut (a.rambaut(sorry_spybots)ed.ac.uk)
- Marc Suchard (msuchard(sorry_spybots)ucla.edu)

Download

Compiled, runnable program can be downloaded from: https://rega.kuleuven.be/cev/ecv/software/pibuss.

Table of Contents

1 Introduction

The BEAST/BEAGLE utility for sequence simulation (πBUSS) provides an easy to use an interface that allows for flexible and extensible phylogenetic data fabrication, delegating computationally intensive tasks to the BEAGLE library and thus making full use of multi-core architectures. This document is intended as a guide through πBUSS. The objective is to show the unexperienced user how to simulate character sequences (nucleotides, amino acids) under a number of Markov models of evolution (HKY, GTR, TN93, GY94). Moreover, we show how the user can set several partitions, with linked or unlinked parameters. All this can be achieved by using the graphical user interface (GUI). πBUSS also has a command-line interface (CLI), that allows for easy integration and pipelining, which may come in handy for big simulation studies. To gain access to the more advanced options and simulate under complex evolutionary scenarios, users are encouraged to generate and edit XML files that can be then loaded into the BEAST software package. All three interfaces will be presented in this tutorial.

πBUSS is cross-platform and will run on Windows, Mac and Linux operating systems. It is implemented in java, and requires Java runtime environment version 1.5 or greater to run its executables. Also, the BEAGLE library needs to be installed. See https://code.google.com/p/beagle-lib/w/list for details on BEAGLE installation and use.

Windows users can run πBUSS by opening a CMD and issuing:

```
 java -Djava.library.path="C:\Program Files (x86)\Common Files\ libhmsbeagle-1.0"
-jar pibuss.jar
```

Note that this assumes BEAGLE is installed at the default location. If this is not the case, the user should supply the appropriate path for Djava.library.path. Also, mind the quotation marks, they are important.
In a UNIX/Linux enviroment, the πBUSS GUI can be called from the command line with:

```
 java -Djava.library.path=/usr/local/lib -jar pibuss.jar
```

where -Djava.library.path again points to the directory where BEAGLE is installed.

Loading a tree topology

In order to simulate character sequences, one will need a tree topology, with correspondent branch lengths, as backbone. In πBUSS, to import a tree, the user needs simply to go the **Trees** tab and click the ‘Choose file’ button. After that, a window will open for you to browse though your files and go the desired tree file (see Figure 1). πBUSS accepts input trees in newick or nexus formats.

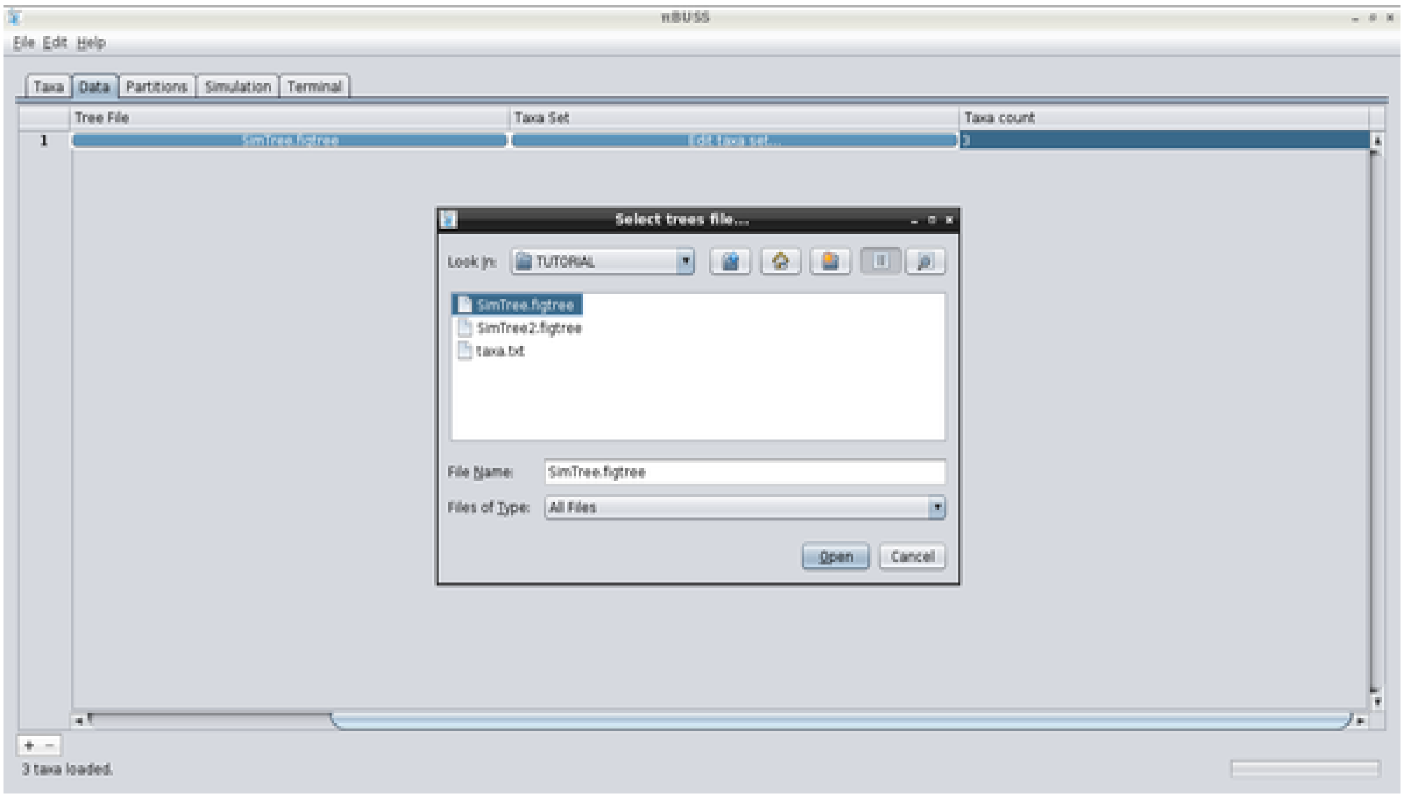


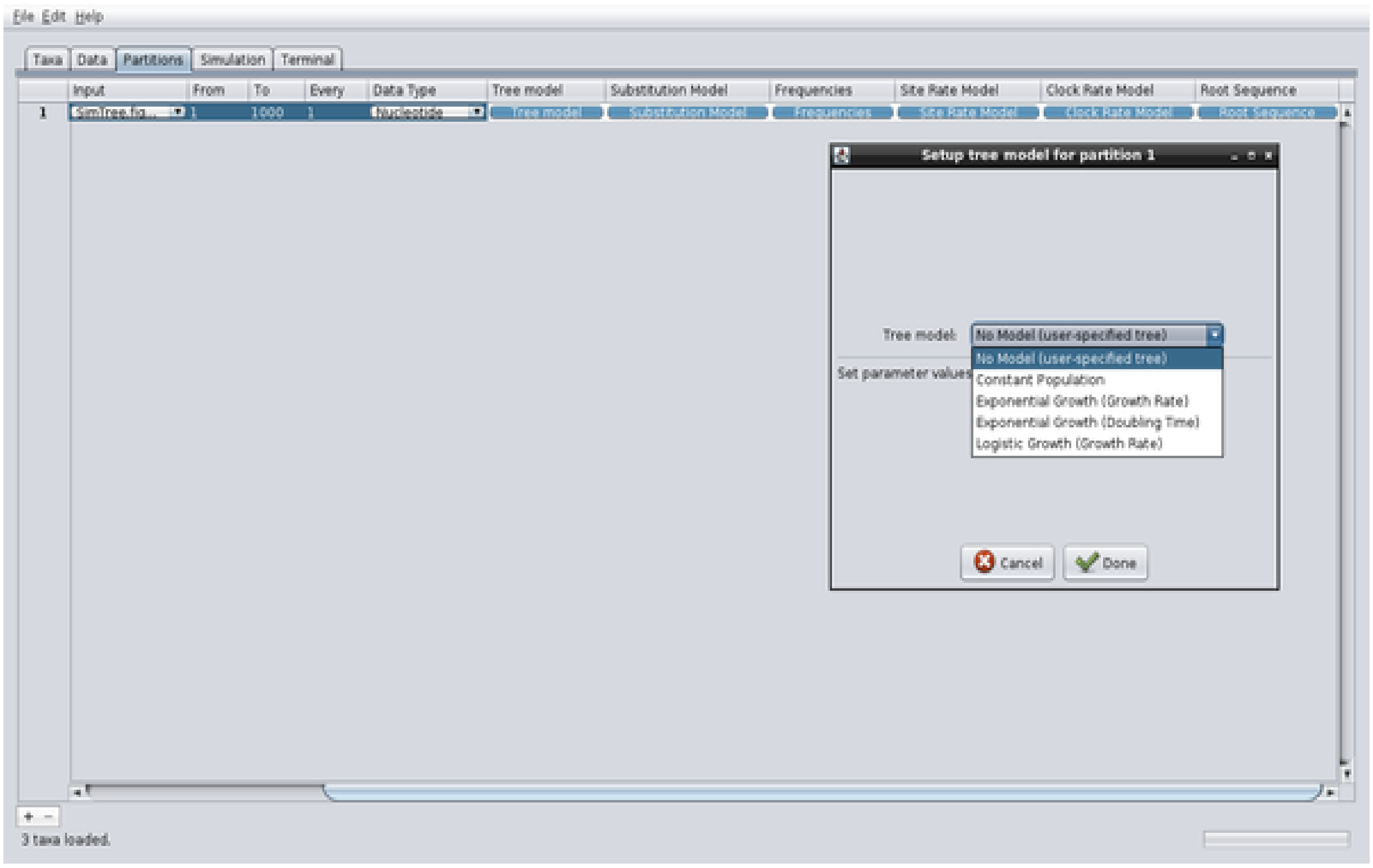
Figure 1. How to load a tree into πBUSS.

