

Package ‘paleoTS’

April 5, 2012

Type Package

Title Analyze paleontological time-series

Version 0.4-3

Date 2012-04-03

Author Gene Hunt

Maintainer Gene Hunt <hunte@si.edu>

Depends mvtnorm

Description his package facilitates analysis of paleontological sequences of trait values from an evolving lineage. Functions are provided to fit, using maximum likelihood, evolutionary models including unbiased random walks, directional evolution, stasis, Ornstein-Uhlenbeck, punctuated change, and evolutionary models in which traits track some measured covariate.

License GPL (>= 2)

Repository CRAN

Date/Publication 2012-04-05 05:42:45

R topics documented:

paleoTS-package	2
as.paleoTS	3
as.paleoTSfit	5
cat.paleoTS	6
compareModels	7
fit.sgs	8
fit3models	10
fitGpunc	12
IC	14
ln.paleoTS	15

logL.GRW	16
logL.joint.GRW	17
LRI	18
lynchD	19
mle.GRW	21
modelCurves	22
opt.covTrack	23
opt.GRW	24
opt.GRW.shift	26
opt.joint.GRW	27
opt.RW.Mult	29
plot.paleoTS	31
sim.covTrack	32
sim.GRW	33
sim.OU	34
sim.punc	36
std.paleoTS	37
Stickleback data	38
sub.paleoTS	39
test.var.het	40

Index **42**

paleoTS-package

Analysis of evoluitional time-series

Description

This package facilitates analysis of paleontological sequences of trait values from an evolving lineage. Functions are provided to fit, using maximum likelihood, evolutionary models including unbiased random walks, directional evolution, stasis, Ornstein-Uhlenbeck, punctuated change, and evolutionary models in which traits track some measured covariate.

Details

Package: paleoTS
 Type: Package
 Version: 0.4-3
 Date: 2011-12-22
 License: GPL 2

Author(s)

Gene Hunt

References

- Hunt, G. 2006. Fitting and comparing models of phyletic evolution: random walks and beyond. *Paleobiology* **32**:578–601.
- Hunt, G., M. Bell & M. Travis. 2008. Evolution towards a new adaptive optimum: phenotypic evolution in a fossil stickleback lineage. *Evolution* **62**:700-710.
- Hunt, G. 2008. Gradual or pulsed evolution: when should punctuational explanations be preferred? *Paleobiology* **34**:360–377.
- Hunt, G. 2008. Evolutionary patterns within fossil lineages: model-based assessment of modes, rates, punctuations and process.. In R.K. Bambach and P.H. Kelley, eds. From Evolution to Geobiology: Research Questions Driving Paleontology at the Start of a New Century:578–601.
- Hunt, G., S. Wicaksono, J. E. Brown, and G. K. Macleod. 2010. Climate-driven body size trends in the ostracod fauna of the deep Indian Ocean. *Palaeontology* **53**(6):1255-1268.

Examples

```
x <- sim.GRW(ns=30, ms=0, vs=0.3) # simulate unbiased random walk
fit3models(x) # compare fits of directional, random walk, and stasis models
```

as.paleoTS	<i>Paleontological time-series class</i>
------------	--

Description

A class for storing information on evolutionary time-series data.

Usage

```
as.paleoTS(mm, vv, nn, tt, MM = NULL, genpars = NULL, label = NULL, start.age = NULL, oldest=c("first", "
read.paleoTS(file = NULL, oldest = "first", reset.time=TRUE, ...)
```

Arguments

mm	vector of sample means
vv	vector of sample variances
nn	vector of sample sizes
tt	vector of sample ages
MM	vector of true population means (for simulated time series)
genpars	generating parameters (for simulated time series)
label	optional label describing time series data
start.age	optional, indicating age of oldest sample (e.g., in millions of years ago)
file	file with paleoTS data; if NULL, then opens file.choose dialog
oldest	by default, samples are listed from oldest to youngest. If file is in reverse order, use oldest = "last"
reset.time	logical; if TRUE, then change time scale to start at t=0 and adjust start.age accordingly.
...	further arguments passed to read.table

Details

Function `as.paleoTS()` combines time series data into an object of class `paleoTS`. This function will usually not be used directly; `read.paleoTS()` is more convenient for getting the relevant data from text files.

If no file name is given for `read.paleoTS`, the user will be prompted to select a file using the `file.choose()` interactive prompt. Samples are assumed to be ordered from oldest to youngest (if not, use `oldest="last"`), with ages indicating time elapsed from the beginning of the sequence, rather than geological age before the present. If sample ages decrease through the sequence, as if given in millions of years ago, `tt` will automatically be converted to time elapsed from the beginning of the sequence. Generally, setting `reset.time = TRUE` is good practice.

Value

<code>mm</code>	vector of sample means
<code>vv</code>	vector of sample variances
<code>nn</code>	vector of sample sizes
<code>tt</code>	vector of sample ages
<code>MM</code>	vector of true population means (for simulated time series)
<code>genpars</code>	generating parameters (for simulated time series)
<code>label</code>	optional label describing time series data
<code>start.age</code>	optional, indicating age of oldest sample (e.g., in millions of years ago)

Note

The analysis estimates the contribution of sampling noise to the observed differences between samples. It does this assuming that one is analyzing sample means, which have sampling variances equal to variance divided by sample size, vv/nn . If one is interested in analyzing statistics other than the sample mean (medians, quantiles, or other statistics), use the the following procedure: set the statistic in question as the `mm` values, replace `vv` with a vector of the squared standard errors for each estimate, and set all values of `nn` to one.

Author(s)

Gene Hunt

References

Hunt, G. 2006. Fitting and comparing models of phyletic evolution: random walks and beyond. *Paleobiology* **32**:578–601.

See Also

[plot.paleoTS](#)

Examples

```
## create paleoTS object
y <- as.paleoTS(mm=rnorm(10), vv=rep(1,10), nn=rep(25,10), tt=1:10, label="white noise time series")
plot(y)
```

`as.paleoTSfit`*Class for fit to paleontological time-series models*

Description

Function combines useful information summarizing model fit.

Usage

```
as.paleoTSfit(logL, parameters, modelName, method, K, n, se)
```

Arguments

logL	log-likelihood of model
parameters	named vector with maximum-likelihood parameter estimates
modelName	name of the model
method	parameterization used: ancestor-descendant (AD) or Joint
K	number of parameters in the model
n	sample size
se	standard errors of parameters from the hessian (optional)

Details

This function is used by all the model-fitting routines (opt.XXX, fitXXX) to create standardized output

Value

A list with the above items

Note

This function is not likely to be called directly by the user.

Author(s)

Gene Hunt

See Also

[as.paleoTS](#)

cat.paleoTS

Miscellaneous functions used internally for punctuations

Description

These functions are used internally in the simulation and fitting of models with heterogeneous evolutionary dynamics. They generally will not be used directly by users.

Usage

```
cat.paleoTS(y)
split4punc(y, gg, overlap=TRUE)
shifts(ns, ng, minb=5)
shift2gg(ss, ns)
```

Arguments

y	a paleoTS object
gg	a vector of integers from 1 to the number of segments, indicating for each sample, the segment to which it belongs
overlap	if TRUE, the initial sample of each segment starts with the last sample of the previous segment
ns	the number of samples in the entire sequence
ng	the number of 'groups', i.e., independently evolving evolutionary segments
minb	the minimum number of samples to include in a segment
ss	an integer, or vector of integers, indicating the index of the initial sample of segments

Details

Function `cat.paleoTS` concatenates a list of multiple paleoTS objects into a single time-series. Note that the ages (`tt`) of the resulting time-series are read directly from the constituent time-series.

