

TraitLab users' manual

Geoff K. Nicholls, David Welch

July 4, 2007

1 Introduction

TraitLab is a software package for simulating, fitting and analysing tree-like binary data under a stochastic Dollo model of evolution. A full description of the Dollo model can be found in Nicholls and Gray [2007]. The core of the package is a Markov chain Monte Carlo sampling algorithm that allows the user to sample from the Bayesian posterior distributions for tree topologies, root ages and trait loss rates for a given data set. Data can be simulated according to the strict Dollo model that is fitted or according to a generalized model which allows for borrowing (horizontal transfer) of traits, heterogeneity in the trait loss rate and biases in the data collection process. Both the raw data and the output of MCMC runs can be inspected using a number of useful graphical and analytical tools provided within the package. TraitLab is freely available and runs within the Matlab computing environment.

2 Getting started

2.1 System requirements

TraitLab is written in the Matlab programming language. Thus, to run TraitLab, Matlab (version 6, release 12.1 or later) must already be installed on your system. Matlab is proprietary software which runs on Windows, Mac OSX or Unix/Linux machines. TraitLab only requires the basic Matlab installation without any additional packages. For more information on Matlab, see www.mathworks.com or ask your local mathematics, statistics or engineering department who may be able to help.

2.2 Download and installation

Download the TraitLab zip archive from

<http://www.math.auckland.ac.nz/~nicholls/TraitLab/>

and extract all files, preserving the subdirectories, to `C:\TraitLab` (or any other convenient location - if using another directory, use that name instead of `C:\TraitLab` in the following.)

2.3 Running TraitLab

Start Matlab in the usual manner and make `C:\TraitLab` the current directory. Add the directory `C:\TraitLab\guifiles` to the Matlab path, by using the Matlab path browser (File menu, Set Path option, select Add Folder), or by typing in the command: `addpath guifiles -end`.

To start TraitLab and bring up the main gui, type `TraitLab` at the Matlab command line.

3 Formatting Data for use by TraitLab

TraitLab accepts data in the Nexus file format (see Maddison et al. [1997] for a full description of the format). The binary data matrix is recorded in the *Data* block, while any clade constraints are recorded in the *Clades* block which is specific to TraitLab. See section 3.4 for an example data file.

3.1 Data block

The first command in the Data block must be the Dimensions command with values for `ntax` (the number of taxa) and `nchar` (the number of characters or traits) defined.

The Format command where the missing and gap characters are defined is optional. If it is not present, the missing character is assumed to be '?' and the gap character is assumed to be '-'.

Note that TraitLab can not treat missing data or gaps. These will not cause an error if they occur in the matrix, but will be recoded as absent (i.e., replaced by a 0) when they are read in. If large amounts of the data is missing, the results from TraitLab may therefore be unreliable.

The Matrix command is compulsory. Rows of the matrix are labeled with the taxa names, which may not contain any whitespace. The content of the matrix must be zeros and ones to indicate absence or presence of the trait. The matrix may be in standard format (where the rows are `nchar` characters long) or in interleaved format (where the matrix is split in sections of a manageable size). If the matrix is interleaved, each section of the matrix must have the rows labeled by the taxa names and sections must be separated by blank lines. Comments may only occur between interleaved sections of the matrix - comments at the start or end of rows will cause errors.

The Charlabels command, followed by a list of exactly `nchar` trait names may also appear in the Data block. All other commands in the Data block are ignored.

3.2 Clades block

Prior knowledge about the structure of the tree and divergence times can be encoded in the *clades* block. The *clades* block consists of a series of clade commands, one for each known clade. It has the following form:

```

BEGIN Clades;

Clade Name = Clade_1
Taxa = taxon_a,...,taxon_k
Rootmin = t1
Rootmax = t2
Originatemin = t3
Originatemax = t4;

Clade Name = Clade_2 ...

End;

```

Each clade command must include a name and the list of taxa that define the clade. The time of the most recent common ancestor (the root) of the clade can be bounded by defining *rootmin* and *rootmax*. The time that clade diverged from all other taxa (the node above the root node) can be bounded by defining *originatemin* and *originatemax*. See Figure 1 for an example of the difference between root and originate bounds. All time bounds on a clade are optional.

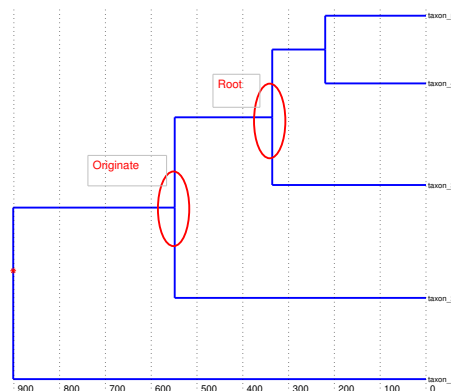


Figure 1: Suppose a clade consists of taxon_3, taxon_4 and taxon_5. If $rootmin = t_1$ and $rootmax = t_2$, then node labeled root here is constrained to lie in the interval (t_1, t_2) . Similarly if $originatemin = t_3$ and $originatemax = t_4$, the node labeled originate is constrained to lie in the interval (t_3, t_4) . Note that one “child” node of the originate node is necessarily the root node of the clade in question, but the other child node is arbitrary (here, it is the leaf node of taxon_2).

