

# Package: rphylo (via r-universe)

June 23, 2026

**Type** Package

**Title** Phylogenetic Analysis with Dependent Discrete Models

**Version** 0.1.1

**Description** Implementation of dependent discrete models (with reversible jump MCMC) derived from 'BayesTraits' V5.0.3  
<<https://github.com/AndrewPMeade/BayesTraits-Release/tree/Release>>. Original software copyright Andrew Meade and contributors, distributed under GPL-3. Modifications for this package by Vivian G. Li <[liguo.vivian@gmail.com](mailto:liguo.vivian@gmail.com)>. The following articles should be referenced when using this package: Pagel, M., A. Meade and D. Barker (2004) ``Bayesian estimation of ancestral character states on phylogenies" <[doi:10.1080/10635150490522232](https://doi.org/10.1080/10635150490522232)>; Pagel, M. (1994) ``Detecting correlated evolution on phylogenies: a general method for the comparative analysis of discrete characters" <[doi:10.1098/rspb.1994.0006](https://doi.org/10.1098/rspb.1994.0006)>; Pagel, M. and A. Meade (2006) ``Bayesian analysis of correlated evolution of discrete characters by reversible-jump Markov chain Monte Carlo" <[doi:10.1086/503444](https://doi.org/10.1086/503444)>.

**License** GPL-3

**Encoding** UTF-8

**Imports** ape, phangorn, expm, stats

**LazyData** true

**Config/roxygen2/version** 8.0.0

**Depends** R (>= 3.5)

**NeedsCompilation** no

**Author** Vivian G. Li [aut, cre], Andrew Meade [ctb] (Original 'BayesTraits' implementation), Mark Pagel [ctb] (Original 'BayesTraits' implementation)

**Maintainer** Vivian G. Li <[liguo.vivian@gmail.com](mailto:liguo.vivian@gmail.com)>

**Repository** <https://cran.r-universe.dev>

**Date/Publication** 2026-06-23 15:28:37 UTC

**RemoteUrl** <https://github.com/cran/rphylo>

**RemoteRef** HEAD

**RemoteSha** 4fe9a951b652e24d1199c6724fc0b77d6fcd0629

## Contents

get_emp_freq . . . . .	2
print.dep_model . . . . .	3
run_dep_model . . . . .	4
scale_trees . . . . .	6
tdata . . . . .	6
tdata2 . . . . .	7
ttree . . . . .	7
ttree2 . . . . .	8
<b>Index</b>	<b>9</b>

---

get_emp_freq	<i>Derive empirical frequencies of states</i>
--------------	---

---

### Description

Compute empirical frequencies of compound states

### Usage

```
get_emp_freq(charfile)
```

### Arguments

charfile      data.frame with taxa as rownames, two binary trait columns

### Value

A numeric vector of length 4 giving the relative frequencies of compound states '00', '01', '10', and '11'.

### Examples

```
get_emp_freq(tdata2)
```

---

print.dep_model	<i>Print method for dep_model objects</i>
-----------------	---

---

**Description**

Nicely formats MCMC output for dep\_model objects.

**Usage**

```
## S3 method for class 'dep_model'  
print(x, ...)
```

**Arguments**

x	An object of class 'dep_model'.
...	additional arguments (ignored)

**Value**

The object 'x', invisibly.

**Examples**

```
fit <- run_dep_model(  
  ttree2,  
  charfile = tdata2,  
  burnin    = 5,  
  iterations = 50,  
  sample_freq = 10,  
  prior     = list(type = "exponential", mean=10),  
  hp        = NULL,  
  revjump   = FALSE,  
  recon_nodes = NULL,  
  tags      = NULL,  
  pis       = c(1,1,1,1),  
  seed      = 42  
)  
  
fit # triggers print.dep_model()
```

---

run\_dep\_model                      *Run a dependent discrete-character model*

---

### Description

Performs Bayesian inference of correlated evolution between two binary traits, on one or more phylogenetic trees using MCMC or RJMCMC.