Function `split4punc` divides a paleoTS time-series into separate component time-series according to the grouping vector `gg`.

Function `shifts` returns a matrix of all allowable shift points for dividing `ns` samples into `ng` segments, subject to the constraint that each segment has at least `minb` samples. Function `shifts2gg` converts a vector of shift points into a grouping vector.

Value

Function `cat.paleoTS` returns a single paleoTS object, and `split4punc` returns a list of multiple paleoTS objects.

Function `shifts` returns a matrix of shift points—the index of samples that initiate new segments. This matrix has `ng-1` rows, where the `i`th row corresponds to the index of the initial sample for segment `i+1`. Each column represents one of the possible shift point combinations. Function `shift2gg` returns a grouping vector from these initial shift points.

Author(s)

Gene Hunt

References

- Hunt, G. 2006. Fitting and comparing models of phyletic evolution: random walks and beyond. *Paleobiology* **32**:578–601.
- Hunt, G. 2008. Gradual or pulsed evolution: when should punctuational explanations be preferred? *Paleobiology* **34**:360-377.

See Also

[fitGpunc](#), [sim.punc](#), [opt.GRW.shift](#)

Examples

```
# show all possible shift points for 20 samples divided into two segments (assuming each has at least 6 samples)
GG<- shifts(ns=20, ng=2, minb=6)
print (GG)
# show resulting grouping vector from first of these
gg<- shift2gg(GG[,1], ns=20)
print (gg)
```

compareModels

Compare output from any set of model fits

Description

Function compares a series of objects of paleoTSfit output from any of the model-fitting functions.

Usage

```
compareModels(..., silent = FALSE)
```

Arguments

... a series of one or more paleoTSfit objects, separated by commas
 silent if TRUE, results of model comparison will not be printed to the screen

Details

This function offers a convenient way to view the performance of multiple fitted models. Log-likelihoods, numbers of parameters (K), AICc scores and Akaike weights are returned in a table. The function checks that all model fits use the same method (parameterization; AD or Joint). If they do not, an error message results. Model fits are only comparable if they are based on the same parameterization.

Value

If silent=FALSE, returns a dataframe with the above information. If silent=TRUE, returns a list of the dataframe summarizing model fits along with the maximum-likelihood estimates for the parameters in each model.

Author(s)

Gene Hunt

See Also

[fit3models](#)

Examples

```
x<- sim.GRW(ns=40)
m1<- opt.URW(x)
m2<- opt.GRW(x)
m3<- fitGpunc(x, ng=2, minb=10)
compareModels(m1,m2,m3)
```

fit.sgs

Analyze evolutionary models with well-sampled punctuations

Description

Functions required to fit evolutionary models with sampled punctuations, i.e., where the transitional period is represented by at least several sampled populations.

Usage

```
fit.sgs(y, minb = 5, oshare = TRUE, pool = TRUE, silent = FALSE, hess = FALSE, meth = "L-BFGS-B", model =
opt.sgs(y, gg, cl = list(fnscale = -1), meth = "L-BFGS-B", hess = FALSE, oshare = TRUE, model = "GRW")
logL.sgs(p, y, gg, model = "GRW")
logL.sgs.omega(p, y, gg, model = "GRW")
```

Arguments

<code>y</code>	a paleoTS object
<code>minb</code>	the minimum number of samples within a segment to consider
<code>oshare</code>	logical, if TRUE, the same variance (ω) is assumed across the starting and ending Stasis segments. If FALSE, separate variances are assumed
<code>pool</code>	logical indicating whether to pool variances across samples
<code>silent</code>	if TRUE, less information is printed to the screen as the model is fit
<code>hess</code>	if TRUE, standard errors are computed from the Hessian matrix
<code>meth</code>	optimization method, to be passed to <code>optim</code>
<code>model</code>	either GRW or URW, indicating whether evolution during the transitional interval is directional (general random walk) or not (unbiased random walk)
<code>p</code>	parameters of the punctuation model for the log-likelihood functions
<code>gg</code>	numeric vector indicating membership of each sample in segments 1, 2, .. <code>ng</code>
<code>cl</code>	control list to be passed to <code>optim</code>

Details

These functions are used to fit a model with a sampled punctuation. Formally, this is a three-segment model that starts as Stasis, transitions to a punctuation of directional evolution (general random walk) or unconstrained (unbiased random walk). The name comes from an abbreviation of the three modes in the segments: Stasis - General Random Walk - Stasis, bearing in mind that the general random walk can be changed to an unbiased random walk. Users are likely only to use `fit.sgs`, which will call the other functions in order to find the best parameter estimates and shift points for the segments.

Value

The log-likelihood functions return the log-likelihood of the model for a given set of parameter values (`p`), assuming that the periods of Stasis have the same variance (`logL.punc.omega`) or different variances (`logL.punc`).

In addition to those specifying the dynamics of stasis and punctuation, parameters include `shift1`, the index of the last sample before the shift to GRW/URW, and `shift2`, the index of the last sample before the return to stasis.

Function `opt.sgs` returns a `paleoTSfit` object. Function `fit.sgs` does the same, but with the following additional elements:

<code>all.logl</code>	log-likelihoods for all tested partitions of the series into segments
<code>GG</code>	matrix of indices of initial samples of each tested segment configuration; each column of <code>GG</code> corresponds to the elements of <code>all.logl</code>

Author(s)

Gene Hunt

References

- Hunt, G. 2006. Fitting and comparing models of phyletic evolution: random walks and beyond. *Paleobiology* **32**:578–601.
- Hunt, G. 2008. Gradual or pulsed evolution: when should punctuational explanations be preferred? *Paleobiology* **34**:360–377.

See Also

[sim.sgs](#), [opt.GRW](#), [fitGpunc](#), [as.paleoTSfit](#)

Examples

```
x<- sim.sgs(ns=c(10, 10, 10), ms=0.5, vs=0.3, omega=0.1)
plot(x)
# compare sampled punctuation to uniform unbiased random walk
w.sgs<- fit.sgs(x, minb=8, model="GRW")
w.urw<- opt.URW(x)
compareModels(w.urw, w.sgs)
```

fit3models

Do model fits for three standard evolutionary models

Description

Functions fits three models to an evolutionary time series: (1) general random walk (=directional evolution), (2) unbiased random walk, and (3) stasis.

Usage

```
fit3models(y, silent = FALSE, method=c("AD", "Joint"), ...)
```

Arguments

y	a paleoTS object
silent	logical, if TRUE, results are not printed
method	parameterization to use: see Details
...	further arguments, passed to optimization functions

Details

The method argument refers to different parameterizations of the model. See the documentation under [opt.joint.GRW](#) for more information about the differences between these parameterizations.

Value

If `silent=FALSE`, function `fit3models` prints and returns a dataframe with log-likelihoods, numbers of parameters, AICc scores and Akaike weights for the three models. If `silent=TRUE`, a list is returned with element `'modelFits'` that has the same dataframe, plus an element `'parameters'` with the sub-elements for all the parameter estimates.

Note

For the `...` argument, only the arguments `pool` and `hess` will work; specifying `meth` in particular will cause an error. If finer control is needed, the optimization functions should be used individually.

Author(s)

Gene Hunt

References

Hunt, G. 2006. Fitting and comparing models of phyletic evolution: random walks and beyond. *Paleobiology* **32**:578–601.