3.2.1 Offset leaves

If the taxa are sampled at significantly different times, the clades block is used to encode this information. If a clade has only one taxon, then the *rootmax* and

rootmin definitions for that clade are upper and lower bounds for the sampling time of that taxon. Time zero is defined as the time of the most recently sampled taxa. The upper and lower time bounds must not be the same, so if a taxon was known to have been sampled 500 years before the most recently sampled taxon, set rootmax to be 501 and rootmin to be 499.

3.3 Other blocks

When data is synthesized using TraitLab, a trees block and a synthesize block is generated. The trees block contains the tree on which the data was synthesized and the synthesize block contains the parameter values used for the synthesis. If either of these blocks is found, TraitLab will assume that the data is synthesized.

The characters block which may contain taxa names and/or trait labels can be read by TraitLab but we advise against its use. Include any relevant information in the data block instead.

All other blocks are ignored by TraitLab.

3.4 Example data file

```
#NEXUS

BEGIN DATA;

DIMENSIONS NTAX=9 NCHAR=30;
FORMAT MISSING=? GAP=- INTERLEAVE ;

MATRIX

taxon_1  000111101011010100010100100000
taxon_2  101111010111001111001011011000
taxon_3  011011010111101110001010011000
taxon_4  010111100111011100110000100100
taxon_5  110111101111011100110100100100
taxon_6  111111010111101111001011011011
taxon_7  111111010110101111001011011011
taxon_8  111111000111101110001011011011
taxon_9  111111010111101111001011011011
;
END;

BEGIN CLADES;

CLADE  NAME = Clade_1
ROOTMIN = 346
ROOTMAX = 422
TAXA = taxon_4, taxon_5;
```

```

CLADE NAME = Clade_2
Orginatemim = 346
TAXA = taxon_1, taxon_8, taxon_9;

END;

```

4 Synthesizing data

Bring up the synthesize GUI (shown in Figure 2) by choosing “synthesize data” in the mode menu from the main TraitLab GUI.

The screenshot shows the following GUI components:

- Set trait parameters (teal panel):**
 - Mean number of traits:
 - Loss Rate:
 - Rate heterogeneity across branches
 - Standard deviation (relative):
 - Impose no empty field assumption
 - Observation classes:
 - Rate heterogeneity across classes
 - Standard deviation:
 - Remove rare traits
- Set borrowing parameters (light blue panel):**
 - Allow borrowing
 - Borrowing rate (relative):
 - Local borrowing
 - Max borrow distance:
- Set clade parameters (olive green panel):**
 - Synthesize clades
 - Number of Clades:
 - Bounds within: %
 - Bounds on origin
 - Bounds on root
 - Bounds on either
- Select tree on which to synthesize data (purple panel):**
 - Synthesize on random tree
 - Branching rate:
 - Taxa:
 - Synthesize on tree from file
 - Tree file: None selected
 - Dir:
 -
 - The file contains 0 trees
 - Use tree:
- Select file for saving synthetic data (light blue panel):**
 - Data file: synthdata.nex
 - Dir: H:\m\projects\TraitLab_beta2\
 -
- Bottom:**

Figure 2: The data synthesizing GUI.

To synthesize data, the tree on which traits are synthesized needs to be

specified, as do parameter values for the trait mutation model, the borrowing model and the clade model. See Nicholls and Gray [2007] for a detailed and rigorous description of the stochastic Dollo model of evolution.

4.1 Selecting a tree

The tree can either be a randomly generated Yule tree (with exponential branch lengths) or any rooted tree (with branch lengths specified) stored in the Newick format in a nexus file.

To synthesize data on a random tree, the branching rate and the number of taxa need to be specified. The branching rate is the parameter θ in the Dollo evolution model, so the mean length of each branch is $1/\theta$ years.

To synthesize data on a tree from a file, click the “Select tree file” button. TraitLab will attempt to extract all readable trees from the specified file. If there is more than one tree found in the file, specify which tree to use in the “use tree” text box. Check that you have selected the correct tree with the “view” button.

If there are trees in a file but TraitLab is unable to read them, check that they are written in the correct format with a root and branch lengths specified.

4.2 Defining trait evolution parameters

Defining the mean number of traits (per taxa), K , and the loss rate, ν , completely specifies the standard Dollo evolution model. These must be specified in the text boxes provided. The loss rate, ν , is a number between 0 and 1 and can be interpreted as the mean proportion of traits that are lost in a lineage over a period of 1000 years.

Note that ν is related to the per trait death rate, μ , which is used in the Dollo model by

$$\nu = 1 - \exp(-1000\mu).$$

Similarly, given the mean number of traits per taxa, K , and a value of μ , the per taxon trait birth rate, λ , is defined by $\lambda = K\mu$.