After the tree topology is loaded, the user can check taxa names and branch lengths (tree heights) at the **Taxa** tab.

Setting a coalescent model

The next step after loading a tree is to set a demographic coalescent model for the simulation. The default is to use the user-specified tree. But sometimes one may be interested in simulating several demographic scenarios for a given set of taxa, and πBUSS offers three such options. Currently, the user can choose between the Constant population and Exponential growth models, either by specifying growth rates or doubling times.

To select a demographic model for the simulation, the user needs to go to the **Partitions** tab and click the *Demographic model* button. Then one can choose between the three models and set the parameters. For the *Constant population* option, all that is needed is to set the population size parameter. For *Exponential Growth*, the user needs to set both the population size parameter and a growth parameter, that can be either expressed as a growth rate or as doubling population times. Figure 2 illustrates this step.


File Edit Help
Taxa
Data
Partitions
Simulation
Terminal
Setup tree model for partition 1
No Model (user-specified tree)
Constant Population
Exponential Growth (Growth Rate)
Exponential Growth (Doubling Time)
Logistic Growth (Growth Rate)
Cancel
Done
+ −

	Input	From	To	Every	Data Type	Tree model	Substitution Model	Frequencies	Site Rate Model	Clock Rate Model	Root Sequence
1	SimTree.fig...	1	1000	1	Nucleotide	Tree model	Substitution Model	Frequencies	Site Rate Model	Clock Rate Model	Root Sequence

Tree model: No Model (user-specified tree)

Set parameter values

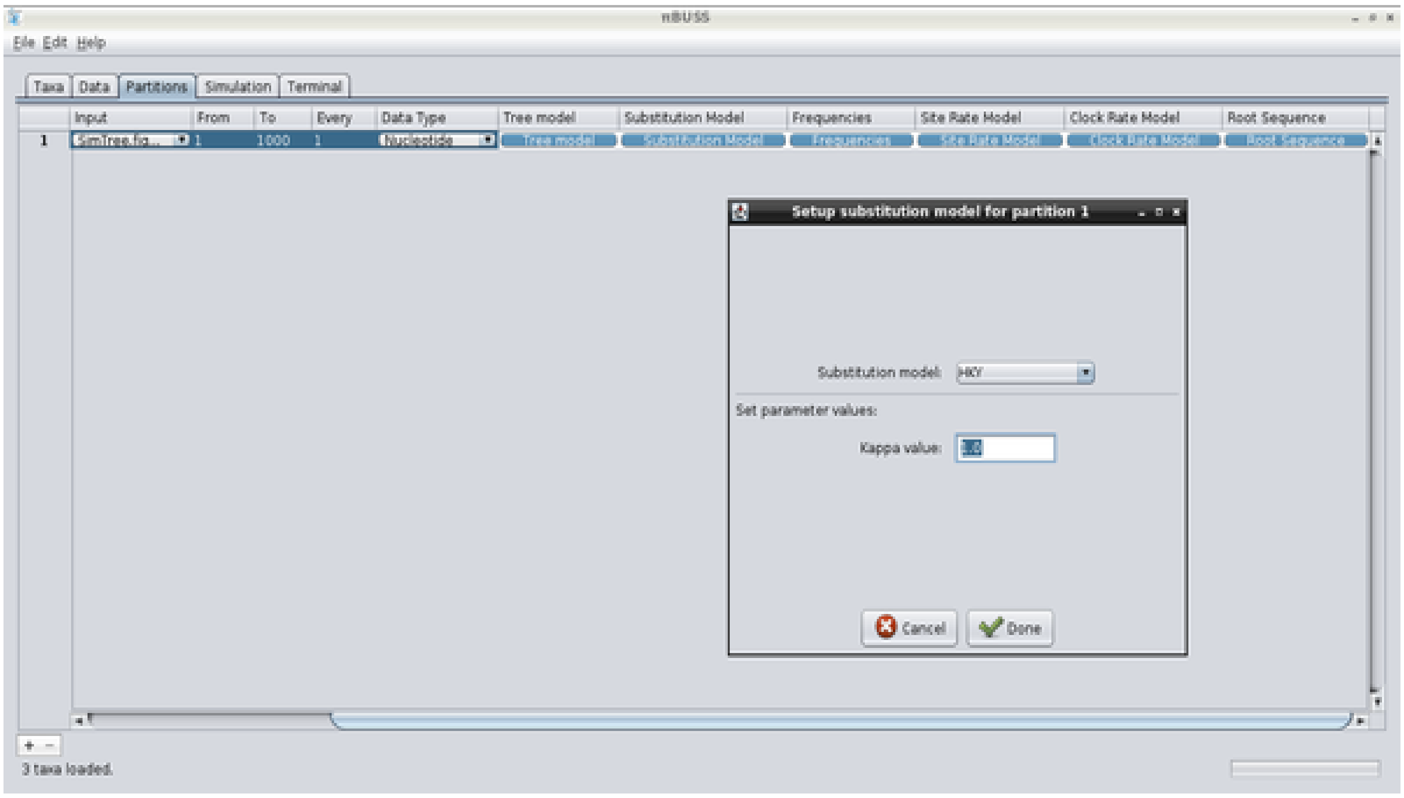
3 taxa loaded.

Figure 2. Choosing a demographic (coalescent) model

Branch substitution model

After setting the demographic model, it is time to choose a branch substitution model, i.e., a Markov model of evolution. πBUSS offers three models of nucleotide sequence evolution (HKY, TN93, GTR) as well as a codon model (GY94). Simulating data under models of different complexity is crucial to assess the accuracy of reconstruction methods and their robustness to model misspecification and assumption violation. Also, simulating data under a codon model allows generating data with a known ratio of non-synonymous to synonymous mutations (dN/dS), which can in turn be used to test methods that try to estimate this quantity and/or detect selection. To choose a substitution model, go the **Partitions** tab, click the ‘Branch Substitution Model’ and select the desired model. For each model, certain parameters have to be specified. For the HKY model, just the kappa parameter needs to be set. For the TN93 model, the user needs to supply two kappa values, and for the GTR model, six base-transition parameters can be specified.

πBUSS also offers the ability to simulate amino acid sequences under several widely used empirical models, such as the BLOSUM62, CPREV, Dayhoff, JTT, FLU, MTREV, WAG and LG. You can also specify your own amino acid (base) frequencies by using the 'Base Frequencies' dropdown menu.



	Input	From	To	Every	Data Type	Tree model	Substitution Model	Frequencies	Site Rate Model	Clock Rate Model	Root Sequence
1	SimTree.fig...	1	1000	1	Nucleotide	Tree model	Substitution Model	Frequencies	Site Rate Model	Clock Rate Model	Root Sequence

Setup substitution model for partition 1

Substitution model: HKY

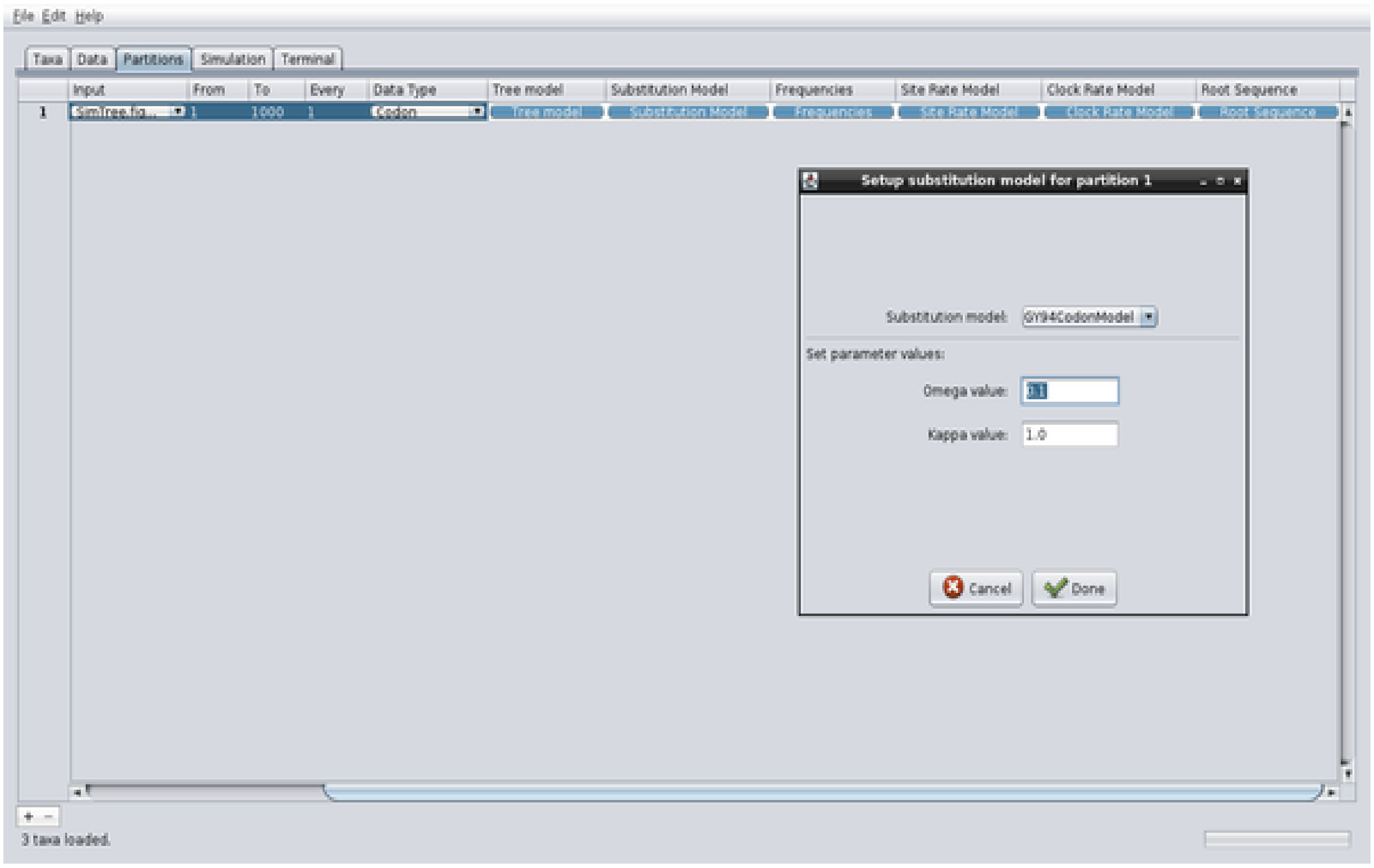
Set parameter values:

Kappa value: 1.0

3 taxa loaded.

