

# Package ‘POUMM’

October 27, 2020

**Type** Package

**Title** The Phylogenetic Ornstein-Uhlenbeck Mixed Model

**Version** 2.1.7

**Date** 2020-10-20

**Maintainer** Venelin Mitov <[vmitov@gmail.com](mailto:vmitov@gmail.com)>

**Description** The Phylogenetic Ornstein-Uhlenbeck Mixed Model (POUMM) allows to estimate the phylogenetic heritability of continuous traits, to test hypotheses of neutral evolution versus stabilizing selection, to quantify the strength of stabilizing selection, to estimate measurement error and to make predictions about the evolution of a phenotype and phenotypic variation in a population. The package implements combined maximum likelihood and Bayesian inference of the univariate Phylogenetic Ornstein-Uhlenbeck Mixed Model, fast parallel likelihood calculation, maximum likelihood inference of the genotypic values at the tips, functions for summarizing and plotting traces and posterior samples, functions for simulation of a univariate continuous trait evolution model along a phylogenetic tree. So far, the package has been used for estimating the heritability of quantitative traits in macroevolutionary and epidemiological studies, see e.g. Bertels et al. (2017) <[doi:10.1093/molbev/msx246](https://doi.org/10.1093/molbev/msx246)> and Mitov and Stadler (2018) <[doi:10.1093/molbev/msx328](https://doi.org/10.1093/molbev/msx328)>. The algorithm for parallel POUMM likelihood calculation has been published in Mitov and Stadler (2019) <[doi:10.1111/2041-210X.13136](https://doi.org/10.1111/2041-210X.13136)>.

**License** GPL (>= 3.0)

**Encoding** UTF-8

**LazyData** true

**Depends** R (>= 3.1.0), stats, Rcpp, methods

**LinkingTo** Rcpp

**Imports** ape, data.table(>= 1.13.2), coda, foreach, ggplot2, lamW, adaptMCMC, utils

**Suggests** testthat, usethis, Rmpfr, mvtnorm, lmtest, knitr, rmarkdown, parallel, doParallel

**RoxygenNote** 7.1.0

**VignetteBuilder** knitr, rmarkdown

**ByteCompile** no

**NeedsCompilation** yes

**URL** <https://venelin.github.io/POUMM/index.html>,  
<https://github.com/venelin/POUMM>

**BugReports** <https://github.com/venelin/POUMM/issues>

**Author** Venelin Mitov [aut, cre, cph] (<a href="https://venelin.github.io">venelin.github.io</a>),  
 Matt Dowle [ctb]

**Repository** CRAN

**Date/Publication** 2020-10-27 14:00:18 UTC

## R topics documented:

analyseMCMCs . . . . .	3
chld . . . . .	4
coef.POUMM . . . . .	4
covFunPOUMM . . . . .	5
covHPDFunPOUMM . . . . .	5
covPOUMM . . . . .	6
covVTipsGivenTreePOUMM . . . . .	7
dVNodesGivenTreePOUMM . . . . .	8
edgesFrom . . . . .	9
fitted.POUMM . . . . .	9
gPOUMM . . . . .	10
H2 . . . . .	10
likPOUMMGivenTreeVTips . . . . .	11
likPOUMMGivenTreeVTipsC . . . . .	12
logLik.POUMM . . . . .	14
loglik_abc_g0_g0Prior . . . . .	15
maxLikPOUMMGivenTreeVTips . . . . .	15
mcmcPOUMMGivenPriorTreeVTips . . . . .	16
nobs.POUMM . . . . .	18
nodeTimes . . . . .	18
OU . . . . .	19
PhylogeneticH2 . . . . .	20
plot.POUMM . . . . .	22
plot.summary.POUMM . . . . .	24
POUMM . . . . .	26
POUMMIIsADevRelease . . . . .	28
pruneTree . . . . .	29
residuals.POUMM . . . . .	29
rTrajectoryOU . . . . .	30
rTrajectoryOUDef . . . . .	31

<i>analyseMCMCs</i>	3
rVNodesGivenTreePOUMM . . . . .	31
simulatePOUMMLikelihoodMainLoop . . . . .	32
simulateTrait . . . . .	33
specPOUMM . . . . .	33
statistics . . . . .	44
summary.POUMM . . . . .	44
validateSpecPOUMM . . . . .	45
validateZTree . . . . .	46
vignetteCachedResults . . . . .	46
<b>Index</b>	<b>47</b>

---

<i>analyseMCMCs</i>	<i>Extract data from an MCMC chain This is an internal function.</i>
---------------------	--

---

## Description

Extract data from an MCMC chain This is an internal function.

## Usage

```
analyseMCMCs(
  chains,
  stat = NULL,
  statName = "logpost",
  start,
  end,
  thinMCMC,
  as.dt = FALSE,
  k = NA,
  N = NA,
  ...
)
```

## Arguments

chains, stat, statName, start, end, thinMCMC, as.dt, k, N, ...  
internal use.

---

chld	<i>Node indices of the direct descendants of n in the phylogeny.</i>
------	--

---

**Description**

Node indices of the direct descendants of n in the phylogeny.

**Usage**

```
chld(tree, n)
```

**Arguments**

tree	an object of class phylo
n	an index of a node (root, internal or tip) in tree

**Value**

An integer vector.

---

coef.POUMM	<i>Extract maximum likelihood fitted parameters (coefficients) from a fitted POUMM model.</i>
------------	---

---

**Description**

Extract maximum likelihood fitted parameters (coefficients) from a fitted POUMM model.

**Usage**

```
## S3 method for class 'POUMM'
coef(object, mapped = FALSE, ...)
```

**Arguments**

object	An object of class POUMM.
mapped	Logical indicating whether the standard POUMM parameters should also be extracted.
...	Not used; added for compatibility with generic function coef.

**Details**

The parameters extracted are the ones used as input to the model's parMapping function.

**Value**

A named vector with the fitted parameters of the model.

---

covFunPOUMM	<i>A vectorized expected covariance function for a given tree and a fitted POUMM model</i>
-------------	--

---

**Description**

A vectorized expected covariance function for a given tree and a fitted POUMM model

**Usage**

```
covFunPOUMM(object, corr = FALSE)
```

**Arguments**

object	an S3 object of class POUMM
corr	logical indicating if an expected correlation function should be returned For non-ultrametric trees, usually the mean root-tip distance is used.

**Value**

a function of three numerical parameters: tau - the phylogenetic distance between a two tips; tanc - the distance from the root to their most recent common ancestor. t - the root-tip distance (assuming that the two tips are at equal distance from the root)

---

covHPDFunPOUMM	<i>A vectorized function returning HPD intervals of the expected covariance for a given tree and a fitted POUMM model</i>
----------------	---

---

**Description**

A vectorized function returning HPD intervals of the expected covariance for a given tree and a fitted POUMM model

**Usage**

```
covHPDFunPOUMM(object, prob = 0.95, corr = FALSE, ...)
```

**Arguments**

object	an S3 object of class POUMM
prob	a Numerical between 0 and 1
corr	logical indicating if an expected correlation HPD interval function should be returned.
...	additional parameters passed to summary.POUMM

**Value**

a function of a numerical matrix  $x$  with 3 columns corresponding to  $\tau$ ,  $t_{\text{anc}}$  and  $t$  (see covFunPOUMM). The function returns a two-column matrix with the lower and upper limit of the HPD for each row in the input matrix.

---

covPOUMM	<i>Expected covariance of two tips at given root-tip time and phylogenetic distance</i>
----------	---

---

**Description**

Expected covariance of two tips at given root-tip time and phylogenetic distance

**Usage**

```
covPOUMM(
  alpha,
  sigma,
  sigmae,
  t,
  tau,
  tanc = t - tau/2,
  corr = FALSE,
  as.matrix = FALSE
)
```

**Arguments**

$\alpha$ , $\sigma$ , $\sigma_e$	POUMM parameters
$t$	A non-negative number or vector time from the beginning of the POUMM process (root-tip distance). If a vector, the evaluation is done on each couple (row) from <code>cbind(t, tau)</code> .
$\tau$	A non-negative number or vector indicating the phylogenetic distance between two tips, each of them located at time $t$ from the root. If a vector, the evaluation is done on each couple (row) from <code>cbind(t, tau)</code> .
$t_{\text{anc}}$	A non-negative number or vector indicating the root-mrca distance for a couple of tips. Defaults to $t - \tau/2$ corresponding to an ultrametric tree.
<code>corr</code>	Logical indicating whether correlation should be returned instead of covariance.
<code>as.matrix</code>	Logical indicating if a variance-covariance matrix should be returned.

**Details**

The function assumes that the two tips are at equal distance  $t$  from the root. This implies that the root-tip distance of their mrca is  $t - \tau/2$ .

**Value**

If `as.matrix == FALSE`, a number. Otherwise a two by two symmetric matrix. If `t` or `tau` is a vector of length bigger than 1, then a vector of numbers or a list of matrices.

---

covVTipsGivenTreePOUMM

*Variance covariance matrix of the values at the tips of a tree under an OU process*

---

**Description**

Variance covariance matrix of the values at the tips of a tree under an OU process

**Usage**

```
covVTipsGivenTreePOUMM(
  tree,
  alpha = 0,
  sigma = 1,
  sigmae = 0,
  tanc = NULL,
  tauij = NULL,
  corr = FALSE
)
```

**Arguments**

<code>tree</code>	A phylo object.
<code>alpha</code> , <code>sigma</code>	Non-negative numeric values, parameters of the OU process.
<code>sigmae</code>	Non-negative numeric value, environmental standard deviation at the tips.
<code>tanc</code>	Numerical matrix with the time-distance from the root of the tree to the mrca of each tip-couple. If NULL it will be calculated.
<code>tauij</code>	Numerical matrix with the time (patristic) distance between each pair of tips. If NULL, it will be calculated.
<code>corr</code>	Logical indicating if a correlation matrix shall be returned.