4.2.1 Rate heterogeneity and the no empty field assumption

To simulate heterogeneity across branches in the loss rate, check the “rate heterogeneity across branches” checkbox. If the box is checked, a lineage specific trait death rate, $\mu_b(i)$ applies to every different branch, i , of the tree where μ_b is a gamma distributed random variable with mean μ and variance $(\sigma_b\mu)^2$. It is necessary to specify the relative standard deviation parameter, σ_b .

Checking the “impose no empty field assumption” checkbox will produce data as if traits are collected based on pre-specified observation classes for which it is assumed every observed taxa will have at least one trait in each class. This is called the no empty field assumption as we assume that each observation class is occupied by at least one trait in each taxa at all times. If this option is checked, it is necessary to specify the number of observation classes. Note that

the option will have the greatest effect on the simulated data when the number of observation classes is close to the mean number of traits per taxa, K .

If the no empty field assumption is imposed, heterogeneity in trait death rates across observation classes may be simulated by checking the “rate heterogeneity across classes” checkbox and specifying a relative standard deviation, σ_c in the associated “standard deviation” text box. Traits in observation class j then have a death rate $\mu_c(j) \sim \Gamma(\text{mean} = \mu, \text{variance} = (\sigma_c \mu)^2)$.

When rates vary across both branches and observation classes, a trait on branch i in observation class j dies at rate $\mu_{b,c}(i, j) \sim \Gamma(\text{mean} = \mu_b(i), \text{variance} = (\sigma_c \mu_b(i))^2)$ where $\mu_b(i) \sim \Gamma(\text{mean} = \mu, \text{variance} = (\sigma_c \mu_b)^2)$.

4.2.2 Rare traits

After the trait data has been simulated, traits that appear in no taxa are automatically discarded. That is, columns in the matrix with no ones are discarded. If the “remove rare traits” checkbox is checked, traits that occur in only one taxa are also discarded. In most real data sets, such traits are not observed.

4.2.3 Borrowing

Data can be synthesized with global or local borrowing. Global borrowing is when all lineages are equally likely to borrow from one another, whereas local borrowing means that only lineages that are close, in the sense that they share a recent common ancestor within some specified time period, may borrow traits from each other.

Global borrowing is synthesized by checking the “allow borrowing” checkbox and specifying a relative borrowing rate, b . Each trait is borrowed at constant rate $b\mu$. When a trait, k , is borrowed, it chooses a lineage, i , uniformly at random from those extant. If trait k is not present in lineage i , k is added to i , otherwise nothing happens.

To make borrowing local instead of global, check the “local borrowing” checkbox and specify a borrowing distance, d . All traits are still borrowed at constant rate $b\mu$ but the target lineage is not chosen from all others. Instead, only lineages that share a common ancestral lineage with the source lineage (the one in which the trait to be borrowed is found) in the last d years may borrow the trait. The target lineage is chosen uniformly at random from all such lineages.

When death rates are heterogeneous and borrowing occurs, a trait on branch i in observation class j is borrowed at rate $b\mu_{b,c}(i, j)$.

4.3 Synthesizing clades

Check the “synthesize clades” checkbox to synthesize clades. When there are n taxa in the synthetic data, up to $n - 1$ clades can be synthesized corresponding to the $n - 1$ internal nodes of the tree. It is necessary to specify the accuracy of the clade bounds, c , in the “bounds within” text box. An accuracy of c means

that, if the chosen node has a time t in the tree, a lower bound of $(1 - c/100)t$ and an upper bound of $(1 + c/100)t$ will be created.

By clicking on the appropriate radio buttons, the bounds can be on the root, on the divergence time (originate bounds) or be chosen uniformly at random between the two. In most real data, the bounds are on the root.

4.4 Output of synthesize GUI

The synthetic data will be saved the `.nex` file specified by the user by clicking the "select data file" button, say `synfile.nex`. This file will contain the synthetic trait matrix, the tree on which the data was synthesized and associated model parameters. An `.par` file, `synfile.par`, will also be generated for record keeping purposes. All options and parameter values used to produce the data as specified in the GUI are recorded here. Checkbox options are represented by 0 for unchecked and 1 for checked.

When the "synthesize now" button is pressed, the data will be synthesized according to the specified parameters and options. Progress can be monitored in the Matlab command window. A figure showing the tree on which the data was synthesized will appear.

Once the data has been saved, diagnostic tools are run on the synthetic data. An explanation of how to interpret the histograms and the depth distance graph that are generated here is given in Section 6.3 below.

5 Setting up an MCMC analysis

MCMC analyses are set up and run in TraitLab from the main GUI shown in Figure 3.

The four coloured panels correspond to four classes of information that need to be provided to start a run: the data source, the initial state of the chain, the model choice (priors and which parameters to estimate) and the run length and output files.

5.1 Data source

Click the "select data file" button and choose a nexus file conforming to the TraitLab requirements described in Section 3. Progress in reading the file can be monitored in the Matlab command window.