Figure 3. Setting a HKY substitution model with kappa parameter of 1.0.

The GY94 codon model takes two parameters, omega and kappa. While omega controls the ratio of non-synonymous and synonymous substitution rates (dN/dS), kappa controls, as usual, the transition/transversion ratio. See Figures 3 and 4 for examples.



	Input	From	To	Every	Data Type	Tree model	Substitution Model	Frequencies	Site Rate Model	Clock Rate Model	Root Sequence
1	SimTree.fig...	1	1000	1	Codon	Tree model	Substitution Model	Frequencies	Site Rate Model	Clock Rate Model	Root Sequence

Setup substitution model for partition 1

Substitution model: GY94CodonModel

Set parameter values:

Omega value: 0.1

Kappa value: 1.0

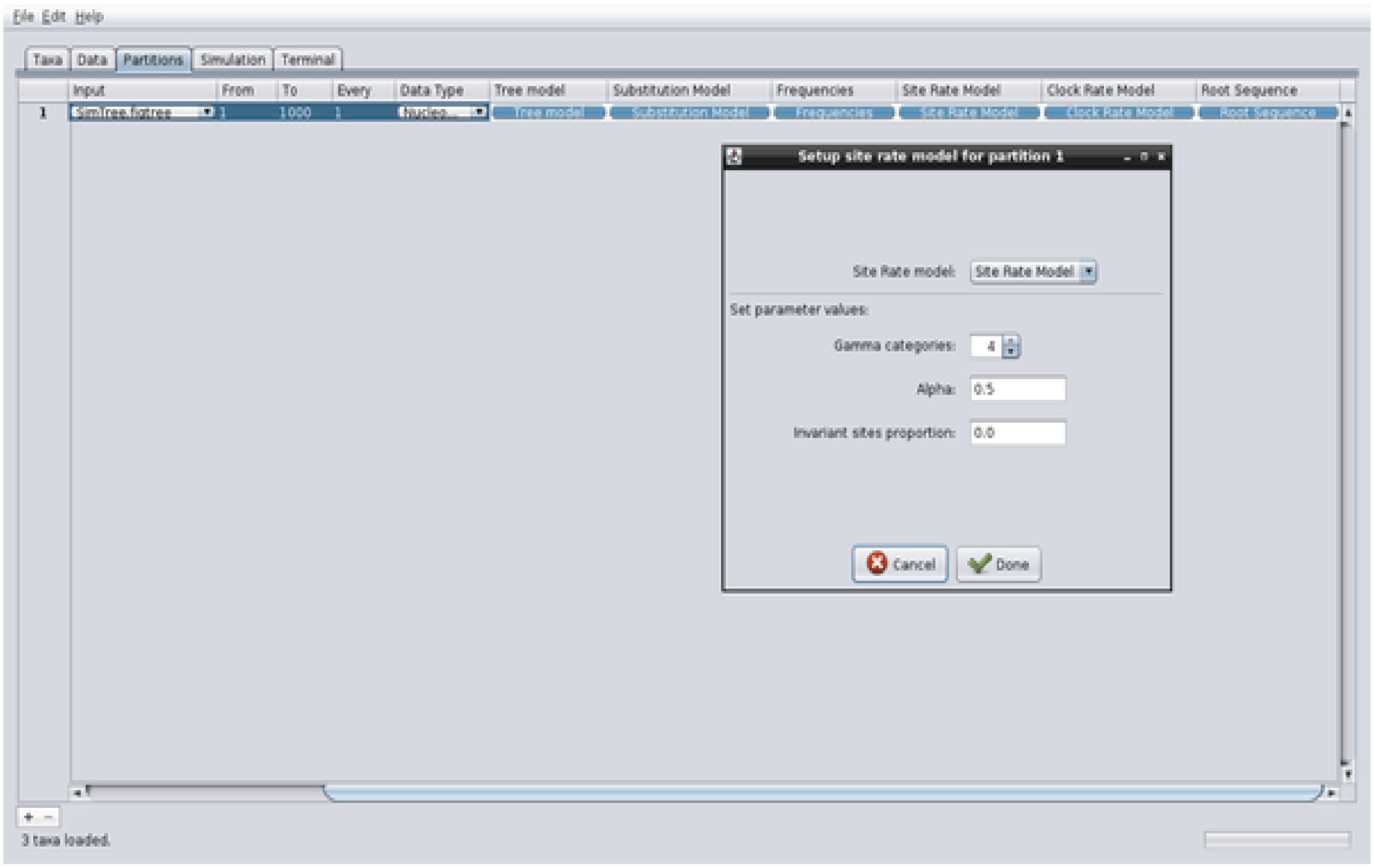
3 taxa loaded.

Figure 4. Setting a codon model (GY94) with kappa = 1.0 and omega = 0.1.

Site rate heterogeneity and base frequencies

Classical models of evolution assume that mutations are independent and identically distributed along a sequence. In the real world, however, this may not be case. Site rate heterogeneity is commonly modeled by assuming that mutation rates are distributed according to a discretized Gamma distribution. The distribution is governed by two parameters: alpha and the number of rate categories. The alpha parameter controls how much heterogeneity is there between sites and the smaller the value of alpha, the more dissimilar the rates between sites, with few sites having high rates and the rest being practically invariant.

Additionally, it may be appropriate to assume that a given proportion of the sites do not mutate at all, i.e., they are invariant. In πBUSS, the user can add site rate heterogeinity to the simulated data by going to the **Partitions** tab and clicking the 'Site Rate Model' button (Figure 5). There the user can specify how many categories should be created for the discretized Gamma distribution, as well as set the proportion of invariant sites.



	Input	From	To	Every	Data Type	Tree model	Substitution Model	Frequencies	Site Rate Model	Clock Rate Model	Root Sequence
1	SimTree.figtree	1	1000	1	Nucleo...	Tree model	Substitution Model	Frequencies	Site Rate Model	Clock Rate Model	Root Sequence

Setup site rate model for partition 1

Site Rate model: Site Rate Model

Set parameter values:

Gamma categories: 4

Alpha: 0.5

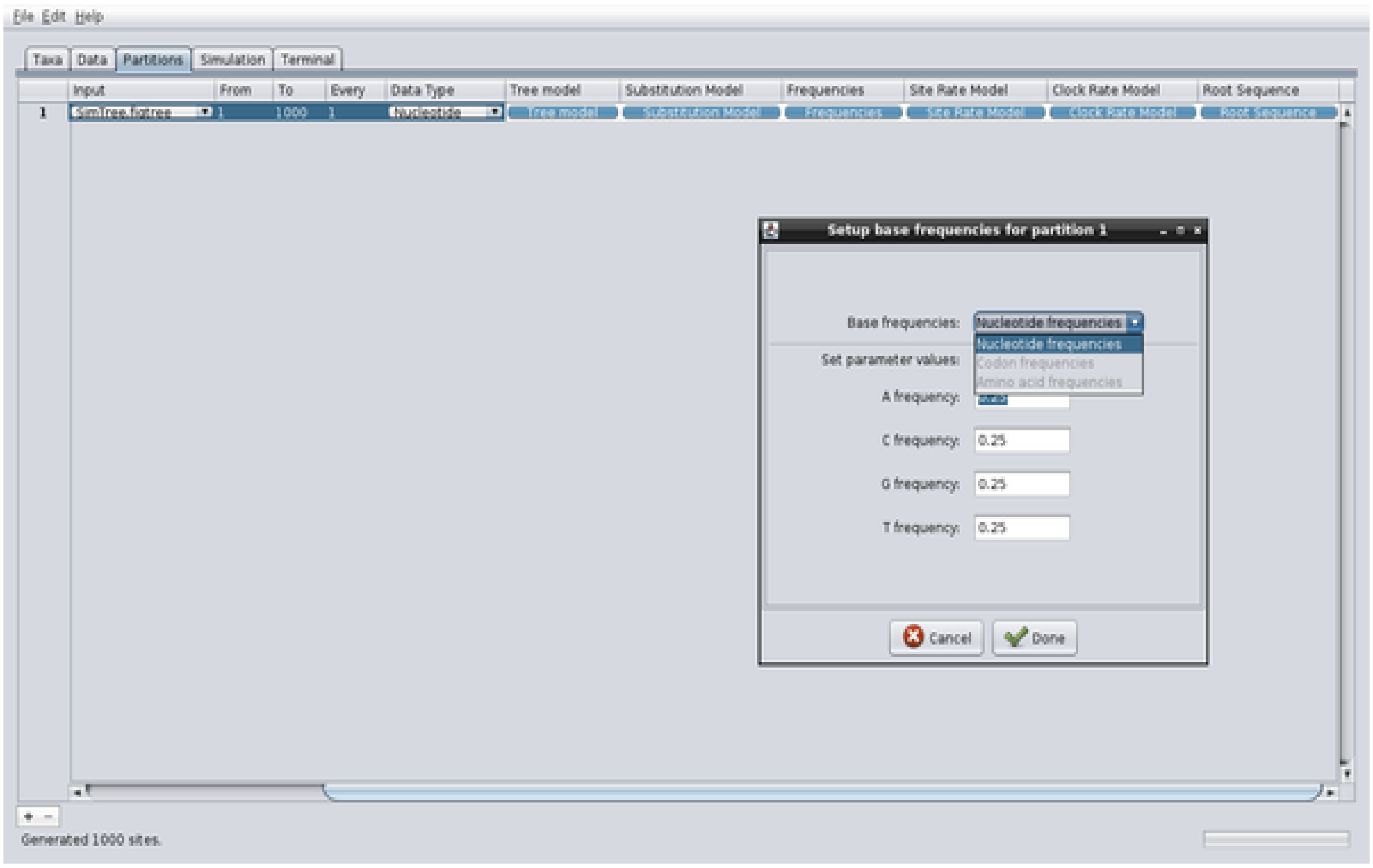
Invariant sites proportion: 0.0

Cancel Done

3 taxa loaded.