**Value**

a variance covariance or a correlation matrix of the tips in tree.

**References**

(Hansen 1997) Stabilizing selection and the comparative analysis of adaptation.

---

dVNodesGivenTreePOUMM *Multivariate density of observed values along a tree given an OU process of evolution and root-value*

---

### Description

Calculates the conditional probability density of observed values at the tips and internal nodes of a tree, given that tree, the value at the root,  $z[N+1]$ , where  $N$  is the number of tips in the tree, known measurement error  $e$  for each value in  $z$ , and a POUMM model of evolution. This function is mostly used to calculate the likelihood of simulated data under known model parameters.

### Usage

```
dVNodesGivenTreePOUMM(
  z,
  tree,
  alpha,
  theta,
  sigma,
  sigmae = 0,
  e = rep(0, length(z)),
  log = TRUE
)
```

### Arguments

<code>z</code>	A numeric vector of size $\text{length}(\text{tree}\$tip.label) + \text{tree}\$Nnode$ representing the observed values at the tips, root and internal nodes.
<code>tree</code>	An object of class <code>phylo</code> .
<code>alpha, theta, sigma</code>	Numeric values, parameters of the OU model.
<code>sigmae</code>	Numeric non-negative value or vector of $\text{length}(z)$ elements (default 0). Specifies the standard deviation of random environmental contribution (and eventually measurement error) to be added to the values. Note that if measurement standard error, $se$ , is known and needs to be added to the environmental contribution, the right way to specify the parameter would be $\sqrt{\text{sigmae}^2 + se^2}$ , not $\text{sigmae} + se$ .
<code>e</code>	Numeric vector of size $\text{length}(z)$ representing exactly known error (sum of environmental contribution and measurement error). Defaults to a vector of zeroes.
<code>log</code>	Logical indicating whether a log-likelihood should be returned instead of a likelihood. Default is <code>TRUE</code> .

### Value

A numeric value, the multivariate probability density of  $z$  under the given parameters.



---

edgesFrom	<i>Edge indices of the edges in tree starting from n</i>
-----------	--

---

**Description**

Edge indices of the edges in tree starting from n

**Usage**

```
edgesFrom(tree, n)
```

**Arguments**

tree	an object of class phylo
n	an index of a node (root, internal or tip) in tree

**Value**

An integer vector.

---

fitted.POUMM	<i>Extract maximum likelihood expected genotypic values at the tips of a tree, to which a POUMM model has been previously fitted</i>
--------------	--

---

**Description**

Extract maximum likelihood expected genotypic values at the tips of a tree, to which a POUMM model has been previously fitted

**Usage**

```
## S3 method for class 'POUMM'
fitted(object, vCov = FALSE, ...)
```

**Arguments**

object	An object of class POUMM.
vCov	A logical indicating whether a list with the genotypic values and their variance covariance matrix should be returned or only a vector of the genotypic values (default is FALSE).
...	Not used; added for compatibility with generic function fitted.

**Value**

If vCov == TRUE, a list with elements g - the genotypic values and vCov - the variance-covariance matrix of these values for the specific tree, observed values z and POUMM ML-fit. If vCov == FALSE, only the vector of genotypic values corresponding to the tip-labels in the tree is returned.

---

 gPOUMM

*Distribution of the genotypic values under a POUMM fit*


---

**Description**

Distribution of the genotypic values under a POUMM fit

**Usage**

```
gPOUMM(z, tree, g0, alpha, theta, sigma, sigmae)
```

**Arguments**

z	A numeric vector of size <code>length(tree\$tip.label)</code> representing the trait values at the tip-nodes.
tree	an object of class <code>phylo</code>
g0	A numeric value at the root of the tree, genotypic contribution.
alpha, theta, sigma	Numeric values, parameters of the OU model.
sigmae	Numeric non-negative value (default 0). Specifies the standard deviation of random environmental contribution (white noise) included in z.

**Value**

a list with elements `V.g`, `V.g_1`, `mu.g`, `V.e`, `V.e_1`, `mu.e`, `V.g.poumm`, `mu.g.poumm`.

---

 H2

*Phylogenetic heritability estimated at time t*


---

**Description**

Phylogenetic heritability estimated at time t

**Usage**

```
H2(alpha, sigma, sigmae, t = Inf, tm = 0)
```

**Arguments**

alpha, sigma	numeric, parameters of the OU process acting on the genetic contributions
sigmae	numeric, environmental standard deviation
t	time from the beginning of the process at which heritability should be calculated, i.e. epidemiologic time
tm	average time for within host evolution from getting infected until getting measured or passing on the infection to another host

---

 likPOUMMGivenTreeVTips

*Density of observed tip-values given a tree, assuming Ornstein-Uhlenbeck process for the genetic contributions along the tree and normally distributed environmental deviations.*

---

### Description

Calculates the (log-)probability density of trait values at the tip-nodes given the tree and assuming that the trait value at the tips is the sum of a genetic contribution,  $g$ , that evolved on the tree according to an OU process with parameters  $\alpha$ ,  $\theta$ ,  $\sigma$  and an environmental deviation,  $e$ , that is distributed normally and independently between the tips of the tree. Note: Without additional assumptions for the distribution of the value at the root of the tree, the likelihood is not defined at  $\alpha=0$ , although this corresponds to the limiting Brownian motion process with mean value  $\theta$  and unit time variance  $\sigma^2$ . Considering the observed data and tree as a fixed parameter and the POUMM parameters as variables, this function is interpreted as the POUMM likelihood.

### Usage

```
likPOUMMGivenTreeVTips(
  z,
  tree,
  alpha,
  theta,
  sigma,
  sigmae = 0,
  g0 = NA,
  g0Prior = NULL,
  log = TRUE,
  pruneInfo = pruneTree(tree, z),
  usempfr = 0,
  maxmpfr = 2,
  precbits = 128,
  debug = FALSE
)
```

### Arguments

<code>z</code>	A numeric vector of size <code>length(tree\$tip.label)</code> representing the trait values at the tip-nodes.
<code>tree</code>	an object of class <code>phylo</code>
<code>alpha</code>	the strength of the selection
<code>theta</code>	long term mean value of the OU process
<code>sigma</code>	the unit-time standard deviation of the random component in the OU process.
<code>sigmae</code>	the standard deviation of the environmental deviation added to the genetic contribution at each tip, by default 0, meaning no environmental deviation.

<code>g0</code>	Numeric, NA or NaN, either a fixed genotypic value at the root of tree or NA or NaN. A NA "Not Available" will cause to analytically calculate the value of <code>g0</code> that would maximize the conditional likelihood of the data given <code>g0</code> . A NaN "Not a Number" will cause integration over <code>g0</code> taking values in $(-\text{Inf}, +\text{Inf})$ assuming that <code>g0</code> is normally distributed with mean <code>g0Prior\$mean</code> and variance <code>g0Prior\$var</code> (see parameter <code>g0Prior</code> ).
<code>g0Prior</code>	Either NULL or a list with named numeric or character members "mean" and "var". Specifies a prior normal distribution for the parameter <code>g0</code> . If characters, the members mean and var are evaluated as R-expressions.
<code>log</code>	Logical indicating whether log-likelihood should be returned instead of likelihood, default is TRUE.
<code>pruneInfo</code>	list returned by <code>pruneTree(tree)</code> to be passed in explicit calls to <code>dVGivenTreeOU</code> .
<code>usempfr</code>	integer indicating if and how mpfr should be used for small parameter values ( $\text{any}(c(\text{alpha}, \text{sigma}, \text{sigmae}) < 0.01)$ ). Using the mpfr package can be forced by specifying an integer greater or equal to 2. Setting <code>usempfr=0</code> (default) causes high precision likelihood calculation to be done on each encounter of parameters with at least 1 bigger log-likelihood value than any of the currently found maximum log-likelihood or the previously calculated log-likelihood value Requires the Rmpfr package. Note that using mpfr may increase the time for one likelihood calculation more than 100-fold. Set <code>usempfr</code> to -1 or less to completely disable Rmpfr functionality.
<code>maxmpfr</code>	integer (not used)
<code>precbits</code>	integer specifying precision bits for mpfr. Default is 512.
<code>debug</code>	logical, if set to TRUE some debugging information is printed during likelihood calculation

**Value**

A numeric with attributes "g0" and "g0LogPrior".

---

likPOUMMGivenTreeVTipsC

*Fast (parallel) POUMM likelihood calculation using the SPLITT library*

---

**Description**

Fast (log-)likelihood calculation using C++ and OpenMP based parallelization.

**Usage**

```
likPOUMMGivenTreeVTipsC(
  integrator,
  alpha,
  theta,
```

```

    sigma,
    sigmae,
    g0 = NA,
    g0Prior = NULL,
    log = TRUE
  )

```

### Arguments

integrator	An Rcpp module object from the class POUMM_AbcPOUMM. This object is to be created using the function pruneTree (see example). This object contains the data and tree (arguments z and tree of the equivalent function dVTipsGivenTreeVTips.).
alpha	the strength of the selection
theta	long term mean value of the OU process
sigma	the unit-time standard deviation of the random component in the OU process.
sigmae	the standard deviation of the environmental deviation added to the genetic contribution at each tip, by default 0, meaning no environmental deviation.
g0	Numeric, NA or NaN, either a fixed genotypic value at the root of tree or NA or NaN. A NA "Not Available" will cause to analytically calculate the value of g0 that would maximize the conditional likelihood of the data given g0. A NaN "Not a Number" will cause integration over g0 taking values in (-Inf,+Inf) assuming that g0 is normally distributed with mean g0Prior\$mean and variance g0Prior\$var (see parameter g0Prior).
g0Prior	Either NULL or a list with named numeric or character members "mean" and "var". Specifies a prior normal distribution for the parameter g0. If characters, the members mean and var are evaluated as R-expressions.
log	Logical indicating whether log-likelihood should be returned instead of likelihood, default is TRUE.