If the file is successfully read, the number of taxa and traits will be shown above the "select data file" button. These numbers should agree with the respective numbers of rows and columns in the matrix in the data file. If any clades were found in the file, the number found is shown below the "select data file" button. If the data was synthesized using TraitLab, this will also be indicated below the "select data file" button.

The names of the taxa and any clades found in the data file will appear in the Matlab command window. The numbers that appear next to the trait

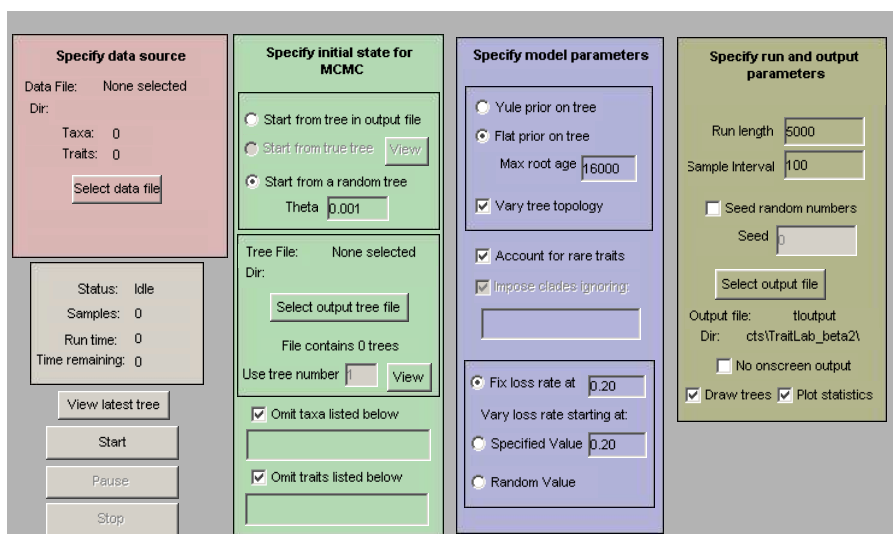


Figure 3: The main TraitLab GUI.

and clade names are the order in which they appear in the file and are used to specify taxa and clades to be ignored (see Sections 5.2.2 and 5.3.1).

5.2 Initial state

5.2.1 Initial tree

The initial tree may be random, be chosen from a previous TraitLab run or, if the data was synthesized within TraitLab, be the true tree on which the data was synthesized.

If a random initial tree is selected, the branching rate, θ , must be specified.

To start from a tree in a previous run, the output file where the tree is to be found needs to be specified using the “select output tree file” button. Once the file has been selected, a tree in the file is specified by the “use tree number” text box. Use the adjacent “view” button to view the specified tree. The trees in the file must have taxa names which are a superset of the taxa names in the current run.

Note that if the initial tree is chosen from an output file, the user should ensure that the chosen tree satisfies all constraints to be imposed in the new run including the maximum root age and any clade constraints. Specifying a tree that does not satisfy all the necessary constraints will result in all proposed states being rejected and a failed run.

5.2.2 Omitting taxa and traits

Any of the taxa and traits can be omitted from a run so long as at least two taxa remain to be analysed. Omit taxa by checking the "omit taxa listed below" checkbox and listing taxon numbers in the space provided. A taxon's number corresponds to its row number in the data matrix in the data file. They are also printed out to the Matlab command window after the data file is read in. Omitting taxa that appear in clade constraints can cause problems, see Section 5.3.1 for more details.

Similarly, traits are omitted by checking the appropriate check box and listing a vector of trait numbers. Trait numbers correspond to columns of the data matrix as they appear in the data file. It is left to the user to find the correct numbers of traits to be masked.

Note that the lists of numbers should satisfy Matlab formatting. That is numbers must be separated with commas or spaces. Sequences of consecutive numbers can be abbreviated using the the colon operator (so write 2:5 for 2, 3, 4, 5). For example, to omit taxa 1, 10,11,12,13 and 20 type "1 10:14 20" in the space provided.

5.3 Model specification

Radio buttons determine the type of the prior imposed on on the tree: either a Yule prior (so trees are given prior weight according to branch lengths which have an exponential penalty) or a flat prior where all trees with a root in a specified range are given equal prior weight. If a flat prior is chosen, a maximum allowable root age must be specified. If clades are imposed, the maximum root age must be greater than the largest of the `rootmin` and `originatemin` bounds imposed.

If the "vary tree topology" box is unchecked, the tree will remain in same shape throughout the run with only the branch lengths varying.

The "account for rare" checkbox determines the form of the likelihood. If it is checked (the standard setting), the calculation of the likelihood takes into account a normal data collection process where traits observed only in a single taxa are not recorded. In this case, any traits that are observed in just a single taxa are automatically removed before the analysis begins. When unchecked, the likelihood is calculated as if all traits present in at least one observed taxon have been recorded. Note that, for real data sets, it is highly unusual for traits present in only one taxon to be recorded and it is thus recommended that the checkbox remain checked.