Figure 5. Site heterogeinity can be simulated by going to the Partitions and click on the 'Site Rate Model' button.

The frequency with which each character (nucleotide or codon) appears is also an important aspect as different organisms may have different base frequencies. In πBUSS, one can specify frequencies for nucleotides or codons by going to the **Partitions** and click the 'Frequency Model' button and specify the frequencies (Figure 6). For nucleotides, four parameters need to be specified (default is 1/4 for all) and for codons, there are 61 frequencies to be set – with 1/61 uniformly for all as default.



	Input	From	To	Every	Data Type	Tree model	Substitution Model	Frequencies	Site Rate Model	Clock Rate Model	Root Sequence
1	SimTree.figtree	1	1000	1	Nucleotide	Tree model	Substitution Model	Frequencies	Site Rate Model	Clock Rate Model	Root Sequence

Setup base frequencies for partition 1

Base frequencies:	Nucleotide frequencies
Set parameter values:	
A frequency:	
C frequency:	0.25
G frequency:	0.25
T frequency:	0.25

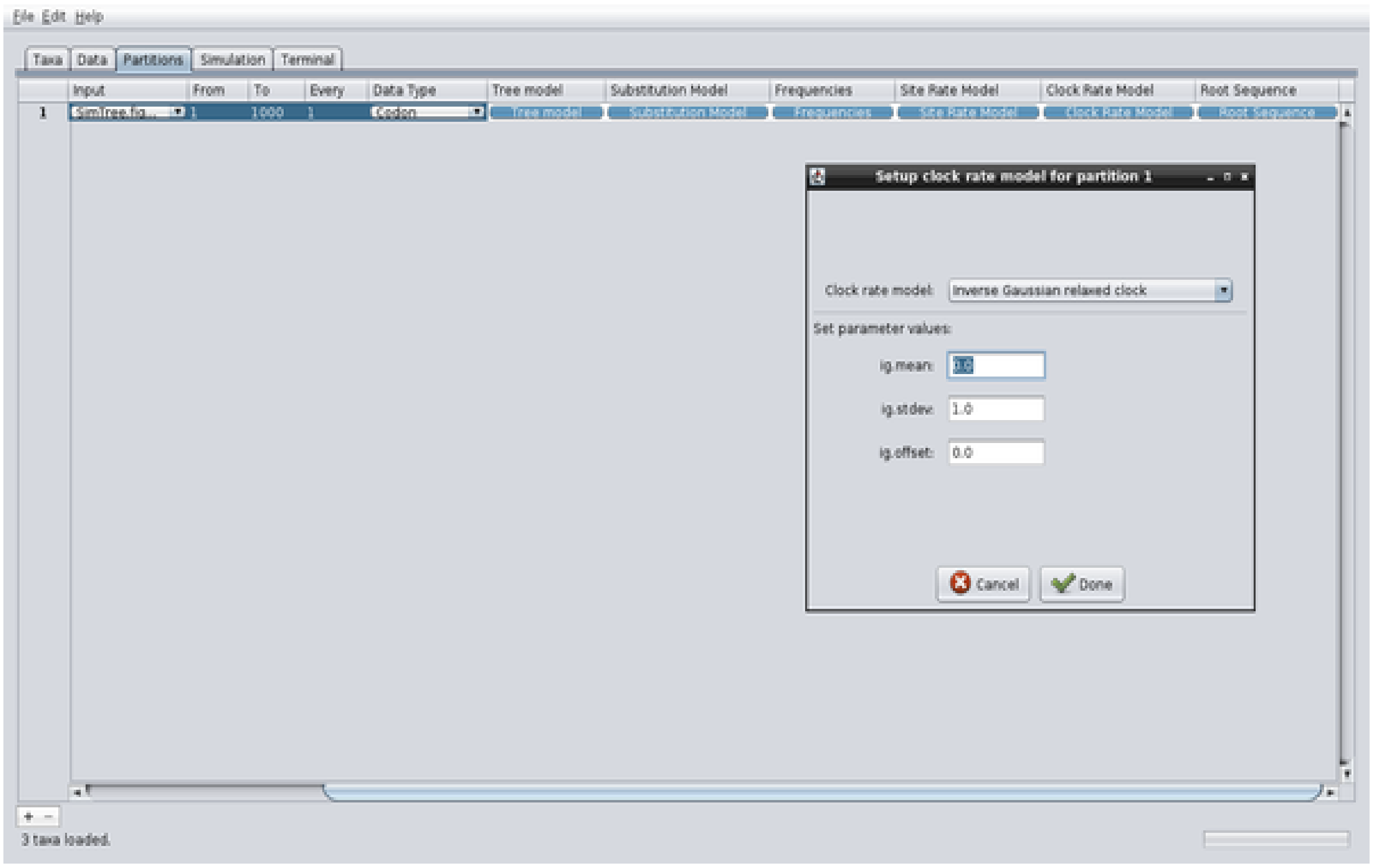
Nucleotide frequencies
Codon frequencies
Amino acid frequencies

Generated 1000 sites.

Figure 6. To specify base frequencies, go to the Partitions table and click the ‘Frequency Model’ button.

Molecular clock

Traditionally, it was assumed that each branch in a phylogeny had the same rate of evolution. Numerous studies have shown this assumption not to hold in a number of real-world situations. πBUSS offers several options of relaxed molecular clock models. Each model specifies a distribution from which a mutation rate is drawn for each branch in the phylogeny (tree). The user can choose between the (default) Strict clock model, as well as three relaxed clock models: Exponential, Log-normal and Inverse Gaussian. Each of the relaxed models specifies a distribution with different parameters. For the Strict model, the user needs to supply a single parameter, the clock rate. For the Exponential model, the mean and offset of the exponential distribution are necessary. For Log-normal and Inverse Gaussian models, the user needs to specify mean, standard deviation and offset. See Figure 7 for an example.



	Input	From	To	Every	Data Type	Tree model	Substitution Model	Frequencies	Site Rate Model	Clock Rate Model	Root Sequence
1	SimTree.fig...	1	1000	1	Codon	Tree model	Substitution Model	Frequencies	Site Rate Model	Clock Rate Model	Root Sequence

Clock rate model: Inverse Gaussian relaxed clock

Set parameter values:

ig.mean: 0.0

ig.stdev: 1.0

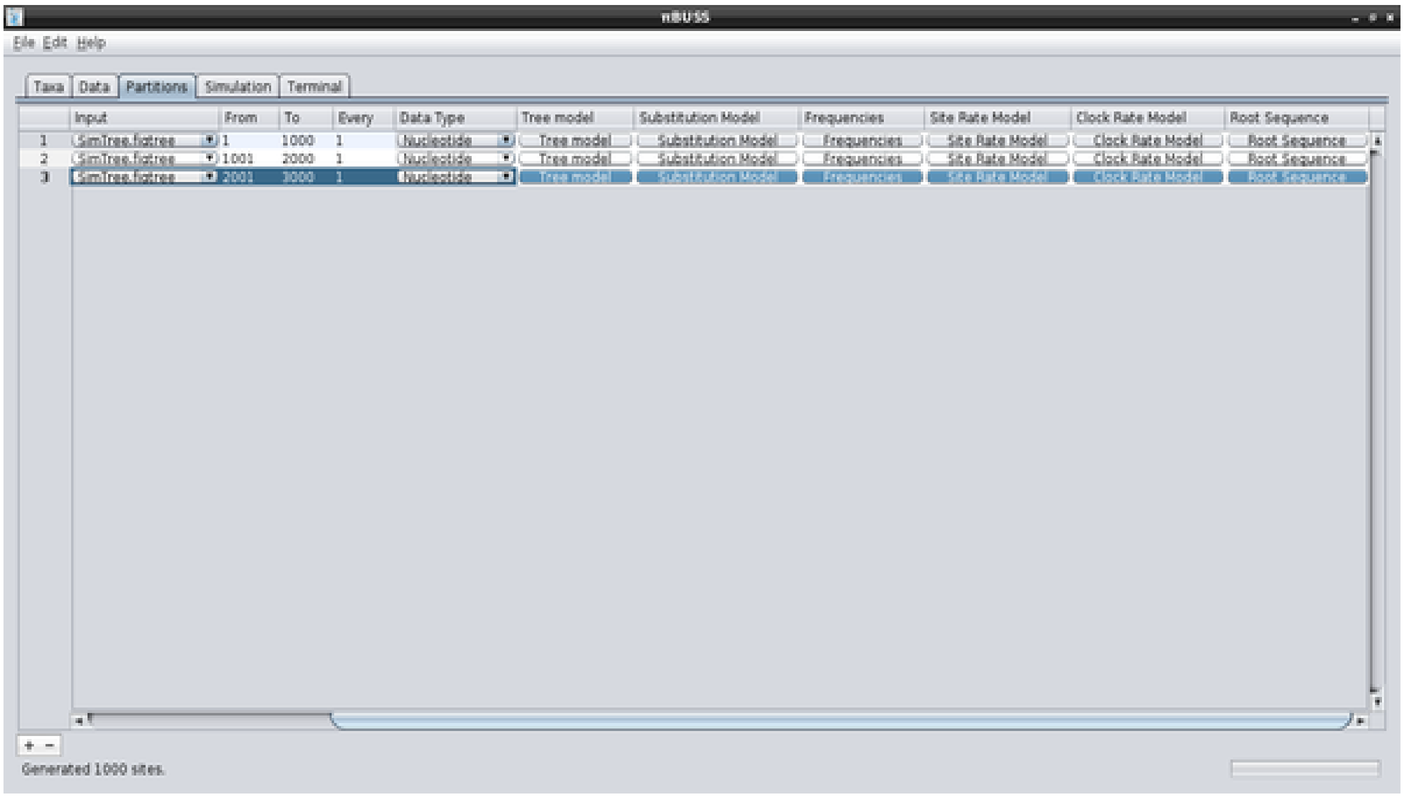
ig.offset: 0.0

3 taxa loaded.