### Details

This function is the C++ equivalent of dVTipsGivenTreePOUMM (aliased also as likPOUMM-GivenTreeVTips). Currently, the function does not support multiple precision floating point operations (supported in dVTipsGivenTreePOUMM). The C++ implementation is based on the library for parallel tree traversal "SPLITT" (<https://github.com/venelin/SPLITT.git>).

### Value

A numeric with attributes "g0" and "g0LogPrior".

### References

Mitov, V., and Stadler, T. (2017). Fast and Robust Inference of Phylogenetic Ornstein-Uhlenbeck Models Using Parallel Likelihood Calculation. *bioRxiv*, 115089. <https://doi.org/10.1101/115089>

Mitov, V., & Stadler, T. (2017). Fast Bayesian Inference of Phylogenetic Models Using Parallel Likelihood Calculation and Adaptive Metropolis Sampling. *Systematic Biology*, 235739. <http://doi.org/10.1101/235739>

**See Also**

dVTipsGivenTreePOUMM

**Examples**

```
## Not run:
N <- 100
tr <- ape::rtree(N)
z <- rVNodesGivenTreePOUMM(tr, 0, 2, 3, 1, 1)[1:N]
pruneInfo <- pruneTree(tr, z)
microbenchmark::microbenchmark(
  likCpp <- likPOUMMGivenTreeVTipsC(pruneInfo$integrator, 2, 3, 1, 1),
  likR <- likPOUMMGivenTreeVTips(z, tr, 2, 3, 1, 1, pruneInfo = pruneInfo))

# should be the same values
likCpp
likR

## End(Not run)
```

---

logLik.POUMM	<i>Extract maximum likelihood and degrees of freedom from a fitted POUMM model</i>
--------------	--

---

**Description**

Extract maximum likelihood and degrees of freedom from a fitted POUMM model

**Usage**

```
## S3 method for class 'POUMM'
logLik(object, ...)
```

**Arguments**

object	An object of class POUMM.
...	not used; included for compliance with generic function logLik.

---

loglik\_abc\_g0\_g0Prior *Processing of the root value and calculation of the maximum log-likelihood for the given coefficients abc, and parameters theta, g0 and g0Prior. This is an internal function.*

---

### Description

Processing of the root value and calculation of the maximum log-likelihood for the given coefficients abc, and parameters theta, g0 and g0Prior. This is an internal function.

### Usage

```
loglik_abc_g0_g0Prior(abc, alpha, theta, sigma, g0, g0Prior)
```

### Arguments

abc	a vector of 3 numerical values denoting the corresponding coefficients in the POUMM likelihood presented as $\exp(a g_0^2 + b g_0 + c)$ .
alpha, theta, sigma	parameters of the OU-process.
g0	initial value at the root of the tree (can be NA). See argument parMapping in ?specifyPOUMM.
g0Prior	list object. See parameter g0Prior in ?specifyPOUMM.

---

maxLikPOUMMGivenTreeVTips

*Find a maximum likelihood fit of the POUMM model*

---

### Description

Find a maximum likelihood fit of the POUMM model

### Usage

```
maxLikPOUMMGivenTreeVTips(
  loglik,
  pruneInfo,
  parLower,
  parUpper,
  parInitML = NULL,
  control = list(factr = 1e+08, fnscale = -1),
  verbose = FALSE,
  debug = FALSE,
  ...
)
```

**Arguments**

loglik	function(par, memo, parFixedNoAlpha)
pruneInfo	a list-object returned by the pruneTree(tree, z) function.
parLower, parUpper	Two named numeric vectors indicating the boundaries of the search region. Default values are parLower = c(alpha = 0, theta = 0, sigma = 0, sigmae = 0) and parUpper = c(alpha = 100, theta = 10, sigma = 20, sigmae = 10).
parInitML	A named vector (like parLower and parUpper) or a list of such vectors - starting points for optim.
control	List of parameters passed on to optim, default list(factr = 1e8, fnscale = -1), see ?optim.
verbose	A logical indicating whether to print informative messages on the standard output.
debug	A logical indicating whether to print debug messages (currently not implemented).
...	currently not used.

**Value**

a list containing an element par and an element value as well as the parameters passed

---

mcmcPOUMMGivenPriorTreeVTips

*MCMC-sampling from a posterior distribution of a P(OU)MM model given tree, values at the tips and a prior distribution*

---

**Description**

MCMC-sampling from a posterior distribution of a P(OU)MM model given tree, values at the tips and a prior distribution

**Usage**

```
mcmcPOUMMGivenPriorTreeVTips(
  loglik,
  fitML = NULL,
  parMapping,
  parInitMCMC,
  parPriorMCMC,
  parScaleMCMC,
  nSamplesMCMC,
  nAdaptMCMC,
  thinMCMC,
  accRateMCMC,
  gammaMCMC,
```



```

    nChainsMCMC,
    samplePriorMCMC,
    pruneInfo,
    ...,
    verbose = FALSE,
    parallelMCMC = FALSE
  )

```

### Arguments

loglik	a log-likelihood function.
fitML	an object returned by the maxLikPOUMMGivenTreeVTips
parMapping	a function(numeric-vector) transforming a sampled vector on the scale of the parameters alpha, theta, sigma, sigmae and g0.
parInitMCMC	a function(chainNumber) returning the starting point of the MCMC as a vector.
parPriorMCMC	a function(numeric-vector) returning the log-prior of the supplied vector
parScaleMCMC	numeric matrix indicating the initial jump-distribution matrix
nSamplesMCMC	integer indicating how many iterations should the mcmc-chain contain
nAdaptMCMC	integer indicating how many initial iterations should be used for adaptation of the jump-distribution matrix
thinMCMC	integer indicating the thinning interval of the mcmc-chain
accRateMCMC	(MCMC) numeric between 0 and 1 indicating the target acceptance rate Passed on to adaptMCMC::MCMC.
gammaMCMC	(MCMC) controls the speed of adaption. Should be in the interval (0.5,1]. A lower gammaMCMC leads to faster adaption. Passed on to adaptMCMC::MCMC.
nChainsMCMC	integer indicating the number of chains to run. Defaults to 1.
samplePriorMCMC	logical indicating if only the prior distribution should be sampled. This can be useful to compare with mcmc-runs for an overlap between prior and posterior distributions.
pruneInfo	a list-object returned from the pruneTree(tree, z) function.
...	Additional arguments. Currently not used except for the following: If ... includes debug = TRUE, some debug messages will be written also outside of the call to loglik.
verbose	Logical indicating if some informal messages should be written during run. This parameter is passed to loglik.
parallelMCMC	Logical indicating if chains should be run in parallel.

### Details

Currently, this function calls the MCMC function from the adaptMCMC package.

### Value

a list of coda objects

---

nobs.POUMM	<i>Number of tips in a phylogenetic tree, POUMM has been fit on.</i>
------------	--

---

**Description**

Number of tips in a phylogenetic tree, POUMM has been fit on.

**Usage**

```
## S3 method for class 'POUMM'
nobs(object, ...)
```

**Arguments**

object	An object of class POUMM.
...	Not used; added for compatibility with generic function nobs.

**Value**

The number of tips in the tree, POUMM has been called on

---

nodeTimes	<i>Calculate the time from the root to each node of the tree</i>
-----------	--

---

**Description**

Calculate the time from the root to each node of the tree

**Usage**

```
nodeTimes(tree, tipsOnly = FALSE)
```

**Arguments**

tree	An object of class phylo.
tipsOnly	Logical indicating whether the returned results should be truncated only to the tips of the tree.

**Value**

A vector of size the number of nodes in the tree (tips, root, internal) containing the time from the root to the corresponding node in the tree.

---

OU	<i>Distribution of an Ornstein-Uhlenbeck Process at Time t, Given Initial State at Time 0</i>
----	---

---

### Description

An Ornstein-Uhlenbeck (OU) process represents a continuous time Markov chain parameterized by an initial state  $x_0$ , selection strength  $\alpha > 0$ , long-term mean  $\theta$ , and time-unit variance  $\sigma^2$ . Given  $x_0$ , at time  $t$ , the state of the process is characterized by a normal distribution with mean  $x_0 \exp(-\alpha t) + \theta(1 - \exp(-\alpha t))$  and variance  $\sigma^2(1 - \exp(-2\alpha t))/(2\alpha)$ . In the limit  $\alpha \rightarrow 0$ , the OU process converges to a Brownian motion process with initial state  $x_0$  and time-unit variance  $\sigma^2$  (at time  $t$ , this process is characterized by a normal distribution with mean  $x_0$  and variance  $t\sigma^2$ ).

### Usage

`dOU(z, z0, t, alpha, theta, sigma, log = TRUE)`

`rOU(n, z0, t, alpha, theta, sigma)`

`meanOU(z0, t, alpha, theta)`

`varOU(t, alpha, sigma)`

`sdOU(t, alpha, sigma)`

### Arguments

<code>z</code>	Numeric value or vector of size n.
<code>z0</code>	Numeric value or vector of size n, initial value(s) to condition on.
<code>t</code>	Numeric value or vector of size n, denoting the time-step.
<code>alpha, theta, sigma</code>	Numeric values or n-vectors, parameters of the OU process; alpha and sigma must be non-negative. A zero alpha is interpreted as the Brownian motion process in the limit $\alpha \rightarrow 0$ .
<code>log</code>	Logical indicating whether the returned density should be on the logarithmic scale.
<code>n</code>	Integer, the number of values to sample.