Hunt, G. 2008. Evolutionary patterns within fossil lineages: model-based assessment of modes, rates, punctuations and process.. *In* R.K. Bambach and P.H. Kelley, eds. *From Evolution to Geobiology: Research Questions Driving Paleontology at the Start of a New Century*:578–601.

See Also

[opt.GRW](#), [opt.joint.GRW](#)

Examples

```
## show difference in parameterizations
### example 1, sequence with a strong trend ###
# two parameterizations usually yield similar Akaike weights under these conditions
x1<- sim.GRW(ns=10, ms=1, vs=0.5)
res1AD<- fit3models(x1, method='AD')
res1Joint<- fit3models(x1, method='Joint')

## example 2, longer & noisy directional walk
## joint parameterization often is better at correctly favoring GRW under these conditions
x2<- sim.GRW(ns=20, ms=0.2, vs=0.1) # step variance relatively low compared to sampling error == Noisy
res2AD<- fit3models(x2, method='AD')
res2Joint<- fit3models(x2, method='Joint')
```

fitGpunc

*Analyze evolutionary models with unsampled punctuations***Description**

Functions required to fit evolutionary models with punctuations that are rapid relative to the temporal spacing of samples (so-called unsampled punctuations).

Usage

```
fitGpunc(y, ng = 2, minb = 5, pool = TRUE, oshare = TRUE, method=c('AD', 'Joint'), silent = FALSE, hess=FALSE)
opt.punc(y, gg, cl = list(fnscale = -1), pool = TRUE, meth = "L-BFGS-B", hess = FALSE, oshare)
opt.joint.punc(y, gg, cl=list(fnscale=-1), pool=TRUE, meth="L-BFGS-B", hess=FALSE, oshare)
logL.punc(p, y, gg)
logL.punc.omega(p, y, gg)
logL.joint.punc(p, y, gg)
logL.joint.punc.omega(p, y, gg)
```

Arguments

y	a paleoTS object
ng	the number of separate segments in the sequence
minb	the minimum number of samples within a segment to consider
pool	logical indicating whether to pool variances across samples
oshare	logical, if TRUE, the same variance (omega) is assumed across all segments. If FALSE, separate variances are assumed for each segment
method	parameterization to use: based on ancestor-descendant (AD) differences, or on Joint consideration of all samples
silent	if TRUE, less information is printed to the screen as the model is fit
hess	if TRUE, standard errors are computed from the Hessian matrix
...	other arguments to send to <code>opt.punc</code>
p	parameters of the punctuation model to be optimized
gg	numeric vector indicating membership of each sample in segments 1, 2, .. ng
cl	control list to be passed to <code>optim</code>
meth	optimization method, to be passed to <code>optim</code>

Details

These functions are used to fit a model with an unsampled punctuation. It is equivalent to a Stasis model in which the optimum instantaneously shifts at one or more points in time; see references below for details. Users are likely only to use `fitGpunc`, which will call the other functions in order to find the best parameter estimates and shift points for the segments.

Value

The log-likelihood functions return the log-likelihood of the model for a given set of parameter values (p), assuming that the periods of Stasis have the same variance (`logL.punc.omega`) or different variances (`logL.punc`).

Functions `fitGpunc` and `opt.punc` return a list with the following elements:

<code>par</code>	parameter estimates
<code>value</code>	the log-likelihood of the optimal solution
<code>counts</code>	returned by <code>optim</code>
<code>convergence</code>	returned by <code>optim</code>
<code>message</code>	returned by <code>optim</code>
<code>p0</code>	initial guess for parameter values at start of optimization
<code>K</code>	number of parameters in the model
<code>n</code>	the number of observations, equal to the number of evolutionary transitions
<code>AIC</code>	Akaike information criterion
<code>AICc</code>	modified Akaike information criterion
<code>BIC</code>	Bayes (or Schwarz) information criterion
<code>se</code>	standard errors for parameter estimates, computed from the curvature of the log-likelihood surface (only if <code>hess = TRUE</code>)
<code>...</code>	other output from call to <code>optim</code>

In addition, function `fitGpunc` also returns the following elements:

<code>shift.start</code>	index of the last sample before a new segment
<code>all.logl</code>	log-likelihoods for all tested partitions of the series into segments
<code>GG</code>	matrix of indices of initial samples of each tested segment configuration; each column of <code>GG</code> corresponds to the elements of <code>all.logl</code>

Author(s)

Gene Hunt

References

- Hunt, G. 2006. Fitting and comparing models of phyletic evolution: random walks and beyond. *Paleobiology* **32**:578–601.
- Hunt, G. 2008. Gradual or pulsed evolution: when should punctuational explanations be preferred? *Paleobiology* **34**:360–377.

See Also

[sim.punc](#), [opt.GRW](#), [fit.sgs](#)

Examples

```
x<- sim.punc(theta=c(0,5), ns=c(20,20), omega=c(0.5,0.5), vp=c(0.2,0.2))
plot(x)
w<- fitGpunc(x, ng=2, minb=7, pool=TRUE, oshare=TRUE)
print (w$parameters)
## add lines to show the solution
segments(x$tt[1], w$parameters[1], x$tt[w$parameters[4]-1], w$parameters[1], lty=1, col="red", lwd=5)
segments(x$tt[w$parameters[4]], w$parameters[2], x$tt[length(x$tt)], w$parameters[2], lty=1, col="red", lwd=5)
```

IC

Compute information criterion scores and Akaike weights for evolutionary models

Description

These functions compute information criteria (IC) or Akaike weights based on information scores (akaike.wts).

Function IC is used internally by the optimization functions and generally will not need to be called directly by the user.

Usage

```
IC(logL, K, n = NULL, method = c("AICc", "AIC", "BIC"))
akaike.wts(aa)
```

Arguments

logL	log-likelihood
K	the number of free parameters
n	sample size for AICc and BIC calculations
method	which information criterion to compute; one of AIC, AICc, or BIC
aa	vector of AIC or AICc values used to compute Akaike weights

Details

These functions are used by the functions as.paleoTSfit and compareModels, and will rarely need to be used directly by the user.

Value

the computed information criterion, or a vector of Akaike weights

Author(s)

Gene Hunt

References

Hunt, G. 2006. Fitting and comparing models of phyletic evolution: random walks and beyond. *Paleobiology* **32**:578–601.

See Also

[opt.GRW](#), [compareModels](#), [as.paleoTSfit](#)

Examples

```
x <- sim.GRW(ns=40, ms=0.1, vs=0.1)
m1<- opt.GRW(x)
m2<- opt.URW(x)
aw<- akaike.wts(c(m1$AICc, m2$AIC)) # easier to use compareModels(m1,m2)
```

In.paleoTS

Log transform paleontological time series data

Description

This function performs an approximate log-transformation (base e) of a paleoTS object.

Usage

```
In.paleoTS(y)
```

Arguments

y a paleontological time series

Details

For a random variable x , its approximate mean on a log scale is the log of its untransformed mean. The approximate variance on a log scale is equal to the squared coefficient of variation.

Value

the converted paleoTS object.

Author(s)

Gene Hunt

References

Hunt, G. 2006. Fitting and comparing models of phyletic evolution: random walks and beyond. *Paleobiology* **32**:578–601.

Lewontin, R. 1966. On the measurement of relative variability. *Systematic Zoology* **15**:141–142.

See Also[std.paleoTS](#)**Examples**

```
y <- sim.GRW(20, 20, 1)
y1 <- ln.paleoTS(y)
print (y1)
```

`logL.GRW`*Compute log-likelihoods for random walk and stasis models*

Description

Returns log-likelihood for general random walk `logL.GRW`, unbiased random walk `logL.URW`, and stasis `logL.Stasis` models.