Figure 7. Setting up an Inverse Gaussian relaxed clock with mean= 0, 1 unit standard deviation and no offset.

Setting multiple partitions

Simulating partitions is a way of reproducing the heterogeneity in evolutionary dynamics we observe in different portions of the genome. Every step described above can be repeated for any number of partitions. πBUSS provides an intuitive way of setting multiple partitions, which can have different models and parameters for sequence evolution, site heterogeneity, base frequencies and molecular clock. Each partition can either have its own topology, or the tree structure can be shared between partitions. Partition settings apply from the position in the alignment specified as 'from' to the position specified as 'to', on every position being the multiple of 'every' parameter.



	Input	From	To	Every	Data Type	Tree model	Substitution Model	Frequencies	Site Rate Model	Clock Rate Model	Root Sequence
1	SimTree.figtree	1	1000	1	Nucleotide	Tree model	Substitution Model	Frequencies	Site Rate Model	Clock Rate Model	Root Sequence
2	SimTree.figtree	1001	2000	1	Nucleotide	Tree model	Substitution Model	Frequencies	Site Rate Model	Clock Rate Model	Root Sequence
3	SimTree.figtree	2001	3000	1	Nucleotide	Tree model	Substitution Model	Frequencies	Site Rate Model	Clock Rate Model	Root Sequence

Generated 1000 sites.

Figure 8. You can set up any number of partitions, each with its own tree topology, model of evolution, molecular clock and base frequency. Just click the '+' button at the bottom left corner of the GUI window at the Partitions tab.

Saving and loading created settings

The software comes with a simple save/load option, that allows to preserve settings generated via GUI to be loaded or edited at some other time. To store GUI settings, go to the main menu's File and select 'Save setting...' to choose name and location for the serialized inputs. To load such preserved settings, again choose File - 'Load setting...' and navigate to the previous location.

Creating and running an XML file

πBUSS provides an XML parser that allows the user to store the simulation settings in XML files, that can be later imported into the program and modified if necessary. Importantly, the generated XMLs can be run in BEAST, invoking the core implementation of the simulator. Another advantage of the XML functionality is that the user can generate XMLs using the GUI and then edit these files to attain more advanced options for data evolving under complex evolutionary scenarios, e.g. an epoch model (see the **Do the evolution, baby!** section below). This adds substantial flexibility to users whose needs exceed the complexity of evolutionary scenarios available through the GUI. XML editing can also be used to simulate discrete phylogeographic trait data (see **Do the evolution, baby!** for an example) . To export an XML, just click the 'Generate XML' button at the **Simulation** tab. To run the generated file, the user just needs to load it into BEAST with BEAGLE. For Windows users, this reduces to double-clicking the BEAST executable and then checking the 'Use BEAGLE library' box. For UNIX/Linux users, issuing

```
java -Djava.library.path=$BEAGLE_PATH beast.jar -beagle path/to/file.xml
```

at the terminal does the job. For further details on BEAST/BEAGLE usage, please see http://beast.bio.ed.ac.uk/.

Using the command line interface (CLI)

πBUSS offers a command line interface (CLI), that can be used for pipelining in big simulation studies to generate large amounts of different simulation settings. The CLI options mirror those available from the GUI, with an intuitive syntax. In table 1 we present the CLI options for each feature in πBUSS.

Parameter	Command	Options
Tree topology	-treeFile	-
Taxa set	-taxaSet	-
Demographic (coalescent) model	-demographicModel	-constantPopulationParameterValues -exponentialGrowthRateParameterValues -exponentialDoublingTimeParameterValues
Markov model of evolution	-branchSubstitutionModel	-HKYsubstitutionParameterValues -GTRsubstitutionParameterValues -TN93substitutionParameterValues -GY94substitutionParameterValues
Site rate heterogeneity	-siteRateModel	-gammaSiteRateModelParameterValues
Molecular clock model	-clockRateModel	-strictClockParameterValues -lognormalRelaxedClockParameterValues -exponentialRelaxedClockParameterValues -inverseGaussianRelaxedClockParameterValues
Base frequencies	-baseFrequencies	-nucleotideFrequencyParameterValues -codonFrequencyParameterValues
Ancestral (root) sequence	-ancestralSequence	-

Table 1 List of command line options for πBUSS.

As a general example, consider simulating some data under an HKY model, gamma-distributed rates using 4 rate categories and no invariant sites, for two partitions of 500 sites each and separate topologies. The commands for this would look like:

```
 java -Djava.library.path=$BEAGLE_PATH -jar pibuss.jar -treeFile Tree1.tree -from
1 -to 500 -every 1 -branchSubstitutionModel HKY -HKYsubstitutionParameterValues
1.0 -siteRateModel gammaSiteRateModel -gammaSiteRateModelParameterValues 4.0 0.5
0.0 : -treeFile Tree1.tree -from 501 -to 1000 -every 1 -branchSubstitutionModel
HKY -HKYsubstitutionParameterValues 10.0 -siteRateModel gammaSiteRateModel -
gammaSiteRateModelParameterValues 4.0 0.5 0.0 : sequences.fasta
```

Check the **Do the evolution, baby!** section for more examples of how to use the CLI.

2 Examples: Do the evolution, baby!

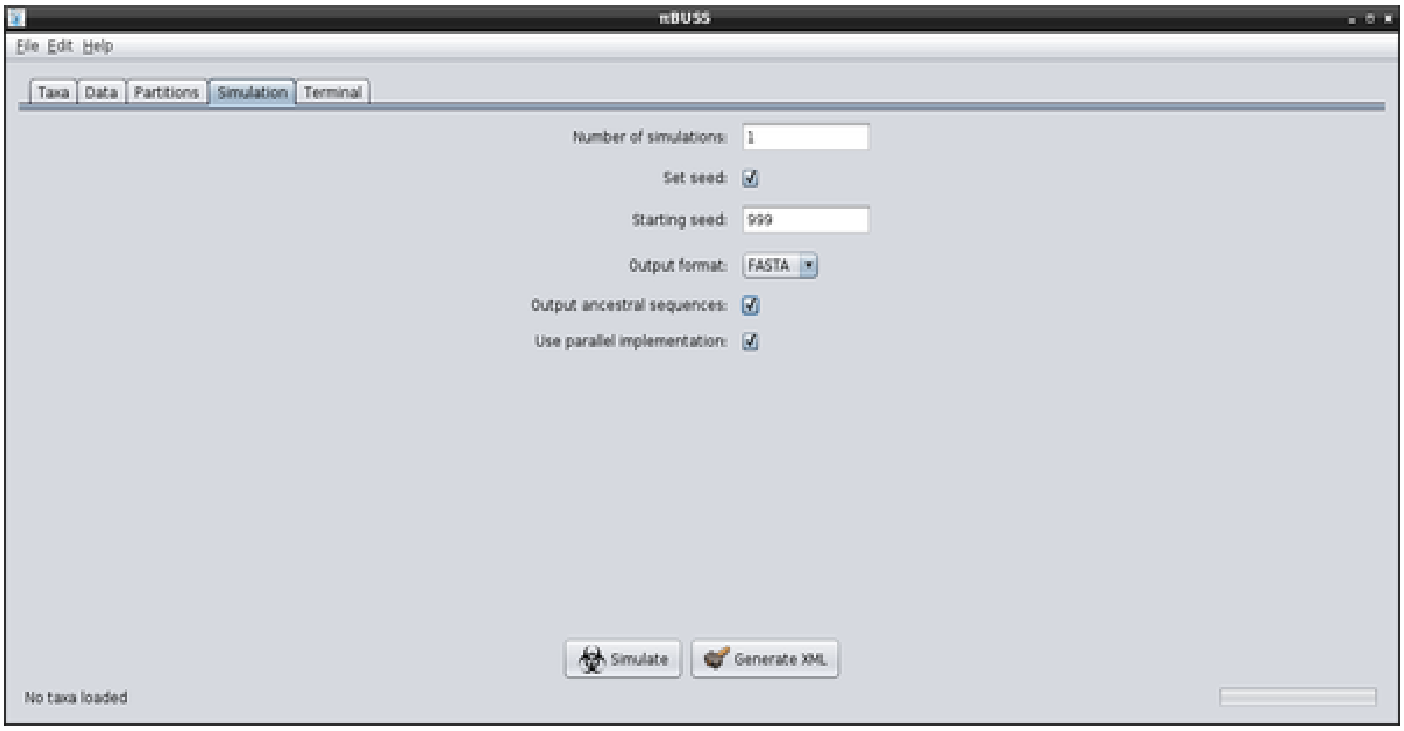
In this section we present a couple of possible simulations that can be done using πBUSS. πBUSS can either directly generate the data or export the settings to an XML file to be run in BEAST, invoking the core implementation of the simulator.