The trait loss rate, ν (see Section 4.2), is an estimable parameter when clades with time constraints are imposed. To estimate ν , choose one of the "vary loss rate" options, either starting at the value specified or at a random value drawn from the prior distribution. If no clade constraints are imposed or to simply keep the loss rate at a constant value throughout the run, choose the "fix loss rate" option and specify a value.

5.3.1 Imposing clades

If clade constraints are present in the data file, they may be applied during a run by checking the “impose clades” checkbox. To omit any clades from the analysis, list the clades to be omitted in space provided using the respective clade numbers (see section 5.2.2 for details on clade numbers and list formatting).

Note that difficulties can arise when omitting taxa that appear in imposed clades. For example, consider clade C that consists of taxa x, y and z with a root time between t_1 and t_2 . If taxon z is omitted from the analysis, it is often incorrect to simply omit it from clade C to get the new clade C' consisting of just taxa x and y with the same root time constraints as C . This is because x and y could have a common ancestor at time t where $t < t_1$. The user is given the option, in the MatLab command window, of simply deleting the offending taxa from the clades in question (as was done to get from clade C to clade C' in the example) or leaving it and causing an error which can be remedied by adding the offending clades to the clades to ignore list. To delete the taxa from the clade, type “1” at the command line when given the option “delete taxa/leave? 1/0”. Be aware that deleting taxa from clades in this way can have unintended consequences: the new clade may be inconsistent with other clades or, when all but one taxon is deleted from a clade, the clade root time constraint is reinterpreted as an offset leaf (see Section 3.2.1).

If problems are encountered omitting taxa that occur in clade constraints, the cleanest solution is to create another data file containing the same data block but a clade block that has been altered to achieve the desired constraints with the offending taxa removed.

5.4 Run and output parameters

Specify the total length of the MCMC run, r , and sub-sample interval, k . The integer part of $(r/k) + 1$ samples from the run are recorded, being the initial state and each k th state thereafter.

The “seed random numbers” checkbox is mainly used for debugging purposes and is generally left unchecked. If it is checked, a seed for the random number generator needs to be specified (it can be any real number). If it is unchecked, random seed will be generated based on the date and time.

5.4.1 Output files

Use the “select output file” button to specify the file and directory to which the run is to be saved. Supposing `tlout` is the chosen file name, three output files are created: `tlout.nex`, `tlout.txt` and `tlout.par`.

The `.par` contains a record all parameter values and options used to create the run. It can be used to perform similar runs in batch mode (see Section 7 for details).

Each sampled state is recorded in the `.nex` and `.txt` files. The sampled trees are in the `.nex` file and sampled scalar values (parameters and densities)

are in the `.txt` file. The recorded parameter values are the root time for the sampled tree, the trait death rate μ , the cladistic loss probability, p (currently not estimated so always equals 1), and the branching rate λ . Note that λ is integrated analytically rather than using the MCMC sampler so that the value recorded here is an independent draw from the marginal posterior distribution of λ at the given state. Also in the `.txt` file are values of the log prior density for the state, the integrated log likelihood (where λ has been integrated out) and the log likelihood (using the value of λ recorded).

5.4.2 Monitoring a run

The progress of the run can be monitored via values output to the Matlab command window, a plot of the most recently sampled tree and trace plots of the parameter and log density values. There is also a status box on the left of the GUI showing the number of samples taken, to run time to date and an estimate of the time remaining.

To plot each tree as it is sampled, check the “draw trees” checkbox. If the box is unchecked, the most recently sampled tree can be plotted by clicking the “view latest tree” button on the left of the GUI.

To see the trace plots of the sampled root time, trait loss rate, log prior density and integrated log likelihood density, check the “plot statistics” checkbox. Note that the log likelihood trace plot is truncated to omit the early sampled values so that the vertical scale of the plot is reasonable. If the data was synthesized using TraitLab, horizontal lines showing the “true” parameter values for the data are shown on the trace plots.

Unless the “no onscreen output” checkbox is checked, when each sample is taken, a row of numbers will appear in the Matlab command window. These numbers show the sample number, the log likelihood value and several statistics showing the proportion of proposed MCMC updates were accepted since the previous sample. For example

```
(5, -3550.486804) 0.33 0.24 0.09 0.14 0.03 0.21 0.38 . . .
```

means sample 5 has log likelihood of around -3550 and, of the updates between sample 4 and sample 5, 0.33 of the states proposed by move type 1 were accepted, 0.24 of those proposed by move type 2 were accepted and so on. A brief description of each of the moves is provided in Table 1. Note that, depending on the options chosen, some moves may not be proposed so the proportion accepted will be 0.

6 Output and data inspection

The raw data and the results of MCMC runs can be inspected in the analysis GUI, shown in Figure 4. Bring up the analysis GUI by choosing “analyse output” in the mode menu from the main TraitLab GUI.

Table 1: The moves used in the MCMC sampler to explore the state space.