### Details

Similar to `dnorm` and `rnorm`, the functions described in this help-page support single values as well as vectors for the parameters `z`, `z0`, `t`, `alpha`, `theta` and `sigma`.

**Value**

dOU returns the conditional probability density(ies) of the elements in z, given the initial state(s) z0, time-step(s) t and OU-parameters by alpha, theta and sigma.

rOU returns a numeric vector of length n, a random sample from the conditional distribution(s) of one or n OU process(es) given initial value(s) and time-step(s).

meanOU returns the expected value of the OU-process at time t.

varOU returns the expected variance of the OU-process at time t.

sdOU returns the standard deviation of the OU-process at time t.

**Functions**

- dOU: probability density
- rOU: random generator
- meanOU: mean value
- varOU: variance
- sdOU: standard deviation

**Examples**

```
z0 <- 8
t <- 10
n <- 100000
sample <- rOU(n, z0, t, 2, 3, 1)
dens <- dOU(sample, z0, t, 2, 3, 1)
var(sample) # around 1/4
varOU(t, 2, 1)
```

**Description**

The phylogenetic heritability,  $H^2$ , is defined as the ratio of the genetic variance over the total phenotypic variance expected at a given evolutionary time t (measured from the root of the tree). Thus, the phylogenetic heritability connects the parameters alpha, sigma and sigmae of the POUMM model through a set of equations. The functions described here provide an R-implementation of these equations.

**Usage**

```
alpha(H2, sigma, sigmae, t = Inf)

sigmaOU(H2, alpha, sigmae, t = Inf)

sigmae(H2, alpha, sigma, t = Inf)

H2e(z, sigmae, tree = NULL, tFrom = 0, tTo = Inf)
```

**Arguments**

H2	Phylogenetic heritability at time t.
sigmae	Numeric, environmental phenotypic deviation at the tips.
t	Numeric value denoting evolutionary time (i.e. distance from the root of a phylogenetic tree).
alpha, sigma	Numeric values or n-vectors, parameters of the OU process; alpha and sigma must be non-negative. A zero alpha is interpreted as the Brownian motion process in the limit $\alpha \rightarrow 0$ .
z	Numerical vector of observed phenotypes.
tree	A phylo object.
tFrom, tTo	Numerical minimal and maximal root-tip distance to limit the calculation.

**Details**

The function sigmae uses the formula  $H2 = \text{varOU}(t, \alpha, \sigma) / (\text{varOU}(t, \alpha, \sigma) + \sigmae^2)$

**Value**

All functions return numerical values or NA, in case of invalid parameters

**Functions**

- alpha: Calculate alpha given time t, H2, sigma and sigmae
- sigmaOU: Calculate sigma given time t, H2 at time t, alpha and sigmae
- sigmae: Calculate sigmae given alpha, sigma, and H2 at time t
- H2e: "Empirical" phylogenetic heritability estimated from the empirical variance of the observed phenotypes and sigmae

**Note**

This function is called sigmaOU and not simply sigma to avoid a conflict with a function sigma in the base R-package.

**See Also**

OU

**Examples**

```

# At POUMM stationary state (equilibrium, t=Inf)
H2 <- H2(alpha = 0.75, sigma = 1, sigmae = 1, t = Inf)      # 0.4
alpha <- alpha(H2 = H2, sigma = 1, sigmae = 1, t = Inf)   # 0.75
sigma <- sigmaOU(H2 = H2, alpha = 0.75, sigmae = 1, t = Inf) # 1
sigmae <- sigmae(H2 = H2, alpha = 0.75, sigma = 1, t = Inf) # 1

# At finite time t = 0.2
H2 <- H2(alpha = 0.75, sigma = 1, sigmae = 1, t = 0.2)    # 0.1473309
alpha <- alpha(H2 = H2, sigma = 1, sigmae = 1, t = 0.2)   # 0.75
sigma <- sigmaOU(H2 = H2, alpha = 0.75, sigmae = 1, t = 0.2) # 1
sigmae <- sigmae(H2 = H2, alpha = 0.75, sigma = 1, t = 0.2) # 1

```

---

plot.POUMM

*Plots of a POUMM-fit*


---

**Description**

Plots of a POUMM-fit

**Usage**

```

## S3 method for class 'POUMM'
plot(
  x,
  type = c("MCMC"),
  doPlot = TRUE,
  interactive = TRUE,
  stat = c("alpha", "theta", "sigma", "sigmae", "g0", "H2tMean"),
  chain = NULL,
  startMCMC = NA,
  endMCMC = NA,
  thinMCMC = 1000,
  statFunctions = statistics(x),
  doZoomIn = FALSE,
  zoomInFilter = paste0("(stat %in% c('H2e', 'H2tMean', 'H2tInf', 'H2tMax') & ",
    "(value >= 0 & value <= 1) ) |",
    "( !stat %in% c('H2e', 'H2tMean', 'H2tInf', 'H2tMax') & ",
    "(value <= median(HPDUpper) + 4 * (median(HPDUpper) - median(HPDLow)) & ",
    "value >= median(HPDLow) - 4 * (median(HPDUpper) - median(HPDLow))))"),
  prettyNames = TRUE,
  showUnivarDensityOnDiag = FALSE,
  ...
)

```

**Arguments**

x	An object of class POUMM.
type	A character indicating the type of plot(s) to be generated. Defaults to "MCMC", resulting in a trace and density plot for the selected statistics (see argument stat).
doPlot	Logical indicating whether a plot should be printed on the currently active graphics device or whether to return a list of ggplot objects for further processing. Defaults to TRUE.
interactive	Logical indicating whether the user should press a key before generating a next plot (when needed to display two or more plots). Defaults to TRUE. Meaningless if doPlot = FALSE.
stat	A character vector with the names of statistics to be plotted. These should be names from the stats-list (see argument statFunctions). Defaults to c("alpha", "theta", "sigma", "sigmae", "H2tMean", "H2tInf").
chain	A vector of integers indicating the chains to be plotted.
startMCMC, endMCMC, thinMCMC	Integers used to extract a sample from the MCMC-chain; passed to summary().
statFunctions	Named list of statistics functions; passed to summary().
doZoomIn	(type MCMC only) A logical value indicating whether the produced plots should have a limitation on the x-axis according to an expression set in zoomInFilter (see below). Default value is FALSE.
zoomInFilter	A character string which evaluates as logical value. If doZoomIn is set to TRUE, this filter is applied to each point in each MCMC chain and the data-point is filtered out if it evaluates to FALSE. This allows to zoomIn the x-axis of density plots but should be used with caution, since filtering out points from the MCMC-sample can affect the kernel densities. Unfortunately, filtering out values is currently the only way to affect the limits of individual facets in ggplot2. The default value is a complicated expression involving the HPD from all MCMC chains (normally one chain from the prior and 2 chains from the posterior): <code>zoomInFilter = paste0("(stat \"(value &gt;= 0 &amp; value &lt;= 1) \")  \", \"(!stat \"(value &lt;= median(HPDUpper) + 4 * (median(HPDUpper) - median(HPDLower)) &amp;\", \"value &gt;= median(HPDLower) - 4 * (median(HPDUpper) - median(HPDLower)))\"))\"").</code> The identifiers in this expression can be any column names found in a summary of a POUMM object.
prettyNames	A logical indicating if greek letters and sub/superscripts should be used for the names of columns in the posterior density pairs-plot.
showUnivarDensityOnDiag	A logical indicating if univariate density plots should be displayed on the main diagonal in the bivariate posterior plot. Defaults to FALSE, in which case the column names are displayed on the diagonal.
...	not used, needed for consistency with the generic plot-function.

**Value**

If doPlot==FALSE, a named list containing a member called data of class data.table and several members of class ggplot.

---

plot.summary.POUMM      *Plot a summary of a POUMM fit*

---

## Description

Plot a summary of a POUMM fit

## Usage

```
## S3 method for class 'summary.POUMM'
plot(
  x,
  type = c("MCMC"),
  doPlot = TRUE,
  stat = c("alpha", "theta", "sigma", "sigmae", "g0", "H2tMean"),
  chain = NULL,
  doZoomIn = FALSE,
  zoomInFilter = paste0("(stat %in% c('H2e', 'H2tMean', 'H2tInf', 'H2tMax') & ",
    "(value >= 0 & value <= 1) ) |",
    "( !stat %in% c('H2e', 'H2tMean', 'H2tInf', 'H2tMax') & ",
    "(value <= median(HPDUpper) + 4 * (median(HPDUpper) - median(HPDLow)) & ",
    "value >= median(HPDLow) - 4 * (median(HPDUpper) - median(HPDLow)))")",
  palette = c("#999999", "#0072B2", "#CC79A7", "#E69F00", "#D55E00", "#56B4E9",
    "#009E73", "#F0E442"),
  prettyNames = TRUE,
  ...
)
```

## Arguments

x	An object of class POUMM.
type	A character indicating the type of plot(s) to be generated. Defaults to "MCMC", resulting in a trace and density plot for the selected statistics (see argument stat). Currently, only 'MCMC' type is supported.
doPlot	Logical indicating whether a plot should be printed on the currently active graphics device or whether only to return a list of plot- objects for further processing. Defaults to TRUE.
stat	A character vector with the names of statistics to be plotted. These should be names from the stats-list (see argument statFunctions). Defaults to c("alpha", "theta", "sigma", "sigmae", "H2tMean", "H2tInf").
chain	A vector of integers indicating the chains to be plotted.
doZoomIn	(type MCMC only) A logical value indicating whether the produced plots should have a limitation on the x-axis according to an expression set in zoomInFilter (see below). Default value is FALSE.



zoomInFilter	A character string which evaluates as logical value. If doZoomIn is set to TRUE, this filter is applied to each point in each MCMC chain and the data-point is filtered out if it evaluates to FALSE. This allows to zoomIn the x-axis of density plots but should be used with caution, since filtering out points from the MCMC-sample can affect the kernel densities. Unfortunately, filtering out values is currently the only way to affect the limits of individual facets in ggplot2. The default value is a complicated expression involving the HPD from all MCMC chains (normally one chain from the prior and 2 chains from the posterior): zoomInFilter = paste0("stat The identifiers in this expression can be any column names found in a summary of a POUMM object.
palette	A vector of colors (can be character strings) corresponding to the different chains (in their order 1 (prior), 2, 3). Defaults to c("#999999", "#0072B2", "#CC79A7", "#E69F00", "#D55E00", "#56B4E9", "#009E73", "#F0E442"), which is a color-blind friendly.
prettyNames	A logical indicating if greek letters and sub/superscripts should be used for the names of columns in the posterior density pairs-plot.
...	Not used; included for compatibility with the generic function plot.