If you want your simulation results to be exactly reproducible, you can set the seed for the random number generator by checking a tick box at the **Simulation** tab. If you are simulating directly from πBUSS, you may also want to use the novel and much faster parallel implementation. This is easily accomplished by checking the corresponding tick box at the **Simulation** tab. Figure 9 shows a screenshot where both these options are enabled.

Using the CLI, the seed is parsed after parsing all the arguments for partitions. The user can specify a seed and choose to use the parallel implementation by adding

```
 [<output-file-name>] [<seed>] [<true|false>]
```

after the last ``:'' sign dividing the desired partitions. The last argument corresponds to the use of the parallel implementation, which is false by default. The ancestral sequences in the inner nodes can be saved by checking the corresponding box in the **Simulation** tab.



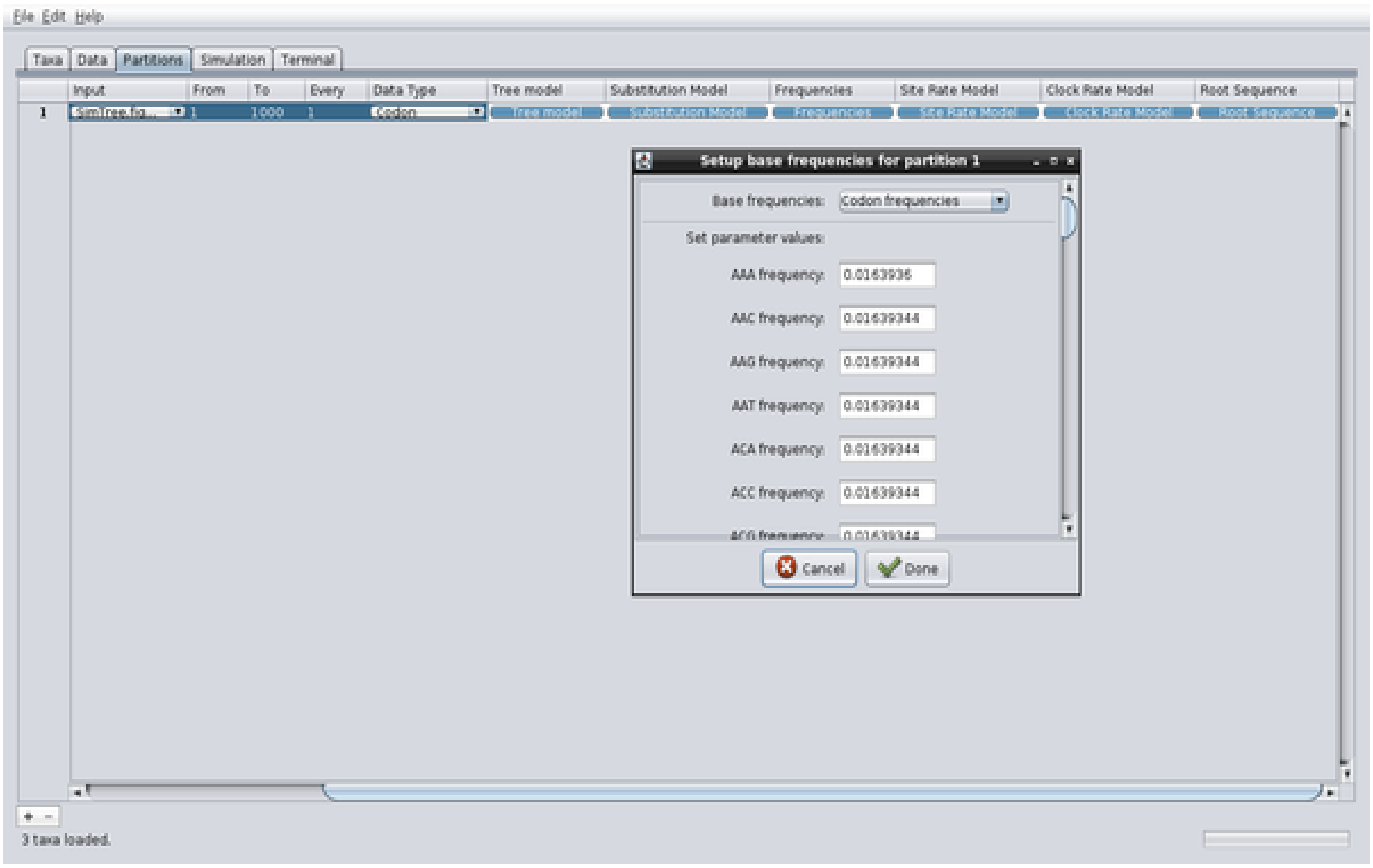
Number of simulations: 1
Set seed: ✓
Starting seed: 999
Output format: FASTA
Output ancestral sequences: ✓
Use parallel implementation: ✓
No taxa loaded

Figure 9. To set the seed for your simulation and choose the parallel BEAGLE implementation, check the corresponding tick boxes at the Simulation tab.

Generating data evolving under GY94 codon model

To generate sequence data evolving according to a codon substitution model we start by importing a backbone tree topology. Load the *SimTree.figtree* tree file shipped with this tutorial as described in *Loading a tree topology* subsection.

Then go to the **Partitions** panel and select the just loaded topology from the drop-down list in the *Data* column. Next click on the button under the *Base Frequencies* column, select codon frequencies as shown in the Figure 10. Confirm your choice and go to the **Simulation** panel to generate an XML or simulate a fasta or nexus file.



	Input	From	To	Every	Data Type	Tree model	Substitution Model	Frequencies	Site Rate Model	Clock Rate Model	Root Sequence
1	SimTree.fig...	1	1000	1	Codon	Tree model	Substitution Model	Frequencies	Site Rate Model	Clock Rate Model	Root Sequence

Setup base frequencies for partition 1

Base frequencies: Codon frequencies

Set parameter values:

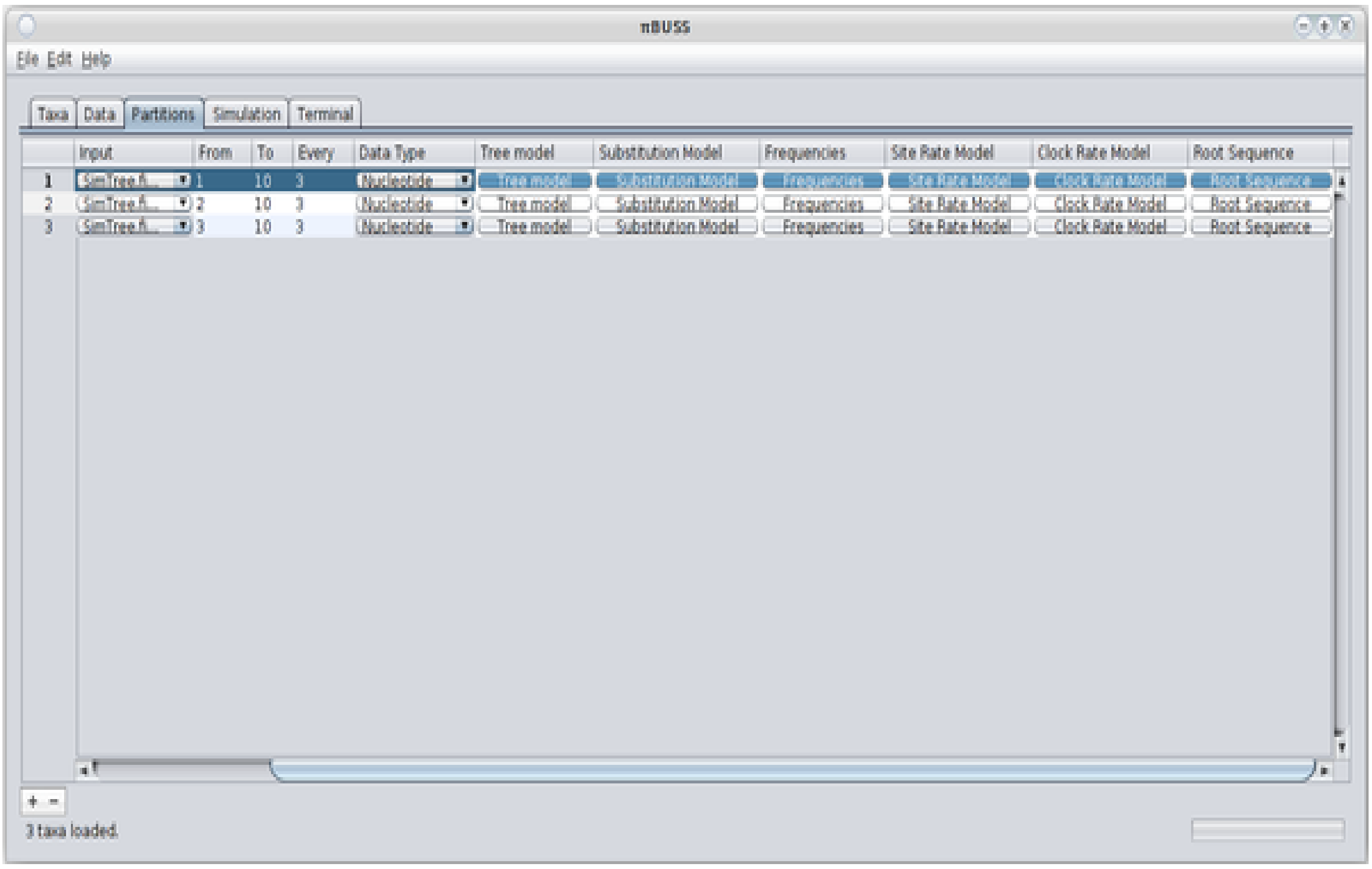
AAA frequency:	0.0163936
AAC frequency:	0.01639344
AAG frequency:	0.01639344
AAT frequency:	0.01639344
ACA frequency:	0.01639344
ACC frequency:	0.01639344
ACG frequency:	0.01639344

3 taxa loaded.