Usage

```
logL.GRW(p, y)
logL.URW(p, y)
logL.Stasis(p, y)
```

Arguments

<code>p</code>	vector of parameters
<code>y</code>	a paleoTS object

Details

For the general random walk, $p = c(mstep, vstep)$; for an unbiased random walk, $p = vstep$; for the stasis model, $p = c(theta, omega)$. In general, users will not be access these functions directly, but instead use the optimization functions, which use these functions to find the best-supported parameter values.

Value

The log-likelihood of the parameter estimates (p), given the data (y).

Author(s)

Gene Hunt

References

Hunt, G. 2006. Fitting and comparing models of phyletic evolution: random walks and beyond. *Paleobiology* **32**:578–601.

See Also

[mle.GRW](#), [opt.GRW](#)

Examples

```
y<- sim.GRW(20, 0, 1)
L1 <- logL.GRW(p=c(0,1), y) # actual parameters
L2 <- logL.GRW(p=c(10,10), y) # should be a bad guess
cat (L1, L2, "\n")
```

logL.joint.GRW

Log-likelihoods for evolutionary models (joint parameterization)

Description

Returns log-likelihood for general random walk (`logL.joint.GRW`), unbiased random walk (`logL.joint.URW`), stasis (`logL.joint.Stasis`) and OU (`logL.joint.OU`) models.

Usage

```
logL.joint.GRW(p, x)
logL.joint.URW(p, x)
logL.joint.Stasis(p, x)
logL.joint.OU(p, x)
```

Arguments

`p` a vector of parameters
`x` a paleoTS object

Details

For the general random walk, $p = c(\text{anc}, \text{mstep}, \text{vstep})$; for an unbiased random walk, $p = c(\text{anc}, \text{vstep})$; for the stasis model, $p = c(\text{theta}, \text{omega})$, and for the OU model $p = c(\text{anc}, \text{vstep}, \text{theta}, \text{alpha})$. In general, users will not be access these functions directly, but instead use the optimization functions, which use these functions to find the best-supported parameter values.

Value

The log-likelihood of the parameter estimates (p), given the data (x).

Warning

Because these functions parameterize the models differently, their log-likelihoods are not comparable to those that do not use the joint parameterization.

Author(s)

Gene Hunt

References

- Hunt, G. 2006. Fitting and comparing models of phyletic evolution: random walks and beyond. *Paleobiology* **32**:578–601.
- Hunt, G., M. Bell & M. Travis. 2008. Evolution towards a new adaptive optimum: phenotypic evolution in a fossil stickleback lineage. *Evolution* **62**:700-710.
- Hunt, G. 2008. Evolutionary patterns within fossil lineages: model-based assessment of modes, rates, punctuations and process.. In R.K. Bambach and P.H. Kelley, eds. From Evolution to Geobiology: Research Questions Driving Paleontology at the Start of a New Century:578–601.

See Also[opt.joint.GRW](#), [logL.GRW](#)**Examples**

```
x<- sim.GRW(ns=20, ms=0, vs=1)
L1<- logL.joint.GRW(p=c(0,0,1), x) # actual parameters
L2<- logL.joint.GRW(p=c(0,10,1), x) # should be a bad guess
cat(L1, L2, "\n")
```

LRI

Log-Rate, Log-Interval (LRI) method of Gingerich

Description

Gingerich (1993) introduced a method that plots on log-log scale, the rate and interval for each pair of samples in an evolutionary sequence. On this plot, the slope is interpreted as an indicator of evolutionary mode (-1 for stasis, 0.5 for random walk, 0 for directional), and the intercept is interpreted as a measur of the rate of evolution over one generation.

Usage

```
LRI(x, gen.per.t = 1e+06, draw = TRUE)
```

Arguments

x	a paleoTS object
gen.per.t	the number of generations per unit time (e.g., 1e6 for yearly generations and time in x is in Myr)
draw	logical, if TRUE, a plot is produced

Details

Following Gingerich (1993), a robust line is fit through the points by minimizing the sum of absolute deviations.

Value

A named vector of three elements: Intercept, slope and GenerationalRate

Note

This method was important in attempts to disentangle evolutionary tempo and mode. I view likelihood-based methods as more informative, and in particular the estimation of 'Generational Rates' using LRI is compromised by sampling error (see the example).

Author(s)

Gene Hunt

References

- Gingerich, P.D. 1993. Quantification and comparison of evolutionary rates. *American Journal of Science* **293-A**:453–478.
- Gingerich, P.D. 2009. Rates of evolution. *Annual Review of Ecology Evolution and Systematics* **40**:657–675.

See Also

[fit3models](#)

Examples

```
xFast<- sim.GRW(ns=20, ms=0.5, vs=0.2) # fast evolution
xSlow<- sim.Stasis(ns=20, theta=10, omega=0) # strict stasis! rates are actually zero
wFast<- LRI(xFast, draw=FALSE)
wSlow<- LRI(xSlow, draw=FALSE)
## LRI usually assigns faster generational rate to Strict Stasis!
print(wFast[3],4)
print(wSlow[3],4)
```

lynchD

Compute rate metric from Lynch (1990)

Description

This function computes D, the rate metric proposed by Lynch (1990). This metric derives from the random walk model, with $D = \sqrt{v_{step}/(2V_p)}$, where v_{step} is the step variance of the unbiased random walk, and V_p is the within sample variance, pooled among samples. Under mutation - drift equilibrium, D is expected to range approximately between $5e-5$ and $5e-3$.

Usage

```
lynchD(y, gen.per.t = 1e+06, pool = TRUE, method=c('AD', 'Joint'), ...)
```

Arguments

y	a paleoTS object
gen.per.t	the number of generations per unit time
pool	logical, if the variance should be pooled across samples in estimating Vstep
method	parameterization to use: based on ancestor-descendant (AD) differences, or on Joint consideration of all samples
...	further arguments, passed to opt.URW or opt.joint.URW

Details

The `gen.per.t` argument indicates the number of organismal generations for each unit of time with which the time-series `y` was measured. For example, if `y$tt` is measured in millions of years, and the species has annual generations, `gen.per.t` would be `1e6`. For difference between the method choices, see Hunt (2008).

Estimation of `Vstep` is done on the original time scale of `y`, and then the values are converted to generational time scales using `gen.per.t`. This is for numerical reasons, as it avoids computations on possible very low numbers.

Value

D	value of rate metric
pooled.var	value of pooled within-sample variance
gen.per.t	number of generations per unit time
vstep	computed <code>Vstep</code> , at the original time scale of <code>y</code>
drift.range	expected minimum and maximum values of D consistent with neutral evolution
result	conclusion reached about the plausibility of neutral evolution

Author(s)

Gene Hunt

References

Lynch, M. 1990. The rate of morphological evolution in mammals from the standpoint of the neutral expectation. *The American Naturalist* **136**:727–741.

Hunt, G. 2008. Evolutionary patterns within fossil lineages: model-based assessment of modes, rates, punctuations and process.. In R.K. Bambach and P.H. Kelley, eds. *From Evolution to Geobiology: Research Questions Driving Paleontology at the Start of a New Century*:578–601.

See Also

[opt.URW](#)

Examples

```
y<- sim.GRW(ns=20, ms=0, vs=1e-4, tt=seq(0,1e6, length.out=20)) # per-year simulation
lynchD(y, gen.per.t=1) # 1 generation per year
```

mle.GRW

Maximum likelihood parameter estimators

Description

These functions calculate maximum likelihood estimators for the general random walk (mle.GRW), unbiased random walk (mle.URW) and stasis (mle.Stasis) models.