### Value

If doPlot==TRUE, the function returns nothing and produces output on the current graphics device as a side-effect. Otherwise, the function returns a list of plot-objects: traceplot and densplot.

### Examples

```
## Not run:
library(POUMM)

set.seed(1)

N <- 1000

# create a random non-ultrametric tree of N tips
tree <- ape::rtree(N)

# Simulate the evolution of a trait along the tree
z <- rVNodesGivenTreePOUMM(
  tree, g0 = 8, alpha = 1, theta = 4, sigma = 1.2, sigmae = .8)

fit <- POUMM(z[1:N], tree, spec = list(nSamplesMCMC = 4e5))

# Summarize the results from the fit in a table:
summary(fit)

# Create plots for some of the inferred parameters/statistics:
p1 <- plot(fit, stat = c("alpha", "theta", "sigma", "sigmae", "H2tMean"),
  doZoomIn = TRUE,
  zoomInFilter = paste("!(stat %in% c('alpha', 'sigma', 'sigmae')) |",
    "(value >= 0 & value <= 8)"),
  doPlot = FALSE)
```

```

pl$traceplot
pl$densplot

## End(Not run)

```

---

POUMM

*The Phylogenetic (Ornstein-Uhlenbeck) Mixed Model*


---

### Description

This is the high-level entry point to the POUMM method. The POUMM function fits the POUMM method to a tree and observed trait-values at its tips and returns an object of class "POUMM".

### Usage

```

POUMM(
  z,
  tree,
  se = 0,
  zName = "z",
  treeName = "tree",
  parDigits = 6,
  usempfr = 0,
  useCpp = TRUE,
  ...,
  spec = NULL,
  doMCMC = TRUE,
  likPOUMM_lowLevelFun = likPOUMMGivenTreeVTipsC,
  verbose = FALSE,
  debug = FALSE
)

```

### Arguments

- |      |   |
|------|---|
| z    | Either a numeric vector containing the phenotypic values at the tips of tree or a named list containing named elements z - a numeric vector and tree - a phylo object (it is possible to specify different element names using the arguments zName and treeName).   |
| tree | A phylo object or NULL in case z is a list.   |
| se   | A non-negative numerical vector (or single number) indicating known measurement standard error (defaults to 0). Note the elements of this vector are assumed to describe the measurement error at individual nodes independent of the environmental contribution (described by the parameter sigmae). The total error standard deviation is thus $\sqrt{\text{sigmae}^2 + \text{se}^2}$ . |

<code>zName, treeName</code>	Character strings used when the parameter <code>z</code> is a list; indicate the names in the list of the values-vector and the tree. Default: <code>'z'</code> and <code>'tree'</code> .
<code>parDigits</code>	Integer specifying rounding to be done on the parameter vector before likelihood calculation. Defaults to 6 decimal digits. This can be useful during maximum likelihood optimization to prevent likelihood calculation on very small but positive values of alpha, but should be used with caution since specifying a small number of digits, i.e. 2 or 3 can result in an infinite loop during optim. Specify a negative number to disable rounding.
<code>usempfr</code>	integer indicating if and how mpfr should be used for small parameter values ( <code>'any(c(alpha, sigma, sigmae) &lt; 0.01)'</code> ). Using the mpfr package can be forced by specifying an integer greater or equal to 2. Setting <code>usempfr=0</code> (default) causes high precision likelihood calculation to be done on each encounter of parameters with at least 1 bigger log-likelihood value than any of the currently found maximum log-likelihood or the previously calculated log-likelihood value Requires the Rmpfr package. Note that using mpfr may increase the time for one likelihood calculation more than 100-fold. Set <code>usempfr</code> to -1 or less to completely disable Rmpfr functionality.
<code>useCpp</code>	Logical indicating whether C++ likelihood calculation should be used for faster vector operations. Defaults to TRUE. Since the C++ likelihood implementation does not support mpfr, <code>useCpp</code> gets disabled when <code>usempfr</code> is bigger than 0.
<code>...</code>	additional arguments passed to the <code>'likPOUMMGivenTreeVTips()'</code> function ( <code>'?dV-GivenTreeOU'</code> for details).
<code>spec</code>	A named list specifying how the ML and MCMC fit should be done. See <code>'?specifyPOUMM'</code> .
<code>doMCMC</code>	Deprecated - replaced by specifying <code>nSamplesMCMC</code> as a member of <code>spec</code> instead (see <code>'?specifyPOUMM'</code> ). logical: should a MCMC fit be performed. An MCMC fit provides a sample from the posterior distribution of the parameters given a prior distribution and the data. Unlike the ML-fit, it allows to estimate confidence intervals for the estimated parameters. This argument is TRUE by default. The current implementation uses the adaptive Metropolis sampler from the package <code>'adaptMCMC'</code> written by Andreas Scheidegger. To obtain meaningful estimates MCMC may need to run for several millions of iterations (parameter <code>nSamplesMCMC</code> set to <code>1e5</code> by default). See parameters ending at MCMC in <code>'?specifyPOUMM'</code> for details.
<code>likPOUMM_lowLevelFun</code>	the low-level function used for POUMM - likelihood calculation. Default value is <code>likPOUMMGivenTreeVTipsC</code> .
<code>verbose, debug</code>	Logical flags indicating whether to print informative and/or debug information on the standard output (both are set to to FALSE by default).

### Value

An object of S3 class `'POUMM'`. This object can be analyzed using S3 generic functions: [summary](#), [plot](#), [AIC](#), [BIC](#), [coef](#), [logLik](#), [fitted](#).

## References

- Mitov, V., and Stadler, T. (2017). Fast and Robust Inference of Phylogenetic Ornstein-Uhlenbeck Models Using Parallel Likelihood Calculation. *bioRxiv*, 115089. <https://doi.org/10.1101/115089>
- Mitov, V., & Stadler, T. (2017). Fast Bayesian Inference of Phylogenetic Models Using Parallel Likelihood Calculation and Adaptive Metropolis Sampling. *Systematic Biology*, 235739. <http://doi.org/10.1101/235739>
- Vihola, M. (2012). Robust adaptive Metropolis algorithm with coerced acceptance rate. *Statistics and Computing*, 22(5), 997-1008. <http://doi.org/10.1007/s11222-011-9269-5>
- Scheidegger, A. (2012). adaptMCMC: Implementation of a generic adaptive Monte Carlo Markov Chain sampler. <http://CRAN.R-project.org/package=adaptMCMC>

## See Also

[specifyPOUMM](#) for parametrizations and custom settings of the POUMM fit.

## Examples

```
## Not run:
# Please, read the package vignette for more detailed examples.
N <- 500
tr <- ape::rtree(N)
z <- rVNodesGivenTreePOUMM(tr, 0, 2, 3, 1, 1)[1:N]
fit <- POUMM(z, tr, spec = specifyPOUMM(nSamplesMCMC = 5e4))
plot(fit)
summary(fit)
AIC(fit)
BIC(fit)
coef(fit)
logLik(fit)
fitted(fit)
plot(resid(fit))
abline(h=0)

# fit PMM to the same data and do a likelihood ratio test
fitPMM <- POUMM(z, tr, spec = specifyPMM(nSamplesMCMC = 5e4))
lmtest::lrtest(fitPMM, fit)

## End(Not run)
```

## Description

We define a dev release as having a sub-release, eg 0.9.15.5 is one whereas 0.9.16 is not.

**Usage**

```
POUMMI$ADevRelease()
```

**Value**

a logical

---

pruneTree	<i>Extract information for fast likelihood calculation using the breadth-first pruning algorithm.</i>
-----------	---

---

**Description**

Extract information for fast likelihood calculation using the breadth-first pruning algorithm.

**Usage**

```
pruneTree(tree, z, se = 0)
```

**Arguments**

tree	a phylo object
z	Numeric vector with length(tree\$tip.label) values corresponding to tree\$tip.label.
se	Non-negative numerical or N-vector indicating known standard measurement error.

**Value**

a list of objects used for likelihood evaluation

---

residuals.POUMM	<i>Extract maximum likelihood environmental contributions (residuals) at the tips of a tree, to which a POUMM model has been fitted.</i>
-----------------	--

---

**Description**

Extract maximum likelihood environmental contributions (residuals) at the tips of a tree, to which a POUMM model has been fitted.

**Usage**

```
## S3 method for class 'POUMM'
residuals(object, ...)
```

**Arguments**

object            An object of class POUMM.  
 ...                Not used; added for compatibility with generic function residuals.

**Value**

The vector of e-values (residuals) corresponding to the tip-labels in the tree.