Figure 10. Setting codon frequencies. Just select codon frequencies in the drop-down menu and you’re done. The default is 1⁄61 for each codon.

Generating codon partitioned data

As before, start by loading the *SimTree.figtree* tree file. This time we will however need to set three separate partitions, from different positions in the alignment and simulating to every third alignment position, just like in Figure 11.



	Input	From	To	Every	Data Type	Tree model	Substitution Model	Frequencies	Site Rate Model	Clock Rate Model	Root Sequence
1	SimTree.fi...	1	10	3	Nucleotide	Tree model	Substitution Model	Frequencies	Site Rate Model	Clock Rate Model	Root Sequence
2	SimTree.fi...	2	10	3	Nucleotide	Tree model	Substitution Model	Frequencies	Site Rate Model	Clock Rate Model	Root Sequence
3	SimTree.fi...	3	10	3	Nucleotide	Tree model	Substitution Model	Frequencies	Site Rate Model	Clock Rate Model	Root Sequence

3 taxa loaded.

Figure 11. Codon partitioned simulation.

We can change the substitution dynamics under which the data will be generated by modifying the default value of the HKY model kappa parameter, for example for the second partition. Change the κ parameter value to 10.0 using the *Branch Substitution Model* button in the second row of the **Partitions** panel and go to **Simulation** panel to generate the output.

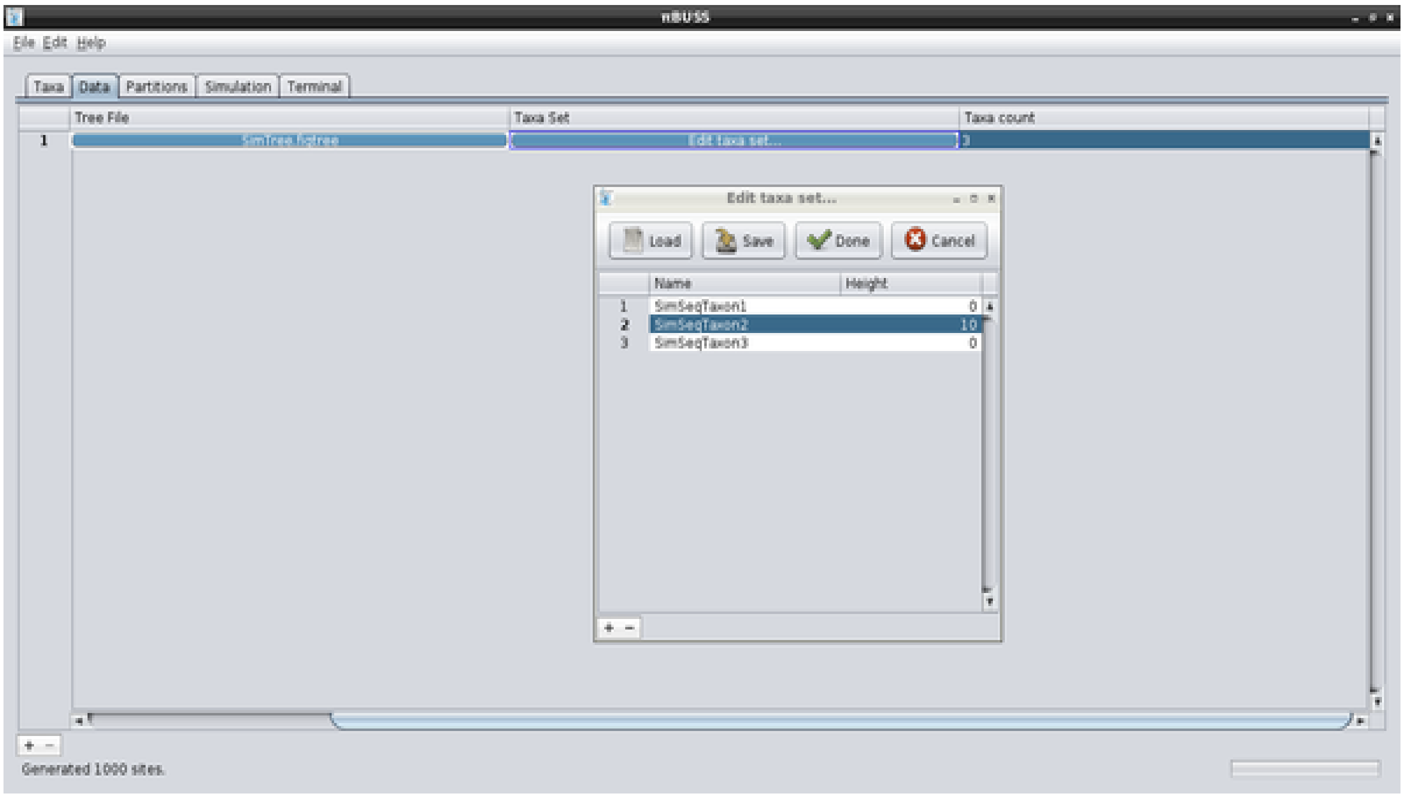
Equivalently one might use the CLI interface, then the above operations sum up to issuing the following command:

```
 java -Djava.library.path=/usr/local/lib -jar pibuss.jar -treeFile
SimTree.figtree.tree -from 1 -to 1000 -every 3 -branchSubstitutionModel HKY -
branchSubstitutionModel HKY -HKYsubstitutionParameterValues 1.0 : -treeFile
SimTree.figtree.tree -from 2 -to 1000 -every 3 -branchSubstitutionModel HKY -
branchSubstitutionModel HKY -HKYsubstitutionParameterValues 10.0 : -treeFile
SimTree.figtree.tree -from 3 -to 1000 -every 3 -branchSubstitutionModel HKY -
branchSubstitutionModel HKY -HKYsubstitutionParameterValues 1.0 :
sequences.fasta
```

Simulating a tree topology

πBUSS can be used to simulate a bifurcating tree topology under the coalescent process driven by a specified model. The only input needed in this case is a set of taxa with corresponding heights. These can be loaded from an existing tree topology or specified using the editor in **Data** panel, under *Taxa set* column.

In this example, we will use a set of taxa shipped with this tutorial in the *taxa.txt* file. Use the editor's *Load* button to import the file like in Figure 12. Set the taxa as simulations backbone in drop-list under *Data* column in **Partitions** panel.



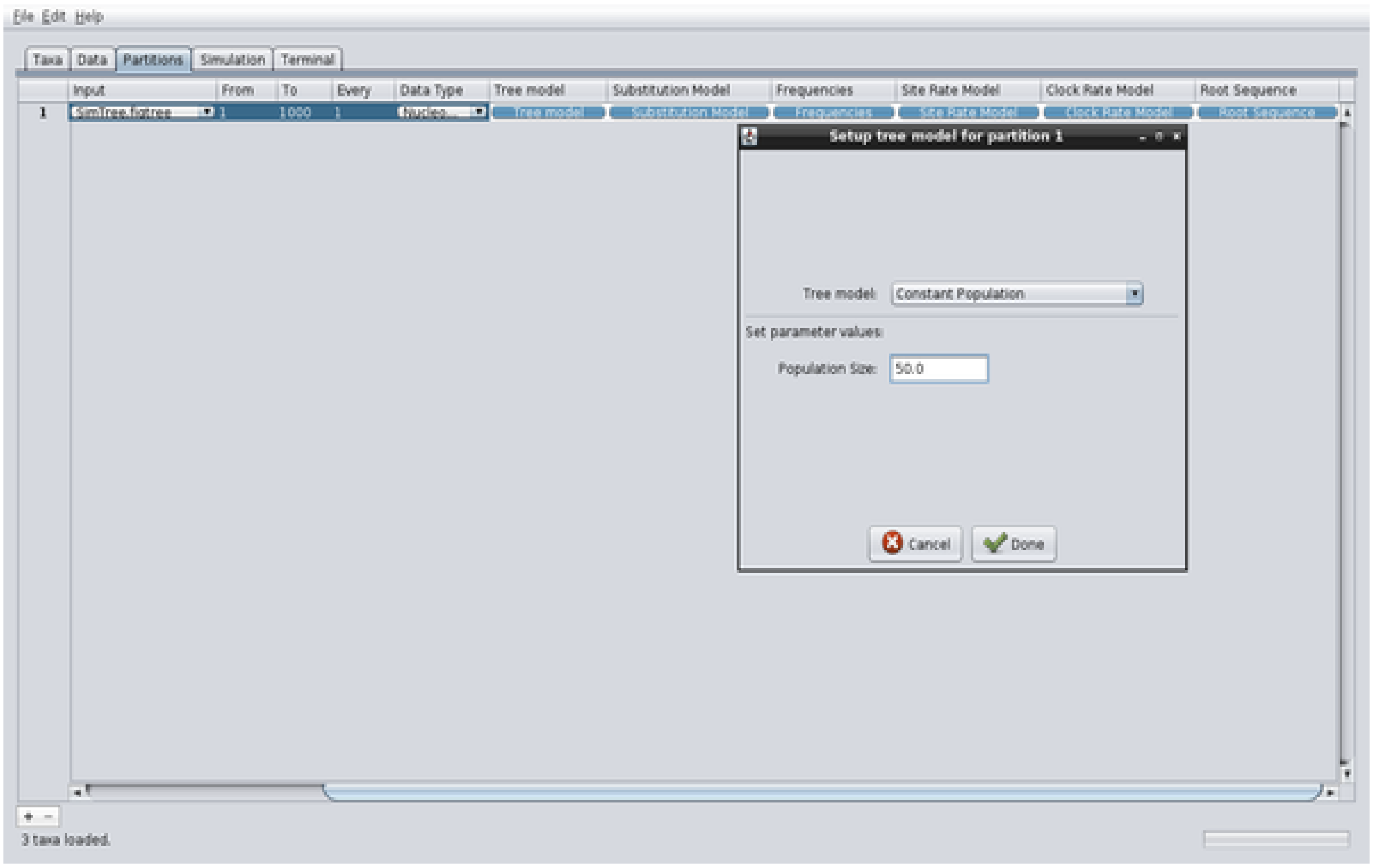
	Tree File	Taxa Set	Taxa count
1	SimTree.figtree	Edit taxa set...	3

	Name	Height
1	SimSeqTaxon1	0
2	SimSeqTaxon2	10
3	SimSeqTaxon3	0

Generated 1000 sites.