Usage

```
mle.GRW(y)
mle.URW(y)
mle.Stasis(y)
```

Arguments

y a paleoTS object

Details

These functions return maximum likelihood estimators for the general random walk mle.GRW, unbiased random walk mle.URW and stasis mle.Stasis models, but **only under a restricted set of circumstances are these valid!** For these estimators to be valid, the sampling error must be the same in all samples, which generally means equal sample size and variances in all samples. For the random walk models, it is also assumed that samples are evenly spaced in time. Because these assumptions usually do not hold for paleontological data, almost all users should instead use the numerical optimization functions (see [opt.GRW](#)). The main purpose for the present functions is to provide starting estimates for numerical optimization.

Value

a vector of parameter estimates, either c(mstep, vstep), or c(theta, omega)

Author(s)

Gene Hunt

References

Hunt, G. 2006. Fitting and comparing models of phyletic evolution: random walks and beyond. *Paleobiology* **32**:578–601.

See Also

[opt.GRW](#), [logL.GRW](#)

Examples

```
## Warning: better to use opt.GRW() for real data
y <- sim.GRW(ns=20, ms=0, vs=1)
mle.GRW(y)
```

modelCurves

Function computes model expectations and 95

Description

This function is used by `plot.paleoTS` to add the model expectations and probability intervals for (some) models. It is not likely to be called directly by the user.

Usage

```
modelCurves(x, w, np = 500)
```

Arguments

x	a paleoTS object
w	fitted model results in the form of a paleoTSfit object
np	the number of points to use for plotting the expectation

Details

The function determines what to draw from the `modelName` of `w`, along with its parameter values.

Value

A list with the following vectors: `tt`, the times plotted; `ee`, the model expectations; `ll` and `uu`, the lower and upper 95 percent probability limits for the model.

Note

If the `modelName` of `w` is not recognized, `ee` is set to `NA` and nothing is added to the plot.

Author(s)

Gene Hunt

Examples

```
x1<- sim.punc(theta=c(0,10), omega=c(2,2))
w1<- fitGpunc(x1, ng=2)
plot(x1, modelFit=w1)
```

```
x2<- sim.GRW(ns=20, ms=1, vs=0.4)
w2<- opt.GRW(x2)
plot(x2, modelFit=w2)
```

opt.covTrack

Covariate-tracking model

Description

Functions to fit a model in which a phenotypic trait tracks changes in a measured covariate (e.g., body size tracks temperature changes).

Usage

```
opt.covTrack(y, z, pool = TRUE, cl = list(fnscale = -1), meth = "L-BFGS-B", hess = FALSE)
opt.covTrack.Mult(y1, z1, cl = list(fnscale = -1), pool = TRUE, hess = FALSE)
logL.covTrack(p, y, z)
logL.Mult.covTrack(p, y1, z1)
```

Arguments

y	a paleoTS object
z	a measured covariate. See Details about its required length.
pool	logical, if TRUE, variance are pooled across samples
cl	control list, passed to function optim
meth	optimization method, passed to function optim
hess	logical, indicating whether to calculate standard errors from the Hessian matrix
y1	a list of paleoTS objects
z1	a list of covariates, corresponding to y1
p	a vector of parameter values

Details

These functions model the changes in a phenotypic trait as linear functions of changes in a covariate, z. Each change in z ($=dz$) yields an expected change in the trait equal to $b*dz$, with normally distributed residual variance *evan*. If z is equal in length to the trait sequence, first differences are taken. If z is of length one less than the trait sequence, it is assumed that the user has already taken the appropriate difference.

For the Mult versions of these, the function estimates the model assuming the *same* relationship exists across all sequences.

Value

An object of class `paleoTSfit`

Note

These functions use the "AD" parameterization.

Author(s)

Gene Hunt

References

Hunt, et al. 2010. Climate-driven body-size trends in the ostracod fauna of the deep Indian Ocean, *Palaeontology* **53**:1255–1268.

See Also

[as.paleoTSfit](#), [opt.GRW](#)

Examples

```
z<- rnorm(20)
x<- sim.covTrack(ns=20, b=2, evar=0.1, z=z)
plot(diff(z), diff(x$mm), xlab="Change in covariate", ylab="Change in Trait")
abline(h=0, lty=3)
abline(v=0, lty=3)
mct<- opt.covTrack(x, z)
print(round(mct$parameters,2)) # should be close to generating values
print(x$genpar)
```

opt.GRW

Numerically find maximum likelihood solutions to evolutionary models

Description

Functions to find maximum likelihood solutions to general random walk (`opt.GRW`), unbiased random walk `opt.URW`, and stasis models `opt.Stasis`.

Usage

```
opt.GRW(y, pool = TRUE, cl = list(fnscale = -1), meth = "L-BFGS-B", hess = FALSE)
opt.URW(y, pool = TRUE, cl = list(fnscale=-1), meth = "L-BFGS-B", hess = FALSE)
opt.Stasis(y, pool = TRUE, cl = list(fnscale=-1), meth = "L-BFGS-B", hess = FALSE)
```

Arguments

y	a paleoTS object
cl	control list, passed to function <code>optim</code>
pool	logical indicating whether to pool variances across samples
meth	optimization method, passed to function <code>optim</code>
hess	logical, indicating whether to calculate standard errors from the Hessian matrix

Details

These functions numerically search a log-likelihood surface for its optimum—they are a convenient wrapper to `optim`. Arguments `meth`, `cl`, and `hess` are passed to `optim`; see that function's help for details. These are included to allow sophisticated users greater control over the optimization; the defaults seem to work well for most, but not all sequences. For `meth="L-BFGS-B"`, some parameters are constrained to be non-negative, which is useful parameters which cannot truly be negative, such as `vstep` (random walk) and `omega` (stasis model).

Initial estimates to start the optimization come from analytical solutions based on assuming equal sampling error across samples and evenly spaced samples in time (functions `mle.GRW`, `mle.URW` and `mle.Stasis`).

Value

A `paleoTSfit` object.

Note

Standard errors computed from the Hessian matrix are reasonably accurate for `mstep` and `theta`, but not as useful for the `vstep` and `omega` because of the asymmetry of the log-likelihood surfaces.

Author(s)

Gene Hunt

References

Hunt, G. 2006. Fitting and comparing models of phyletic evolution: random walks and beyond. *Paleobiology* **32**:578–601.

See Also

[logL.GRW](#), [fit3models](#), [opt.RW.Mult](#), [sim.GRW](#), [as.paleoTSfit](#)

Examples

```
## generate data for a directional sequence
y <- sim.GRW(ns=30, ms=1, vs=1)
plot(y)
m.rw<- opt.GRW(y)
```

```

m.rwu<- opt.URW(y)
m.sta<- opt.Stasis(y)

## print log-likelihoods; easier to use function fit3models()
compareModels(m.rw, m.rwu, m.sta)

```

opt.GRW.shift

Functions for random walks with shifting parameters

Description

Functions to simulate and to infer a model with random walk dynamics, with parameter values that shift at one or more points in the sequence.

Usage

```
opt.GRW.shift(y, ng = 2, minb = 5, model = 1, pool = TRUE, silent = FALSE)
```

Arguments

y	a paleoTS object
ng	the number of different segments in the sequence
minb	the minimum number of samples to consider as a segment
model	options for variants of random walk to fit (see Details).
pool	if TRUE, pool phenotypic variances across samples
silent	if TRUE, do not print information on fitting to screen

Details

This model divides an evolutionary sequence into two or more non-overlapping parts called segments, and then fits some version of the random walk model to each segment separately.