---

rTrajectoryOU	<i>Generation of a random trajectory of an OU process starting from a given initial state</i>
---------------	---

---

**Description**

Generates a trajectory  $x_t$  given initial state  $z_0$  according to an Ornstein-Uhlenbeck process.

**Usage**

```
rTrajectoryOU(z0, t, alpha, theta, sigma, steps = 1)
```

**Arguments**

$z_0$                 Numeric value, initial state.  
 t                    Numeric value or vector of size steps, denoting the time-step(s).  
 alpha, theta, sigma        Numeric values, parameters of the OU process.  
 steps                Integer, number of steps.

**Value**

A numeric vector of length steps containing the generated values at times  $0+t$ ,  $0+2t$ , ...,  $0+steps*t$ .

**Examples**

```
z0 <- 0
nSteps <- 100
t <- 0.01
trajectory <- rTrajectoryOU(z0, t, 2, 2, 1, steps = nSteps)
plot(trajectory, type = 'l')
```

---

rTrajectoryOUDef	<i>Generation of a random trajectory of an OU process starting from a given initial state (only for test purpose)</i>
------------------	---

---

**Description**

Generation of a random trajectory of an OU process starting from a given initial state (only for test purpose)

**Usage**

```
rTrajectoryOUDef(z0, t, alpha, theta, sigma, steps = 1)
```

**Arguments**

z0	Numeric value, initial state.
t	Numeric value or vector of size steps, denoting the time-step(s).
alpha	Numeric values, parameters of the OU process.
theta	Numeric values, parameters of the OU process.
sigma	Numeric values, parameters of the OU process.
steps	Integer, number of steps.

**Details**

Used for test purpose only. This is an internal function and is appropriate for small time-steps only.

---

rVNodesGivenTreePOUMM	<i>Random generation of values along a phylogenetic tree following a branching OU process</i>
-----------------------	---

---

**Description**

Random generation of values along a phylogenetic tree following a branching OU process

**Usage**

```
rVNodesGivenTreePOUMM(tree, z0, alpha, theta, sigma, sigmae = 0)
```

**Arguments**

tree	An object of class phylo (package ape).
z0	Numeric value indicating the initial state at the root of the tree.
alpha, theta, sigma	Numeric values, parameters of the OU process.
sigmae	Numeric non-negative value (default 0). Specifies the standard deviation of random environmental contribution and or measurement standard error to be added to the values (white noise). Note that if measurement standard error, se, is known and needs to be added to the environmental contribution, the right way to specify the parameter would be $\text{sqrt}(\text{sigmae}^2 + \text{se}^2)$ .

**Value**

A numeric vector containing the generated values at all nodes (internal and tips) of the phylogenetic tree.

---

simulatePOUMMLikelihoodMainLoop

*Writes verbose messages of the order of tree traversal during likelihood calculation*

---

**Description**

Writes verbose messages of the order of tree traversal during likelihood calculation

**Usage**

```
simulatePOUMMLikelihoodMainLoop(tree)
```

**Arguments**

tree	A phylo object.
------	-----------------

**Value**

Nothing



---

simulateTrait	<i>Simulate a trait on a tree under a ML fit of the POUMM model</i>
---------------	---

---

**Description**

Use the maximum likelihood parameters of the model to simulate trait values on a phylogenetic tree.

**Usage**

```
simulateTrait(object, tree = NULL)
```

**Arguments**

object	an S3 object of class POUMM
tree	a phylo object. If NULL (default) the trait is simulated on the tree, on which the POUMM object has been fit.

**Details**

This function is a shortcut to calling [rVNodesGivenTreePOUMM](#), which will map the inferred parameters of the model back to the original POUMM parameters alpha, theta, sigma, sigmae, and g0.

**Value**

a numerical vector containing the simulated trait value for each tip in the tree.

**See Also**

[rVNodesGivenTreePOUMM](#)

---

specPOUMM	<i>Specifying a POUMM fit</i>
-----------	-------------------------------

---

**Description**

Specification and validation of POUMM/PMM settings.

**Usage**

```
specifyPOUMM(  
  z = NULL,  
  tree = NULL,  
  zMin = -10,  
  zMean = 0,  
  zMax = 10,  
  zVar = 4,  
  zSD = sqrt(zVar),  
  tMin = 0.1,  
  tMean = 2,  
  tMax = 10,  
  parMapping = NULL,  
  parLower = NULL,  
  parUpper = NULL,  
  g0Prior = NULL,  
  parInitML = NULL,  
  control = NULL,  
  parPriorMCMC = NULL,  
  parInitMCMC = NULL,  
  parScaleMCMC = NULL,  
  nSamplesMCMC = 1e+05,  
  nAdaptMCMC = nSamplesMCMC,  
  thinMCMC = 100,  
  accRateMCMC = 0.01,  
  gammaMCMC = 0.50001,  
  nChainsMCMC = 3,  
  samplePriorMCMC = TRUE,  
  parallelMCMC = FALSE,  
  validateSpec = TRUE  
)
```

```
specifyPOUMM_ATS(  
  z = NULL,  
  tree = NULL,  
  zMin = -10,  
  zMean = 0,  
  zMax = 10,  
  zVar = 4,  
  zSD = sqrt(zVar),  
  tMin = 0.1,  
  tMean = 2,  
  tMax = 10,  
  parMapping = NULL,  
  parLower = NULL,  
  parUpper = NULL,  
  g0Prior = NULL,  
  parInitML = NULL,
```

```
control = NULL,  
parPriorMCMC = NULL,  
parInitMCMC = NULL,  
parScaleMCMC = NULL,  
nSamplesMCMC = 1e+05,  
nAdaptMCMC = nSamplesMCMC,  
thinMCMC = 100,  
accRateMCMC = 0.01,  
gammaMCMC = 0.50001,  
nChainsMCMC = 3,  
samplePriorMCMC = TRUE,  
parallelMCMC = FALSE,  
sigmaeFixed = 0  
)
```

```
specifyPOUMM_ATSG0(  
z = NULL,  
tree = NULL,  
zMin = -10,  
zMean = 0,  
zMax = 10,  
zVar = 4,  
zSD = sqrt(zVar),  
tMin = 0.1,  
tMean = 2,  
tMax = 10,  
parMapping = NULL,  
parLower = NULL,  
parUpper = NULL,  
g0Prior = NULL,  
parInitML = NULL,  
control = NULL,  
parPriorMCMC = NULL,  
parInitMCMC = NULL,  
parScaleMCMC = NULL,  
nSamplesMCMC = 1e+05,  
nAdaptMCMC = nSamplesMCMC,  
thinMCMC = 100,  
accRateMCMC = 0.01,  
gammaMCMC = 0.50001,  
nChainsMCMC = 3,  
samplePriorMCMC = TRUE,  
parallelMCMC = FALSE,  
sigmaeFixed = 0  
)
```

```
specifyPOUMM_ATSSeG0(  
z = NULL,
```

```
tree = NULL,
zMin = -10,
zMean = 0,
zMax = 10,
zVar = 4,
zSD = sqrt(zVar),
tMin = 0.1,
tMean = 2,
tMax = 10,
parMapping = NULL,
parLower = NULL,
parUpper = NULL,
g0Prior = NULL,
parInitML = NULL,
control = NULL,
parPriorMCMC = NULL,
parInitMCMC = NULL,
parScaleMCMC = NULL,
nSamplesMCMC = 1e+05,
nAdaptMCMC = nSamplesMCMC,
thinMCMC = 100,
accRateMCMC = 0.01,
gammaMCMC = 0.50001,
nChainsMCMC = 3,
samplePriorMCMC = TRUE,
parallelMCMC = FALSE
)

specifyPMM(
z = NULL,
tree = NULL,
zMin = -10,
zMean = 0,
zMax = 10,
zVar = 4,
zSD = sqrt(zVar),
tMin = 0.1,
tMean = 2,
tMax = 10,
parMapping = NULL,
parLower = NULL,
parUpper = NULL,
g0Prior = NULL,
parInitML = NULL,
control = NULL,
parPriorMCMC = NULL,
parInitMCMC = NULL,
parScaleMCMC = NULL,
```

```
nSamplesMCMC = 1e+05,  
nAdaptMCMC = nSamplesMCMC,  
thinMCMC = 100,  
accRateMCMC = 0.01,  
gammaMCMC = 0.50001,  
nChainsMCMC = 3,  
samplePriorMCMC = TRUE,  
parallelMCMC = FALSE  
)
```

```
specifyPMM_SSeG0(  
  z = NULL,  
  tree = NULL,  
  zMin = -10,  
  zMean = 0,  
  zMax = 10,  
  zVar = 4,  
  zSD = sqrt(zVar),  
  tMin = 0.1,  
  tMean = 2,  
  tMax = 10,  
  parMapping = NULL,  
  parLower = NULL,  
  parUpper = NULL,  
  g0Prior = NULL,  
  parInitML = NULL,  
  control = NULL,  
  parPriorMCMC = NULL,  
  parInitMCMC = NULL,  
  parScaleMCMC = NULL,  
  nSamplesMCMC = 1e+05,  
  nAdaptMCMC = nSamplesMCMC,  
  thinMCMC = 100,  
  accRateMCMC = 0.01,  
  gammaMCMC = 0.50001,  
  nChainsMCMC = 3,  
  samplePriorMCMC = TRUE,  
  parallelMCMC = FALSE  
)
```

```
specifyPOUMM_ATH2tMeanSe(  
  z = NULL,  
  tree = NULL,  
  zMin = -10,  
  zMean = 0,  
  zMax = 10,  
  zVar = 4,  
  zSD = sqrt(zVar),
```

```
tMin = 0.1,
tMean = 2,
tMax = 10,
parMapping = NULL,
parLower = NULL,
parUpper = NULL,
g0Prior = NULL,
parInitML = NULL,
control = NULL,
parPriorMCMC = NULL,
parInitMCMC = NULL,
parScaleMCMC = NULL,
nSamplesMCMC = 1e+05,
nAdaptMCMC = nSamplesMCMC,
thinMCMC = 100,
accRateMCMC = 0.01,
gammaMCMC = 0.50001,
nChainsMCMC = 3,
samplePriorMCMC = TRUE,
parallelMCMC = FALSE
)

specifyPOUMM_ATH2tMeanSeG0(
  z = NULL,
  tree = NULL,
  zMin = -10,
  zMean = 0,
  zMax = 10,
  zVar = 4,
  zSD = sqrt(zVar),
  tMin = 0.1,
  tMean = 2,
  tMax = 10,
  parMapping = NULL,
  parLower = NULL,
  parUpper = NULL,
  g0Prior = NULL,
  parInitML = NULL,
  control = NULL,
  parPriorMCMC = NULL,
  parInitMCMC = NULL,
  parScaleMCMC = NULL,
  nSamplesMCMC = 1e+05,
  nAdaptMCMC = nSamplesMCMC,
  thinMCMC = 100,
  accRateMCMC = 0.01,
  gammaMCMC = 0.50001,
  nChainsMCMC = 3,
```

```
    samplePriorMCMC = TRUE,  
    parallelMCMC = FALSE  
  )  
  
  specifyPMM_H2tMeanSe(  
    z = NULL,  
    tree = NULL,  
    zMin = -10,  
    zMean = 0,  
    zMax = 10,  
    zVar = 4,  
    zSD = sqrt(zVar),  
    tMin = 0.1,  
    tMean = 2,  
    tMax = 10,  
    parMapping = NULL,  
    parLower = NULL,  
    parUpper = NULL,  
    g0Prior = NULL,  
    parInitML = NULL,  
    control = NULL,  
    parPriorMCMC = NULL,  
    parInitMCMC = NULL,  
    parScaleMCMC = NULL,  
    nSamplesMCMC = 1e+05,  
    nAdaptMCMC = nSamplesMCMC,  
    thinMCMC = 100,  
    accRateMCMC = 0.01,  
    gammaMCMC = 0.50001,  
    nChainsMCMC = 3,  
    samplePriorMCMC = TRUE,  
    parallelMCMC = FALSE  
  )  
  
  specifyPMM_H2tMeanSeG0(  
    z = NULL,  
    tree = NULL,  
    zMin = -10,  
    zMean = 0,  
    zMax = 10,  
    zVar = 4,  
    zSD = sqrt(zVar),  
    tMin = 0.1,  
    tMean = 2,  
    tMax = 10,  
    parMapping = NULL,  
    parLower = NULL,  
    parUpper = NULL,
```

```

g0Prior = NULL,
parInitML = NULL,
control = NULL,
parPriorMCMC = NULL,
parInitMCMC = NULL,
parScaleMCMC = NULL,
nSamplesMCMC = 1e+05,
nAdaptMCMC = nSamplesMCMC,
thinMCMC = 100,
accRateMCMC = 0.01,
gammaMCMC = 0.50001,
nChainsMCMC = 3,
samplePriorMCMC = TRUE,
parallelMCMC = FALSE
)