### Usage

```
run_dep_model(
  trees,
  charfile,
  burnin,
  iterations,
  sample_freq = 1000L,
  tags = NULL,
  recon_nodes = NULL,
  revjump = FALSE,
  hp = NULL,
  hp_dev = HP_DEV_DEFAULT,
  prior = list(type = "exponential", mean = 10),
  pis = c(1, 1, 1, 1),
  seed = NULL,
  verbose = TRUE
)
```

### Arguments

trees	multiPhylo object (trees should already be scaled if desired)
charfile	data.frame with taxa as rownames, two binary trait columns
burnin	integer number of iterations to discard before sampling
iterations	integer number of post-burnin iterations
sample_freq	integer record one sample every sample_freq iterations
tags	named list of character vectors defining nodes by their descendant taxa. Example: list(root_node = c("Taxon1", "Taxon2", "Taxon3"))
recon_nodes	character vector of tag names to reconstruct ancestral states for. Must be names present in tags. Set to NULL to skip ASR.
revjump	logical: use RJMCMC (default FALSE)
hp	hyperprior bounds list or NULL. If non-NULL, the parameters of 'prior' are themselves estimated. Format depends on prior type: exponential: list(mean = c(min, max)); uniform: list(min = c(min, max), max = c(min, max)); gamma: list(shape = c(min, max), scale = c(min, max)); lognormal: list(location = c(min, max), scale = c(min, max)).

hp_dev	numeric: proposal SD for hyperprior moves (default 1.0)
prior	list specifying prior on rate values. Supported forms include: list(type="exponential", mean=10); list(type="uniform", min=0, max=100)
pis	numeric vector of length 4 giving root state prior weights for states 00, 01, 10, 11; or "stationary". Weights need not sum to 1 (they are normalised internally). Default: c(1,1,1,1) matches 'pis none' in BayesTraits. Use uniform distribution: c(0.25,0.25,0.25,0.25). Use empirical distribution: get_emp_freq(charfile). Use stationary distribution: "stationary".
seed	integer random seed (NULL = random)
verbose	logical print progress (default TRUE)

**Value**

object of class "dep\_model" containing: \$log\_lh numeric vector of sampled log-likelihoods;  
 \$rates data.frame of sampled full rate vectors (8 columns);  
 \$n\_classes integer vector of sampled rate class counts;  
 \$mapping list of sampled mapping vectors (mapping between rate values and rate classes);  
 \$prior\_samples data.frame of sampled prior parameters (NULL if no hp);  
 \$anc named list of ASR matrices [n\_samples x 4] per recon node, or NULL if recon\_nodes = NULL.  
 NA rows indicate the node was not monophyletic on that tree.  
 \$settings list of run settings.

**Examples**

```
tree <- ttree2
chars <- tdata2

fit <- run_dep_model(ttree2,
  charfile = tdata2,
  burnin = 10,
  iterations = 30,
  sample_freq = 10,
  prior = list(type = "exponential", mean=10),
  hp = NULL,
  revjump = TRUE,
  recon_nodes = c("root_node"),
  tags = list(root_node = ttree2[[1]]$tip.label),
  pis = get_emp_freq(tdata2),
  seed = 42
)
```

scale\_trees                      *Scale branch lengths of phylogenetic trees*

---

**Description**

Rescales all branch lengths in a ‘multiPhylo’ object so that the mean branch length across all trees equals ‘target\_mean\_bl’.

**Usage**

```
scale_trees(multiphylo, target_mean_bl = 0.1)
```

**Arguments**

**multiphylo**      An object of class ‘multiPhylo’.  
**target\_mean\_bl** target mean branch length. Default: 0.1.

**Value**

A list with:

**trees** A ‘multiPhylo’ object with rescaled branch lengths.  
**scalar** The scalar multiplier applied to all branch lengths.

**Examples**

```
scale_trees(ttree2)
```

---

tdata                              *Example trait dataset*

---

**Description**

Binary trait data corresponding to ‘tree’.

**Usage**

```
tdata
```

**Format**

A data frame with taxa as row names and 2 binary trait columns.

**Source**

Simulated data.

---

tdata2	<i>Example trait dataset (60 taxa)</i>
--------	--

---

**Description**

Binary trait data corresponding to 'tree2'.

**Usage**

```
tdata2
```

**Format**

A data frame with taxa as row names and two binary trait columns.

**Source**

Simulated data.

---

ttree	<i>Example phylogenetic tree</i>
-------	----------------------------------

---

**Description**

A small phylogenetic tree for demonstrating rphylo analyses.

**Usage**

```
ttree
```

**Format**

An object of class 'phylo'.

**Source**

Simulated data.

---

`ttree2`*Example phylogenetic trees (500 trees)*

---

**Description**

500 phylogenetic trees for demonstrating rphylo analyses.

**Usage**

```
ttree2
```

**Format**

An object of class 'multiPhylo' containing 500 trees.

**Source**

Simulated data.

# Index

## \* datasets

tdata, 6

tdata2, 7

ttree, 7

ttree2, 8

get\_emp\_freq, 2

print.dep\_model, 3

run\_dep\_model, 4

scale\_trees, 6

tdata, 6

tdata2, 7

ttree, 7

ttree2, 8