The model argument has four options:

model=1, general random walk model, step variance shared across segments

model=2, general random walk model, step mean shared across segments

model=3, unbiased random walk, separate step variance for each segment
 model=4, general random walk, separate step mean and variance for each segment

Value

A paleoTSfit object.

Author(s)

Gene Hunt

References

- Hunt, G. 2006. Fitting and comparing models of phyletic evolution: random walks and beyond. *Paleobiology* **32**:578–601.
- Hunt, G. 2008. Gradual or pulsed evolution: when should punctuational explanations be preferred? *Paleobiology* **34**:360–377.

See Also

[sim.GRW.shift](#), [opt.GRW](#), [opt.RW.Mult](#), [as.paleoTSfit](#)

Examples

```
x<- sim.GRW.shift(ns=c(20,20), ms=c(0,1), vs=c(0.2, 0.2))
plot(x)
w.shift<- opt.GRW.shift(x, ng=2, model=1)
print (w.shift$par)
print (w.shift$shift.start) # estimated first sample in second segment
```

opt.joint.GRW

Optimize evolutionary models (joint parameterization)

Description

Functions to find maximum likelihood solutions to general random walk (`opt.joint.GRW`), unbiased random walk (`opt.joint.URW`), stasis (`opt.joint.Stasis`) and OU models (`opt.joint.OU`).

Usage

```
opt.joint.GRW(x, pool = TRUE, cl = list(fnscale = -1), meth = "L-BFGS-B", hess = FALSE)
opt.joint.URW(x, pool = TRUE, cl = list(fnscale = -1), meth = "L-BFGS-B", hess = FALSE)
opt.joint.Stasis(x, pool = TRUE, cl = list(fnscale = -1), meth = "L-BFGS-B", hess = FALSE)
opt.joint.OU(x, pool = TRUE, cl = list(fnscale = -1), meth = "L-BFGS-B", hess = FALSE)
```

Arguments

<code>x</code>	a paleoTS object
<code>pool</code>	logical indicating whether to pool variances across samples
<code>cl</code>	control list, passed to function <code>optim</code>
<code>meth</code>	optimization method, passed to function <code>optim</code>
<code>hess</code>	logical, indicating whether to calculate standard errors from the Hessian matrix

Details

These functions numerically search a log-likelihood surface for its optimum—they are a convenient wrapper to `optim`. Arguments `meth`, `cl`, and `hess` are passed to `optim`; see the help for that function for details. These are included to allow sophisticated users greater control over the optimization; the defaults seem to work well for most, but not all sequences. For `meth="L-BFGS-B"`, some parameters are constrained to be non-negative, which is useful parameters which cannot truly be negative, such as `vstep` (random walk) and `omega` (stasis model).

Initial estimates to start the optimization come in part from analytical solutions based on assuming equal sampling error across samples and evenly spaced samples in time (functions `mle.GRW`, `mle.URW` and `mle.Stasis`).

Value

An object of class `paleoTSfit`

Warning

Measures of model fit (log-likelihoods, AIC scores, etc) are not comparable between the two parameterizations.

Note

These optimizations are performed using a parameterization of the GRW, URW and Stasis models that differs from the functions `opt.GRW`, `opt.URW`, and `opt.Stasis`. Those functions models are fit from the *differences* between adjacent samples, removing the autocorrelation in the time-series. In the joint parameterization implemented in functions `opt.joint.GRW`, `opt.joint.URW`, `opt.joint.Stasis` and `opt.joint.OU`, models are fit using the actual sample values, with the autocorrelation among samples accounted for in the log-likelihood function. For each model, the joint distribution of sample means is multivariate normal, with means and variance-covariances determined by evolutionary parameters and sampling errors. Note that the Orstein-Uhlenbeck model is at present only implemented using the joint parameterization.

For details on this parameterization of the models, see the paper Hunt, et al (2008). In general, the two different parameterizations yield similar results as to the relative support for different models. In my experience, the two approaches tend to differ appreciably only with relatively long sequences that have small differences between consecutive samples. In these circumstances, the alternative parameterization is better able to distinguish true evolutionary patterns from sampling error, and is less prone to falsely favoring the Stasis model.

The general random walk (GRW) and unbiased random walk (URW) models require an additional parameter that specifies the phenotype at the start of the sequence. This parameter is called `anc` in the vector of returned parameter estimates, and in general is rather similar to the trait value of the first sample in the sequence. This extra parameter is not necessary for the stasis model because the initial value does not figure into the likelihood calculations (which are completely determined by the position of optimum and the variance around this optimum).

Author(s)

Gene Hunt

References

- Hunt, G. 2006. Fitting and comparing models of phyletic evolution: random walks and beyond. *Paleobiology* **32**:578–601.
- Hunt, G., M. Bell & M. Travis. 2008. Evolution towards a new adaptive optimum: phenotypic evolution in a fossil stickleback lineage. *Evolution* **62**:700-710.
- Hunt, G. 2008. Evolutionary patterns within fossil lineages: model-based assessment of modes, rates, punctuations and process.. *In* R.K. Bambach and P.H. Kelley, eds. *From Evolution to Geobiology: Research Questions Driving Paleontology at the Start of a New Century*:578–601.

See Also

[logL.joint.GRW](#), [opt.GRW](#), [as.paleoTSfit](#)

Examples

```
x<- sim.GRW(ns=30, ms=1, vs=1)
plot(x)

# easier to use fit3models(, method='Joint')
m.urw<- opt.joint.URW(x)
m.grw<- opt.joint.GRW(x)
m.sta<- opt.joint.Stasis(x)
cat(m.urw$AICc, m.grw$AICc, m.sta$AICc, "\n") # print AICc scores
```

 opt.RW.Mult

Functions to analyze multiple time-series jointly

Description

These functions are used to estimate parameters of the general random walk over two or more sequences. Three different models may be fit: the same parameters are present in all time series (opt.RW.mult); the same mean step but different step variances in each series (opt.RW.SameMs); the same step variance but different mean steps in each series (opt.RW.SameVs).

Usage

```
logL.Mult(p, y, model = c("GRW", "URW"))
logL.SameMs(p, y)
logL.SameVs(p, y)
opt.RW.Mult(y1, cl=list(fnscale=-1), model=c("GRW", "URW"), pool=TRUE, meth="L-BFGS-B", hess=FALSE)
opt.RW.SameMs(y1, cl=list(fnscale=-1), pool=TRUE, meth="L-BFGS-B", hess=FALSE)
opt.RW.SameVs(y1, cl=list(fnscale=-1), pool=TRUE, meth="L-BFGS-B", hess=FALSE)
```

Arguments

p	a vector of parameter values
y	a <i>list</i> of paleoTS objects.
y1	a <i>list</i> of paleoTS objects.
model	RW for general random walk, RWu for unbiased random walk
pool	logical, if TRUE variances are pooled across samples
c1	optimization option, passed to <code>optim</code>
meth	optimization option, passed to <code>optim</code>
hess	optimization option, passed to <code>optim</code>

Details

These functions work just as their counterparts for the analysis of single sequences; see those help functions for more detail.

Value

Varies by function, see corresponding functions for the analysis of single sequences for more information.

Author(s)

Gene Hunt

References

Hunt, G. 2006. Fitting and comparing models of phyletic evolution: random walks and beyond. *Paleobiology* **32**:578–601.