```

### Arguments

- z, tree** a numeric vector and a phylo object on which the fit is to be done. These arguments are used in order to guess meaningful values for the `parLower`, `parUpper` and `parPriorMCMC` arguments. See also, `zMin`, `zMean`, ..., `tMax` below.
- zMin, zMean, zMax, zVar, zSD, tMin, tMean, tMax** summary statistics of the observed tip-values (`z`) and root-tip distances (`t`). Some of these values are used for constructing default parameter values and limits; These arguments are given default values which will most likely be meaningless in your specific use-case. The default values will be overwritten with the corresponding statistics from the `z` and `tree` arguments if these were specified. If none of `tree` and `z`, nor these parameters are specified, then the arguments `parLower`, `parUpper`, `parPriorMCMC` must be specified explicitly.
- parMapping** An R-function that can handle, both, a numeric vector or a numeric matrix as argument. This function should transform the input vector or each row-vector (if the input is matrix) into a (row-)vector of the POUMM parameters `alpha`, `theta`, `sigma`, `sigmae`, `g0`. For a vector input the function should return a vector with named elements `alpha`, `theta`, `sigma`, `sigmae`, `g0`. For a matrix input the function should return a matrix with the same number of rows and columns `alpha`, `theta`, `sigma`, `sigmae`, `g0`. Only finite non-negative values are allowed for `alpha`, `sigma`, and `sigmae`. Returning `Inf`, `-Inf`, `NA` or `NaN` for any of these parameters will result in an error during likelihood calculation. Only finite numerical values are allowed for `theta`. The parameter `g0` is treated in a special way and can assume either a finite numerical value or one of `NA` or `NaN`. If `g0 = finite value`, this value is used together with the corresponding values of `alpha`, `theta`, `sigma`, and `sigmae` for likelihood calculation. If `g0 = NA` (meaning value Not Available), the value of `g0` is calculated analytically during likelihood calculation in order to maximise one of the following:

1. if a normal prior for `g0` was specified (see `g0Prior`),  $pdf(z|\alpha, \theta, \sigma, \sigma_e, g0, tree)xprior(g0)$ .
2. otherwise,  $pdf(z|\alpha, \theta, \sigma, \sigma_e, g0, tree)$ .



If  $g0 = \text{NaN}$  (meaning Not a Number), then the likelihood is marginalized w.r.t. the  $g0$ 's prior distribution (see `g0Prior`), i.e. the likelihood returned is:  $pdf(z|\alpha, \theta, \sigma, \sigma_e, tree) = \int pdf(z|\alpha, \theta, \sigma, \sigma_e, g0) x pdf(g0) dg0; g0 \text{ from } -\infty \text{ to } +\infty$  In this case ( $g0 = \text{NaN}$ ), if `g0Prior` is not specified, it is assumed that `g0Prior` is the stationary OU normal distribution with mean, theta, and variance, `varOU(Inf, alpha, sigma)`.

Examples:

```
# Default for POUMM: identity for alpha, theta, sigma, sigmae, NA for g0.
parMapping = function(par) {
  if(is.matrix(par)) {
    atsseg0 <- cbind(par[, 1:4, drop = FALSE], NA)
    colnames(atsseg0) <- c("alpha", "theta", "sigma", "sigmae", "g0")
  } else {
    atsseg0 <- c(par[1:4], NA)
    names(atsseg0) <- c("alpha", "theta", "sigma", "sigmae", "g0")
  }
  atsseg0
}
```

`parLower, parUpper`

two named numeric vectors of the same length indicating the boundaries of the search region for the ML-fit. Calling `parMapping` on `parLower` and `parUpper` should result in appropriate values of the POUMM parameters alpha, theta, sigma, sigmae and `g0`. By default, the upper limit for alpha is set to  $69.31 / tMean$ , which corresponds to a value of alpha so big that the time for half-way convergence towards theta from any initial trait value is 100 times shorter than the mean root-tip distance in the tree. Examples:

```
# Default for POUMM:
parLower = c(alpha = 0, theta = zMin - 2 * (zMax - zMin), sigma = 0, sigmae = 0)
parUpper = c(alpha = 69.31 / tMean, theta = zMax + 2 * (zMax - zMin),
             sigma = sigmaOU(H2 = .99, alpha = 69.31 / tMean, sigmae = 2 * zSD,
                           t = tMean),
             sigmae = 2 * zSD)
```

`g0Prior`

Either NULL or a list with named numeric or character members "mean" and "var". Specifies a prior normal distribution for the parameter `g0`. If characters, the members "mean" and "var" are evaluated as R-expressions - useful if these are functions of some of other parameters. Note that if `g0Prior` is not NULL and `g0` is not NaN (either a fixed number or NA), then the likelihood maximization takes into account the prior for `g0`, that is, the optimization is done over the product  $p(g0) \times \text{lik}(\text{data} | g0, \text{other parameters and tree})$ . This can be helpful to prevent extremely big or low estimates of `g0`. To avoid this behavior and always maximize the likelihood, use `g0Prior = NULL`.

`parInitML`

A named vector (like `parLower` and `parUpper`) or a list of such vectors - starting points for optim.

`control`

List of parameters passed on to optim in the ML-fit, default `list(factr=1e9)`, see `?optim`.

parPriorMCMC	<p>A function of a numeric parameter-vector returning the log-prior for this parameter vector. Example:</p> <pre># Default for POUMM: parPriorMCMC = function(par) {   dexp(par[1], rate = tMean / 6.931, TRUE) +   dnorm(par[2], zMean, 10 * zSD, TRUE) +   dexp(par[3], rate = sqrt(tMean / (zVar * 0.6931)), TRUE) +   dexp(par[4], rate = 2 / zSD, TRUE) }</pre>
parInitMCMC	<p>a function(chainNo, fitML) returning an initial state of an MCMC as a vector. The argument fitML can be used to specify an initial state, close to a previously found likelihood optimum. Example:</p> <pre># Default for POUMM: parInitMCMC = function(chainNo, fitML) {   if(!is.null(fitML)) {     parML &lt;- fitML\$par   } else {     parML &lt;- NULL   }    init &lt;- rbind(     c(alpha = 0, theta = 0, sigma = 1, sigmae = 0),     parML,     c(alpha = 0, theta = 0, sigma = 1, sigmae = 1)   )    init[(chainNo - 1) %% nrow(init) + 1, ] }</pre>
parScaleMCMC	Numeric matrix indicating the initial jump-distribution matrix for the MCMC fit. Default for POUMM is diag(4);
nSamplesMCMC	Integer indicating the length of each MCMC chain. Defaults to 1e5.
nAdaptMCMC	Logical indicating whether adaptation of the MCMC jump distribution should be done with respect to the target acceptance rate (accRateMCMC) or integer indicating how many initial MCMC iterations should be used for adaptation of the jump-distribution matrix (see details in ?POUMM). Defaults to nSamplesMCMC meaning continuous adaptation throughout the MCMC.
thinMCMC	Integer indicating the thinning interval of the mcmc-chains. Defaults to 100.
accRateMCMC	numeric between 0 and 1 indicating the target acceptance rate of the adaptive Metropolis sampling (see details in ?POUMM). Default 0.01.
gammaMCMC	controls the speed of adaption. Should be in the interval (0.5,1]. A lower gamma leads to faster adaption. Default value is 0.50001.
nChainsMCMC	integer indicating the number of chains to run. Defaults to 3 chains, from which the first one is a sample from the prior distribution (see samplePriorMCMC).