Figure 12. Loading a taxa set from tab-separated file. Once you load the taxa set onto πBUSS you are ready to simulate under any of the coalescent dynamics available.

Now it's time to set the demographic model that will drive the coalescent. Click on the editor button under *demographic model* column and choose the *Constant Population* model with *Population Size* parameter of 50.0, like shown in Figure 13 below.



Tree model: Constant Population

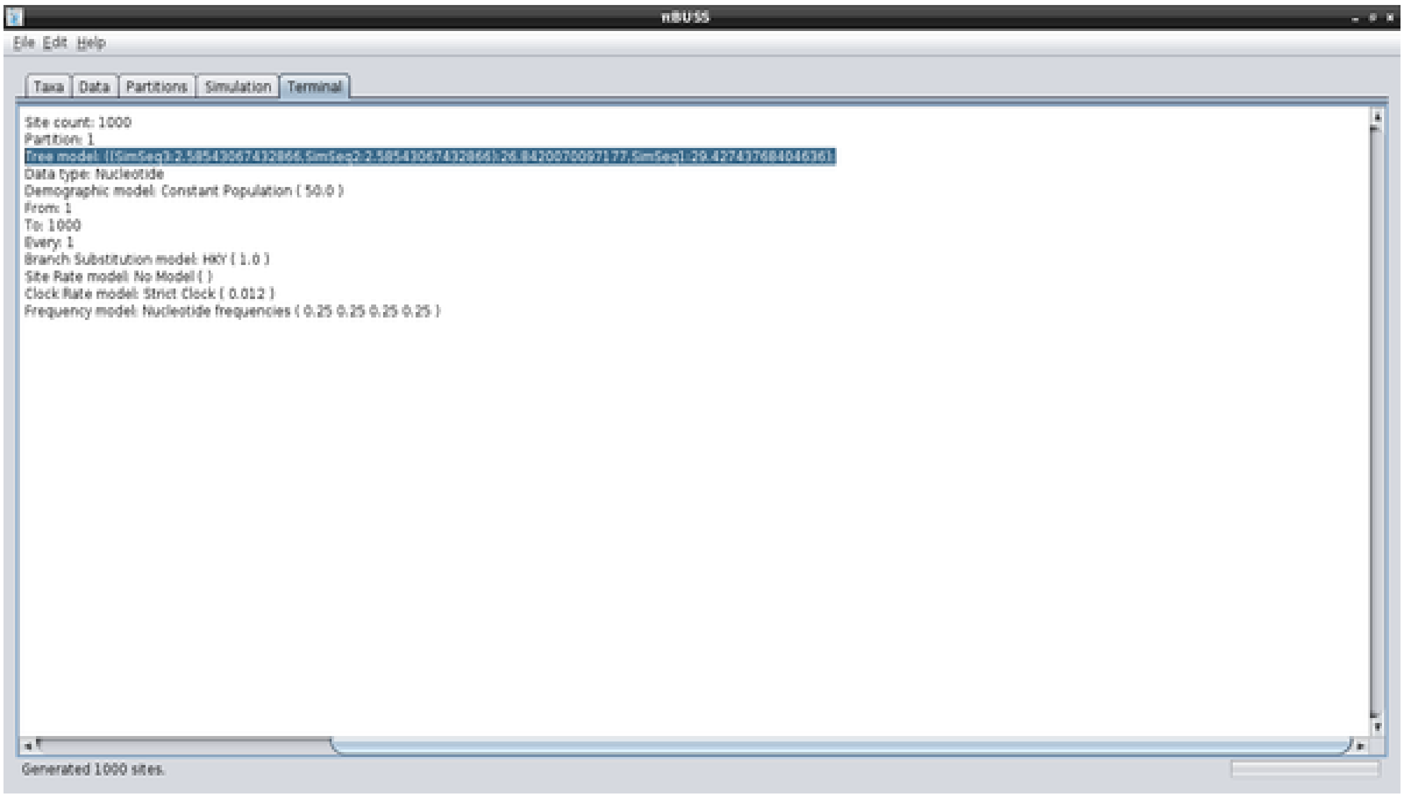
Set parameter values:

Population Size: 50.0

3 taxa loaded.

Figure 13. Loading a taxa set from tab-separated file.

To generate the data, click on the *Simulate* button in the **Simulation** panel. The generated tree (newick format) can be copy-pasted from **Terminal** panel as shown in Figure 14.



```
Site count: 1000
Partition: 1
Tree model: ((SimSeq3:2.58543067432866,SimSeq2:2.58543067432866):26.8420070097177,SimSeq1:29.427437684046361)
Data type: Nucleotide
Demographic model: Constant Population ( 50.0 )
From: 1
To: 1000
Every: 1
Branch Substitution model: HKY ( 1.0 )
Site Rate model: No Model ( )
Clock Rate model: Strict Clock ( 0.012 )
Frequency model: Nucleotide frequencies ( 0.25 0.25 0.25 0.25 )
```

Generated 1000 sites.

Figure 14. Terminal output.

Extending a created XML to generate discrete phylogeographical trait data

By modelling phylogeography as a continuous-time Markov chain (CTMC) we are able to make use of the same computational machinery used for the branch substitution Markov models. In this setting, each location is treated as a state of the chain and we can set transition/transversion parameters in the same fashion.

In addition to the command-line and graphical interface, πBUSS is capable of producing XML files that can be used with BEAST software to invoke the core implementation of the software. For large and complex simulation studies this should be the preferred way of interacting with the software.

We can use the default XML generated for us by πBUSS as a canvas to extend to discrete geographical trait simulation. Start with loading the *SimTree.figtree* tree file and setting it as data in **Partitions** panel. Now go to the **Simulation** panel and click on *Generate XML* button, saving the file as *generate_sequences.xml*. Open the generated file in your favourite text editor.

We need to define the discrete geographical locations that πBUSS will sample from. After the *taxa* block paste the following code:

```
<generalDataType id="geography">
   <state code="location1"/>
   <state code="location2"/>
   <state code="location3"/>
</generalDataType>
```

Scroll to the *frequencyModel* block. Because we have specified K = 3 discrete locations we need to specify base frequencies for them. Delete the existing *frequencyModel* block and paste there following code instead:

```
<frequencyModel id="freqModel">
   <generalDataType idref="geography"/>
   <frequencies>
      <parameter dimension="3" value="0.3 0.3 0.4"/>
   </frequencies>
</frequencyModel>
```

Now scroll further down to the substitution model block (defined between *HKYModel* tags if you generated the XML using default settings). We need to define a geographical substitution model, driven by a rate matrix. We specify K × (K − 1) = 6 entries such that the underlying instantaneous rate matrix is asymmetric (non-reversible). Replace the existing block with:

```
<complexSubstitutionModel id="originModel.simulation" randomizeIndicator="false">
   <generalDataType idref="geography"/>
   <rootFrequencies>
     <frequencyModel idref="freqModel"/>
   </rootFrequencies>
   <rates>
     <parameter id="rates.simulation" dimension="6" value="1.0"/>
   </rates>
</complexSubstitutionModel>
```

Now reference the created substitution model inside *siteModel* tags like this:

```
<siteModel id="geoSiteModel.simulation">
   <substitutionModel>
     <complexSubstitutionModel idref="originModel.simulation"/>
   </substitutionModel>
</siteModel>
```

Finally, all the new elements need to be referenced in the *beagleSequenceSimulator* block. We specify one replicate, as every taxa can ultimately have only one geographical trait:

```
<beagleSequenceSimulator id="simulator">
   <partition from="1" to="1">
     <treeModel idref="treeModel1"/>
     <complexSubstitutionModel idref="originModel.simulation"/>
     <siteModel idref="geoSiteModel.simulation"/>
     <strictClockBranchRates idref="branchRates1"/>
     <frequencyModel idref="freqModel"/>
   </partition>
</beagleSequenceSimulator>
```

Resulting document is available as *generate_discrete_phylogeography_SimTree.xml* file shipped with this tutorial. The created XML can be invoked within BEAST to generate the trait data by issuing the following command:

```
 java -Djava.library.path=$BEAGLE_PATH -jar beast.jar -beagle_CPU
generate_sequences.xml
```

Note that the presented example is just a fraction of analysis possible by editing the πBUSS/BEAGLE XMLs and we urge the users to play with the available options.

3 Final remarks

License

Availability and requirements

Beagle Sequence Simulator’s source code is freely available as a GoogleCode repository: http://code.google.com/p/beast-mcmc/ Compiled, runnable packages targeting all major platforms along with tutorial on using the software are hosted at: https://rega.kuleuven.be/cev/ecv/software/pibuss.

Citing πBUSS

We have invested a lot of time and effort in creating πBUSS, please cite it when using it in your research. BibTeX entry:

```
@article{bielejec2014,
author = {Bielejec, Filip and Lemey, Philippe and Carvalho, Luiz M. and Baele, Guy
title = {πBUSS: a parallel BEAST/BEAGLE utility for sequence simulation under comp
volume = {15},
number = {1},
pages = {133},
year = {2014},
journal = {BMC Bioinformatics}
}
```

Rega homepage