See Also

[logL.GRW](#), [opt.GRW](#), [fit3models](#)

Examples

```
## create two sequences, with different parameter values
y1<- sim.GRW(ns=20, ms=0, vs=1)
y2<- sim.GRW(ns=20, ms=0, vs=0.2)

## fit some models with at least some shared dynamics across sequences
m1<- opt.RW.Mult(list(y1,y2), model="GRW")
m2<- opt.RW.SameMs(list(y1, y2))
m3<- opt.RW.SameVs(list(y1,y2))

## fit separate models to each sequence
compareModels(m1, m2, m3)
```

plot.paleoTS	<i>Plots paleoTS objects</i>
--------------	------------------------------

Description

Plots paleontological time series, showing trait means (with error bars) over time.

Usage

```
## S3 method for class 'paleoTS'  
plot(x, nse = 1, pool = FALSE, add = FALSE, modelFit=NULL, pch=19, lwd=1.5, ylim=NULL, ...)
```

Arguments

x	a paleoTS object
nse	size of error bars, in standard errors
pool	logical, indicating if should pool variances among samples in computing standard errors
add	logical, if TRUE data are added to existing plot
modelFit	optional paleoTSfit object with model fit results to add to plot
pch	plotting symbol
lwd	line width for plotting
ylim	y limit for plotting (optional)
...	other arguments sent to plot

Details

If modelFit is not NULL, then the expectations of that model, plus 95

Value

No values are returned, the results are plotted.

Author(s)

Gene Hunt

See Also

[as.paleoTS](#)

Examples

```
y <- sim.GRW(50, 0, 1)  
plot(y)  
ys <- sub.paleoTS(y, k=0.2) # same sequence, sub-sampled  
plot (ys, add=TRUE, col="red")
```

`sim.covTrack`*Simulate time-series that tracks a covariate*

Description

Function to simulate a time-series of trait values in which each evolutionary increment is linearly related to changes in a covariate

Usage

```
sim.covTrack(ns = 20, b = 1, evar = 0.1, z, nn = rep(20, times = ns), tt = 0:(ns-1), vp = 1)
```

Arguments

<code>ns</code>	number of samples in time-series
<code>b</code>	slope of the relationship between evolutionary changes and changes in the covariate
<code>evar</code>	residual variance around covariate-trait relationship
<code>z</code>	a measured covariate
<code>nn</code>	vector of the number of individuals in each sample
<code>tt</code>	vector of sample ages, increases from oldest to youngest
<code>vp</code>	within-population trait variance

Details

See `opt.covTrack` for a description of the model.

Value

An object of class `paleoTSfit`

Author(s)

Gene Hunt

References

Hunt, et al. 2010. Climate-driven body-size trends in the ostracod fauna of the deep Indian Ocean, *Palaeontology* **53**:1255–1268.

See Also

[opt.covTrack](#), [as.paleoTSfit](#)

Examples

```

z<- rnorm(20)
x<- sim.covTrack(ns=20, b=2, evar=0.3, z)
plot(diff(z), diff(x$mm), xlab="Change in covariate", ylab="Change in Trait")
abline(h=0, lty=3)
abline(v=0, lty=3)

```

sim.GRW

*Simulate evolutionary time-series***Description**

Simulate the evolution of a trait according to general random walk or stasis models.

Usage

```

sim.GRW(ns = 20, ms = 0, vs = 0.1, vp = 1, nn = rep(20, ns), tt = 0:(ns-1))
sim.Stasis(ns = 20, theta = 0, omega = 0, vp = 1, nn = rep(20,ns), tt = 0:(ns-1))

```

Arguments

ns	number of samples in time-series
ms	mean of the step distribution, random walk model
vs	variance of the step distribution, random walk model
vp	within-population trait variance
nn	vector of the number of individuals in each sample
tt	vector of sample ages, increases from oldest to youngest
theta	evolutionary optimum, stasis model
omega	evolutionary variance, stasis model

Details

See reference below for details on parameterization of the models.

Briefly, the general random walk model considers time in discrete steps. The duration of steps does not matter as long as many steps occur between sampled populations. At each time step, an evolutionary change is drawn at random from a distribution of possible evolutionary "steps." It turns out that the long-term dynamics of an evolving lineage depend only on the mean and variance of this step distribution. The former, *mstep*, determined the directionality in a sequence and the latter, *vstep*, determines its volatility.

The stasis model is based on the parameterization of Sheets and Mitchell (2001). Under this model, there is an evolutionary optimum, *theta*, with some amount of true variance, *omega*, around this optimum.

Value

A paleoTS object.

Author(s)

Gene Hunt

References

Hunt, G. 2006. Fitting and comparing models of phyletic evolution: random walks and beyond. *Paleobiology* **32**:578–601.
 Sheets, H. D., and C. E. Mitchell. 2001. Why the null matters: statistical tests, random walks and evolution. *Genetica* **112-113**:105-125.

See Also

[opt.GRW](#)

Examples

```
## generate and plot two paleoTS objects
y.rw <- sim.GRW(ns=20, ms=0.5, vs=0.1)
y.st <- sim.Stasis(ns=20)
layout(1:2)
plot(y.rw, col="red")
plot(y.st, col="blue")
layout(1)
```

sim.OU

Simulate evolutionary time-series

Description

Generates an evolutionary time-series according to an Orstein-Uhlenbeck (OU) model.

Usage

```
sim.OU(ns = 20, anc = 0, theta = 10, alpha = 0.3, vs = 0.1, vp = 1, nn = rep(20, ns), tt = 0:(ns-1))
ou.M(anc, theta, aa, tt)
ou.V(vs, aa, tt)
```

Arguments

ns	number of samples in time-series
anc	ancestral phenotype at the start of the series
theta	phenotype of the evolutionary optimum
alpha	strength of the attracting force pulling the population to the optimum

vs	step variance of the random walk component of change
vp	within-population trait variance
nn	vector of the number of individuals in each sample
tt	vector of sample ages, increases from oldest to youngest
aa	strength of the attracting force pulling the population to the optimum (same as alpha)

Details

See Hansen (1997) for a description of this model in a macroevolutionary context. This model also arises naturally in microevolution as a finite population evolving in the vicinity of an optimum in the adaptive landscape; see Lande (1976) and Estes & Arnold (2007).

Functions `ou.M` and `ou.V` are used internally by `sim.OU` to generate the means and variances of an OU process.

Value

A paleoTS object for `sim.OU`. For `ou.M` and `ou.V`, a vector of means or variances, respectively, are generated.

Author(s)

Gene Hunt

References

- Lande, R. 1976. Natural selection and random genetic drift in phenotypic evolution. *Evolution* **30**:314-334.
- Hansen, T. 1997. Stabilizing selection and the comparative analysis of adaptation. *Evolution* **51**:1341-1351.
- Estes, S. & Arnold, S. J. 2007. Resolving the paradox of stasis: models of stabilizing selection explain evolutionary divergence on all timescales. *American Naturalist* **169**:227-244.
- Hunt, G., M. Bell & M. Travis. 2008. Evolution towards a new adaptive optimum: phenotypic evolution in a fossil stickleback lineage. *Evolution* **62**:700–710.

See Also

[sim.GRW](#), [opt.joint.OU](#)

Examples

```
x1<- sim.OU(ns=100, anc=0, theta=10, alpha=0.2, vs=0.1, vp=0.1, nn=rep(100, times=100), tt=0:99)
plot(x1)
```

sim.punc

*Simulate evolutionary time-series with changing dynamics***Description**

These functions simulate evolutionary time-series in which the dynamics are not constant over time. These correspond to unsampled punctuations (`sim.punc`), sampled punctuations (`sim.sgs`), and random walks with parameter values change over time.