samplePriorMCMC	Logical indicating if sampling from the prior should be done for the first chain (see nChainsMCMC). This is useful to compare mcmc's for an overlap between prior and posterior distributions. Default is TRUE.
parallelMCMC	Logical indicating whether the MCMC chains should be run in parallel. Setting this option to TRUE results in using <code>foreach::foreach() %dopar% { }</code> construct for the MCMC fit. In order for parallel execution to be done, you should create a computing cluster and register it as parallel back-end (see example in package vignette and the web-page <a href="https://github.com/tobigithub/R-parallel/wiki/R-parallel-Setups">https://github.com/tobigithub/R-parallel/wiki/R-parallel-Setups</a> ).
validateSpec	Logical indicating whether the passed parameters should be validated. This parameter is used internally and should always be TRUE.
sigmaeFixed	fixed value for the sigmae parameter (used in specifyPOUMM_ATS and specifyPOUMM_ATSG0).

### Value

A named list to be passed as a spec argument to POUMM.

### Functions

- `specifyPOUMM`: Specify parameters for fitting a POUMM model. Parameter vector is `c(alpha, theta, sigma, sigmae)`. Default model settings.
- `specifyPOUMM_ATS`: Fitting a POU model with fixed sigmae. Parameter vector is `c(alpha, theta, sigma)`.
- `specifyPOUMM_ATSG0`: Fitting a POU model with fixed sigmae. Parameter vector is `c(alpha, theta, sigma, g0)`.
- `specifyPOUMM_ATSseG0`: Fitting a POUMM model with sampling of g0. Parameter vector is `c(alpha, theta, sigma, sigmae, g0)`.
- `specifyPMM`: Specify parameter for fitting a PMM model. Parameter vector is `c(sigma, sigmae)`.
- `specifyPMM_SseG0`: Specify parameter for fitting a PMM model with sampling of g0. Parameter vector is `c(sigma, sigmae, g0)`.
- `specifyPOUMM_ATH2tMeanSe`: Fitting a POUMM model with a uniform prior for the phylogenetic heritability at mean root-tip distance. Parameter vector is `c(alpha, theta, H2tMean, sigmae)`.
- `specifyPOUMM_ATH2tMeanSeG0`: Fitting a POUMM model with a uniform prior for the phylogenetic heritability at mean root-tip with sampling of g0. Parameter vector is `c(alpha, theta, H2tMean, sigmae, g0)`.
- `specifyPMM_H2tMeanSe`: Fitting a PMM model with a uniform prior for the phylogenetic heritability at mean root-tip distance. Parameter vector is `c(H2tMean, sigmae)`.
- `specifyPMM_H2tMeanSeG0`: Fitting a PMM model with a uniform prior for the phylogenetic heritability at mean root-tip distance with sampling of G0. Parameter vector is `c(H2tMean, sigmae, g0)`.

---

statistics	<i>Extract statistics from sampled or inferred parameters of a POUMM fit</i>
------------	--

---

**Description**

Extract statistics from sampled or inferred parameters of a POUMM fit

**Usage**

```
statistics(object)

## S3 method for class 'POUMM'
statistics(object)
```

**Arguments**

object            An object of class "POUMM".

**Details**

This is a generic method.

**Methods (by class)**

- POUMM: Relevant statistics from the sampled parameters of a POUMM fit

---

summary.POUMM	<i>Summarize the results of a POUMM-fit</i>
---------------	---

---

**Description**

Summarize the results of a POUMM-fit

**Usage**

```
## S3 method for class 'POUMM'
summary(
  object,
  ...,
  startMCMC = NA,
  endMCMC = NA,
  thinMCMC = 1000,
  stats = statistics(object),
  mode = c("short", "long", "expert")
)
```

**Arguments**

object	a POUMM object returned by POUMM-function (see ?POUMM).
...	Not used, but declared for consistency with the generic method summary.
startMCMC, endMCMC	integers indicating the range of the MCMC chains to be used for the analysis (excluding the initial warm-up phase)
thinMCMC	thinning interval of the MCMC chain to avoid strong autocorrelation between sampled elements;
stats	a named list of functions of the form function(par) number , which are called for each sample of each mcmc chain in object. Defaults to a call of statistics(object) returning a list of statistics functions relevant for the object. See also statistics.
mode	a character indicating the desired format of the returned summary as follows: 'short' - a data.table with the ML and MCMC estimates of heritability, model parameters, root-value and other statistics. 'long' - same information as in 'short' but including also the samples, which can be convenient for

---

validateSpecPOUMM      *Validate a POUMM specification*

---

**Description**

Validate a POUMM specification

**Usage**

```
validateSpecPOUMM(spec)
```

**Arguments**

spec	A list object returned by one of the specifyPOUMM or specifyPMM functions with possibly modified entries afterwards.
------	--

**Value**

The function either returns TRUE or exits with an error message if it finds a problem with the specifcaiton.

---

validateZTree	<i>Validate phenotypic values and phylogenetic tree</i>
---------------	---

---

**Description**

Validate phenotypic values and phylogenetic tree

**Usage**

```
validateZTree(z, tree)
```

**Arguments**

<code>z</code>	trait (phenotypic) values at the tips of the tree
<code>tree</code>	A phylo object with the same number of tips as the length of <code>z</code> .

**Value**

The function either returns TRUE or exits with an error message if it finds a problem with the specification.

---

vignetteCachedResults	<i>Cached objects for the POUMM vignettes and examples</i>
-----------------------	--

---

**Description**

A list containing a simulated tree, trait-values and POUMM objects (model fits). To use these objects in examples you can load them into the global workspace with the command: `'data(vignetteCachedResults); list2env(vignetteCachedResults, globalenv());'`.

**Usage**

```
vignetteCachedResults
```

**Format**

This is a list containing the following named elements:

**g, z, e** numeric vectors of simulated genotypic values, phenotypic values and measurement errors.

**tree** a simulated phylogenetic tree.

**fitPOUMM, fitPOUMM2, fitH2tMean** POUMM fit objects to tree and `z`.

# Index

- \* **datasets**
  - vignetteCachedResults, 46
- AIC, 27
- alpha (PhylogeneticH2), 20
- analyseMCMCs, 3
- BIC, 27
- chld, 4
- coef, 27
- coef.POUMM, 4
- covFunPOUMM, 5
- covHPDFunPOUMM, 5
- covPOUMM, 6
- covVTipsGivenTreePOUMM, 7
- dOU (OU), 19
- dVNodesGivenTreePOUMM, 8
- dVTipsGivenTreePOUMM
  - (likPOUMMGivenTreeVTips), 11
- edgesFrom, 9
- fitted, 27
- fitted.POUMM, 9
- gPOUMM, 10
- H2, 10
- H2e (PhylogeneticH2), 20
- likPOUMMGivenTreeVTips, 11
- likPOUMMGivenTreeVTipsC, 12
- logLik, 27
- logLik.POUMM, 14
- loglik\_abc\_g0\_g0Prior, 15
- maxLikPOUMMGivenTreeVTips, 15
- mcmc.poumm
  - (mcmcPOUMMGivenPriorTreeVTips), 16
- mcmcPOUMMGivenPriorTreeVTips, 16
- meanOU (OU), 19
- nobs.POUMM, 18
- nodeTimes, 18
- OU, 19
- PhylogeneticH2, 20
- plot, 27
- plot.POUMM, 22
- plot.summary.POUMM, 24
- POUMM, 26
- POUMMIsADevRelease, 28
- pruneTree, 29
- residuals.POUMM, 29
- rOU (OU), 19
- rTrajectoryOU, 30
- rTrajectoryOUDef, 31
- rVNodesGivenTreePOUMM, 31, 33
- sdOU (OU), 19
- sigmae (PhylogeneticH2), 20
- sigmaOU (PhylogeneticH2), 20
- simulatePOUMMLikelihoodMainLoop, 32
- simulateTrait, 33
- specifyPMM (specPOUMM), 33
- specifyPMM\_H2tMeanSe (specPOUMM), 33
- specifyPMM\_H2tMeanSeG0 (specPOUMM), 33
- specifyPMM\_SSeG0 (specPOUMM), 33
- specifyPOUMM, 28
- specifyPOUMM (specPOUMM), 33
- specifyPOUMM\_ATH2tMeanSe (specPOUMM), 33
- specifyPOUMM\_ATH2tMeanSeG0 (specPOUMM), 33
- specifyPOUMM\_ATS (specPOUMM), 33
- specifyPOUMM\_ATSG0 (specPOUMM), 33
- specifyPOUMM\_ATSSeG0 (specPOUMM), 33
- specPOUMM, 33
- statistics, 44

summary, [27](#)

summary.POUMM, [44](#)

validateSpecPOUMM, [45](#)

validateZTree, [46](#)

varOU (OU), [19](#)

vignetteCachedResults, [46](#)