Move	Description
1	vary internal vertex time
2	interchange neighbouring edges
3	interchange randomly chosen edges
4	move edge to new neighbouring location
5	move edge to new random location
6	rescale whole tree
7	rescale random subtree
8	random walk on μ
9	(unused)
10	(unused)
11	vary leaf time
12	vary root time

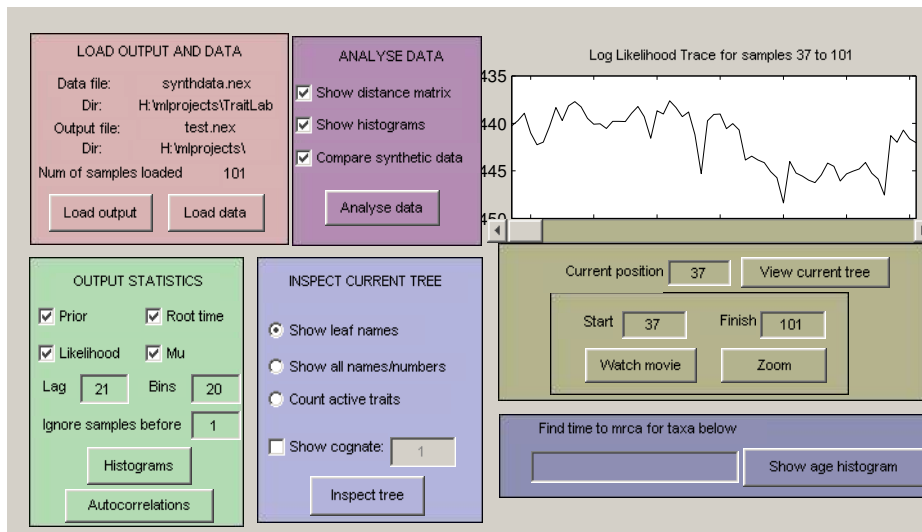


Figure 4: The analysis GUI.

The tools provided for exploring the output of MCMC runs include viewing sampled trees, making trace plots and histograms of sampled statistics from the `.txt` output file and making histograms of the root time of user specified clades through the run.

The data inspection tools provided are histograms of the number of traits per taxon and of the number of taxa per trait, comparisons with synthetic data and plots based on a maximum *a posteriori* estimator of the time the most recent common ancestor for a pair of taxa.

6.1 Loading data and output

When the analysis GUI is called, the output from the most recent completed run (if there is any) along with the currently loaded data is automatically loaded into the analysis GUI. If no run is in memory, nothing is passed to the analysis GUI and data and output can be loaded using the “load output” and “load data” buttons. Note that it is left to the user to ensure that the loaded output and data are compatible.

6.2 Exploring MCMC output

6.2.1 Viewing sampled trees

At the top right of the GUI is a trace plot of the log likelihood of sampled states in the loaded run. Trees from the loaded run can be viewed using the “view current tree” button and (when data is also loaded) more closely investigated using the “inspect current tree” functions. The current tree is the tree at the current position. The current position is shown as a red line on the log likelihood and can be specified either in the “current position” text box or by using the slider below the trace plot.

To zoom in on a section of the trace plot, specify the first and last samples of the section in the “start” and “finish” text boxes and click the “zoom” button.

The sampled trees between “start” and “finish” can be watched as a movie by clicking the “watch movie” button. The movie shows approximately 3 samples per second and can be halted by closing the window it shows in.

When data is loaded, the inspect tree functions can be used to closely investigate sampled trees. If the “show leaf names”, the standard plot of the current tree is drawn when the “inspect tree” button is clicked. The “show all names/numbers” option is mainly used for debugging and shows the node numbering used to internally represent the tree. The numbers on the leaf nodes shown in this representation are those used to specify taxa for the TMRCA histograms (see Section 6.2.3).

The “count active traits” option shows the standard tree plot of the current state where, at each internal node of the tree, the number of traits found exclusively in the clade defined by that node is shown. For example, suppose that the current tree has three taxa a , b and c in which (a, b) form a clade and there are 20 traits found in a and b that are not found in c . Then the number 20 will

be shown at the internal node (a, b) . Note that the number shown at the root of the tree is always the total number of traits in data set being analysed (once traits found only in taxa not in the tree have been removed).

By checking the “show trait” checkbox and specifying a trait number, the path of the specified trait on the current tree can be viewed. According to the standard Dollo model, the trait must have been present on the lineages shown in black, so must have been born either above the root of the tree or on the lineages shown in red. It was either not present or died at some point on the lineages in blue. An example of a trait history is shown in Figure 5.

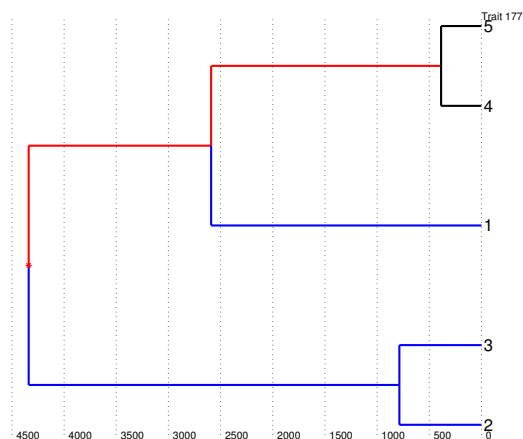


Figure 5: An example of the use of the “show trait” function for inspecting a tree. The specified trait here is found in taxa 4 and 5, so must have been present on the lineages shown in black and could have arisen on any of the lineages shown in red (or above the root).

6.2.2 Autocorrelations and histograms

The “output statistics” area provides functions for calculating and plotting histograms and lag autocorrelation functions of any of the log prior density, the log likelihood density, the tree root time or the trait death rate, μ stored in the `.txt` output file. Choose which statistics to plot using the appropriate checkboxes.

To plot the autocorrelation functions, it is necessary to specify the maximum lag for which autocorrelations are calculated in the “lag” text box. The maximum value of the lag should be significantly smaller than the number of samples being analysed. To ignore the first part of the run as burn-in, specify the first sample to use in the “ignore samples before” text box. Trace plots of the selected statistics are automatically drawn alongside the lag autocorrelation plots. The title of the lag autocorrelation plot has the variable name, the integrated autocorrelation time, τ_f , and the number M , an estimate of number

of samples after which the normalised autocovariance function is approximately zero and N , the number of samples analysed. An rough estimate of the number of independent samples obtain for statistic f is N/τ_f . An example of an autocorrelation plot is shown in Figure 6. For further information about how to interpret autocorrelation plots and associated statistics, see Nicholls [1998] or Geyer [1992].

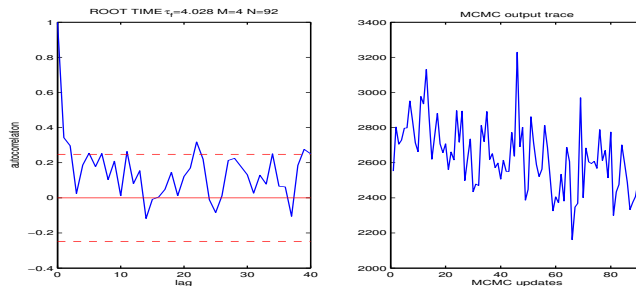


Figure 6: The lag autocorrelation function of the variable (root time, in this case) is plotted on the right. On the left is the trace plot of the variable.

To plot histograms of the sampled statistics, specify the number of bins to use in the histogram. A burn-in can be specified using the “ignore samples before” text box. The histograms of the root time and μ represent their respective estimated marginal posterior distributions.

6.2.3 Histogram of TMRCA for given taxa

The marginal posterior distribution of the time to the most recent common ancestor (TMRCA) for any group of taxa in the analysis can be estimated using the “show age histogram” button. Specify the taxa of interest by using their respective numbers in the text box provided. The taxon numbers can be found using the “inspect tree” function (see Section 6.2.1). The first part of the run can be ignored as burn-in using the “ignore samples before” text box and the number of bins to use in the histogram is specified in the “bins” text box, both found in the output statistics section of the GUI.

When the “show age histogram” button is clicked, the TMRCA for the given taxa in each sampled tree is found and this information is output to the Matlab command window (copy it to a text file to analyse it elsewhere) as well as being made into a histogram. The posterior mean, the 95% posterior credible interval and the posterior probability that the given taxa form a clade are all output to the Matlab command window.

6.3 Inspecting the data

Clicking the “analyse data” button will produce a number of plots which are described below. Note that both data and output needs to be loaded to view these plots since a tree is required for the construction of the the distance-depth plot.

6.3.1 Data histograms

If the “show histograms” checkbox is checked, histograms of the number of taxa per trait and the number of traits per taxon will be generated. The taxa per trait histogram is a histogram made from the column sums of the data matrix. If there are k taxa and L traits in total, each trait occurs in between 0 and k taxa, thus the horizontal axis in the graph runs from 0 to x and the vertical axis (the frequency) runs from 0 to (at most) L .

The traits per taxon histogram is a histogram made from the row sums of the data matrix. It is the empirical distribution of the number of traits found in each taxa.

If the “compare synthetic data” checkbox is checked, synthetic data according to the standard stochastic Dollo model is generated on the current tree and histograms of the synthetic data are displayed below those for the loaded data.

If the model fits the data well and the tree is a representative draw from the posterior distribution, the two pairs of histograms should be qualitatively very similar. Consistent qualitative differences between the two pairs of histograms may indicate some significant model misspecification.

6.3.2 Distance depth and distance matrix plots

When all traits, including those found only in a single taxa, are observed it is possible to analytically calculate the maximum a posteriori estimate of the time to the most recent common ancestor for a pair of taxa under the standard Dollo model (see equation 10 of Nicholls and Gray [2007]). The distance matrix plot (check the “show distance matrix” checkbox) and the distance depth relation plot (which is always shown when the “analyse data” button is pressed) are based on this calculation. Note that we assume that rare traits (traits that occur in only one taxon) are not observed, so, in order to make an accurate estimate of the TMRCA, rare traits are synthesized and blended with the data in the file to make these plots.

The distance matrix graph is simply a heat plot of a matrix where this distance has been calculated for each pair. Bluer colours represent closer taxa, red more distant and green intermediate. It is best understood by referring to the example in Figure 7.

For the depth distance relation plot, the maximum a posteriori estimate for the TRMCA for each pair is calculated and the time of the MRCA of each pair in the specified tree is found. The two times for each pair are then plotted against each other. See Figure 8 for an example. The tree used is the current output tree (or, when the graph appears after synthesizing data, the tree on

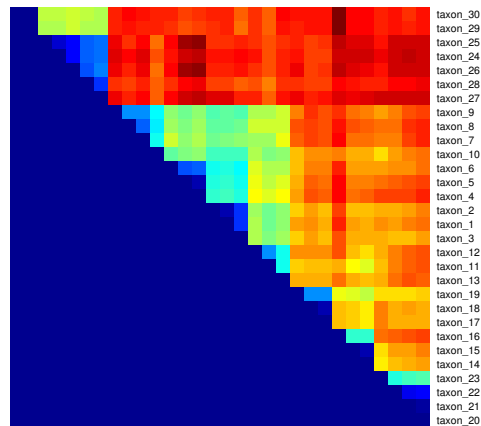


Figure 7: An example of the distance matrix graph. The column labels are the same as the row labels. The pixel at row i , column j represents the maximum a posteriori estimate of the TMRCA between taxon i and taxon j . The darker the blue the closer the taxa are, the darker the red the further apart they are. Lighter and green colours are intermediate. Note that the ordering of taxa is as it occurs in the data file, so the obvious clustering that is shown here will not occur if rows in the data file are randomly ordered.

which the data was synthesized. If the standard Dollo model and the specified tree fit the data well, the points should lie along the line $y = x$. Systematic deviation of the points away from the line $y = x$ may indicate some significant model misspecification.

7 Running TraitLab in batch mode

TraitLab can be run without the GUI interface directly from a Unix, Linux, Mac or Matlab command line.

All parameters for a run are stored in the file called `batchtlinput.txt` in the main TraitLab directory. This has exactly the same form as a `.par` output file which is automatically created at the start of every run and has options corresponding to those in the main TraitLab GUI. Either modify `batchtlinput.txt` field by field, or simply copy an existing `.par` file into, modify as necessary and save. It may be easiest, if many similar runs are being made, to set up a short model run using the GUI and then use the `.par` file as a template for the other runs in batch mode.

To start a run in batch mode make the main TraitLab directory the current working directory and type

```
matlab -nodisplay < batchTraitLab.m > outlog &
```

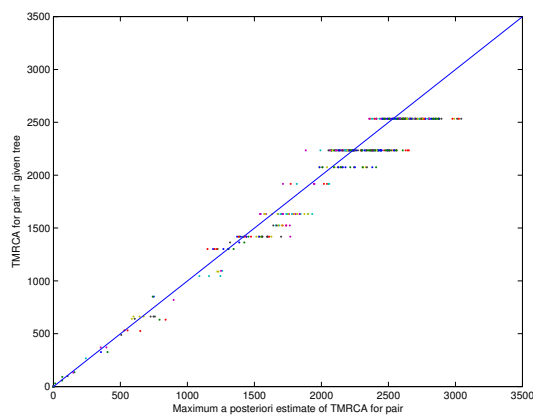


Figure 8: An example of the depth distance relation graph. Each point represents a pair of taxon. The position of the point along the x -axis is the maximum a posteriori estimate of time to the most common ancestor for the given taxon pair, while the position along the y -axis is given by the time of the most recent common ancestor of the pair in the given tree. The line shown in blue is $y = x$. The points are horizontally stratified as a single ancestral node in the tree may be the common ancestor of many pairs of taxa. The example here show a very good fit of data and model.

at the command line.

Any onscreen output will be dumped to outlog while the normal `.nex`, `.txt` and `.par` output files will be created in the location specified in `batchtlinput.txt`. For long runs, consider adding `nohup` as a prefix to the above command.

To run the batch version from the Matlab command line, make the main TraitLab directory the current directory and type `batchTraitLab`.

References

- Charles J. Geyer. Practical markov chain monte carlo. *Statistical Science*, 7(4): 473–483, November 1992.
- D. R. Maddison, D. L. Swofford, and W. P. Maddison. Nexus: An extensible file format for systematic information. *Systematic Biology*, 46(4):590–621, 1997.
- Geoff K. Nicholls. *Physics 707 Inverse Problems Lecture Notes*, chapter 9. University of Auckland, 1998. URL <http://www.math.auckland.ac.nz/~phy707/>.
- Geoff K. Nicholls and Russell D. Gray. Dated ancestral trees from binary trait data. *Journal of the royal statistical society series B*, 2007. to appear.