Usage

```
sim.punc(ns = c(10, 10), theta = c(0, 1), omega=rep(0,length(theta)), nn = rep(30, sum(ns)), tt = 0:(sum(ns)-1), v)
sim.sgs(ns = c(20, 20, 20), theta = 0, omega = 1, ms = 1, vs = 0.1, nn=rep(30, sum(ns)), tt = 0:(sum(ns)-1), v)
sim.GRW.shift(ns = c(10, 10), ms = c(0, 1), vs = c(0.5, 0.5), nn = rep(30, sum(ns)), tt = 0:(sum(ns)-1), v)
```

Arguments

<code>ns</code>	vector with the number of samples in each segment
<code>theta</code>	evolutionary optimum, Stasis model
<code>omega</code>	evolutionary variance, Stasis model
<code>nn</code>	a vector of sample sizes per sample
<code>tt</code>	vector of sample ages, increasing from oldest to youngest
<code>vp</code>	within-sample phenotypic variance
<code>ms</code>	mean step in GRW model for punctuated interval
<code>vs</code>	step variance in GRW model for punctuated interval

Details

These three models are elaborations of the standard random walk, directional evolution and stasis models to allow for heterogeneous evolutionary dynamics within a sequence. These heterogeneous dynamics include sampled punctuations (`sim.sgs`), unsampled punctuations `sim.punc` and shifting random walk dynamics (`sim.GRW.shift`).

Value

A paleoTS object.

Author(s)

Gene Hunt

References

Hunt, G. 2006. Fitting and comparing models of phyletic evolution: random walks and beyond. *Paleobiology* **32**:578–601.
 Hunt, G. 2008. Gradual or pulsed evolution: when should punctuational explanations be preferred? *Paleobiology* **34**:360-377.

See Also

[opt.punc](#), [opt.sgs](#), [opt.GRW.shift](#)

Examples

```
# illustrate differences between sampled and unsampled punctuations
x1<- sim.sgs(ms=0.4, omega=0.5)
x2<- sim.punc(ns=c(30,30), theta=c(0, mean(x1$mm[41:60])), omega=c(0.5,0.5))
layout(1:2)
plot(x1, nse=2, col="red", main="Sampled Punctuation")
rect(20,min(x1$mm), 40, max(x1$mm))
text(30, max(x1$mm), "sampled\ntransition", font=3, pos=1, cex=0.8)
plot(x2, nse=2, col="black", main="Unsampled Punctuation")
rect(30,min(x2$mm),31,max(x2$mm))
text(31, mean(x2$mm), "unsampled\ntransition", font=3, pos=4, cex=0.8)
```

std.paleoTS

Standardize paleontological time series data

Description

This function standardizes a paleoTS object, converting the data to within-populations standard deviation units.

Usage

```
std.paleoTS(y, zero = "start")
```

Arguments

y	a paleontological time series
zero	if zero = "start", trait values are translated such that the initial starting value of the sequence is set to be zero. All other values are ignored.

Details

The standardization expresses each sample mean as the deviation from the overall mean, divided by the pooled within-sample variance. Sample variances are also divided by the pooled sample variance.

Essentially, this converts paleontological time-series data into standard deviation units, similar to the computation of evolutionary rates in haldanes. This operation *does not* change the relative fit of models, but it does facilitate the comparison of parameter estimates across time-series with different units.

Value

the converted paleoTS object.

Author(s)

Gene Hunt

References

Hunt, G. 2006. Fitting and comparing models of phyletic evolution: random walks and beyond. *Paleobiology* **32**:578–601.

See Also

[ln.paleoTS](#)

Examples

```
y<- sim.GRW(20, 0, 1, vp=10)
ys<- std.paleoTS(y)
fit3models(y)
fit3models(ys) # note Akaike weights do not change from standardization
```

Stickleback data

Stickleback data from Bell et al. (2006)

Description

Dorsal spine, pterygiophore, and pelvic score data from a fossil stickleback lineage.

Usage

```
data(dorsal.spines)
data(ptyrygiophores)
data(pelvic.score)
```

Format

paleoTS objects for time-series of stickleback data.

Details

These data are already in the form of a paleoTS object, with sample means (mm), variances (vv) sample sizes (nn), and ages in Kyr (tt).

Source

Bell, M.A., M.P. Travis and D.M. Blouw 2006. Inferring natural selection in a fossil threespine stickleback. *Paleobiology* **32**:562-577.

Examples

```

data(dorsal.spines)

# get subset of samples from invading lineage (tt>=4.5 Kyr), only those with nn>=5
ok<- dorsal.spines$tt >= 4.5 & dorsal.spines$nn >=5
ds2<- sub.paleoTS(dorsal.spines, ok=ok)

# convert time scale to generations (500 gen per Kyr), reset starting time to t=0
ds2$tt<- ds2$tt*(1000/2)
ds2$tt<- ds2$tt - ds2$tt[1]

# compare URW (drift) and OU (adaptive) models
m.urw<- opt.joint.URW(ds2, pool=TRUE)
m.ou<- opt.joint.OU(ds2, pool=TRUE)
compareModels(m.urw, m.ou)

plot(ds2, modelFit=m.ou, pool=TRUE)

```

sub.paleoTS

Subset an evolutionary time series

Description

This function subsets a paleoTS object, returning a shorter series of specified or random samples from the original.

Usage

```
sub.paleoTS(y, ok = NULL, k = 0.1)
```

Arguments

y	a paleoTS object
ok	vector indicating which samples to retain; can be logical or numeric
k	proportion of samples to retain, chosen randomly

Details

If ok = NULL, the subsetting is done by retaining a proportion k of the initial samples, chosen at random. If ok is specified, those samples indicated will be retained.

Value

the subbetted paleoTS object

Author(s)

Gene Hunt

References

Hunt, G. 2006. Fitting and comparing models of phyletic evolution: random walks and beyond. *Paleobiology* **32**:578–601.

See Also

[as.paleoTS](#)

Examples

```
y <- sim.GRW(ns=100, ms=0, vs=0.3)
plot(y, col='grey')
ys1<- sub.paleoTS(y, ok=y$mm > mean(y$mm)) # subsetting with logical ok
ys2<- sub.paleoTS(y, ok=1:10) # subsetting with numeric ok
ys3<- sub.paleoTS(y, k=0.1) # subsetting random 10%
plot(ys1, add=TRUE, col="red")
plot(ys2, add=TRUE, col="blue")
plot(ys3, add=TRUE, col="green")
```

test.var.het

Variance heterogeneity test

Description

Tests for heterogeneity of sample variances in a paleoTS object.

Usage

```
test.var.het(y, method = "Bartlett")
pool.var(y, nn=NULL, ret.paleoTS=FALSE)
```

Arguments

y	a paleoTS object
method	test to be used; currently only Bartlett's test is implemented
nn	if y is a vector of variances, nn is a vector of sample sizes
ret.paleoTS	logical, indicating if the function should return a new paleoTS object with variances replaced by pooled variance.

Details

Tests for variance heterogeneity among samples. In the absence of evidence for heterogeneity, it may be desirable to replace individual estimates of phenotypic variance with a variance estimate pooled over all samples.

Function `pool.var` is used internally in `test.var.het`. It returns the value of the pooled variance.

Value

Function `pool.var` either returns a `paeIoTS` object, or the pooled variance.

Function `test.var.het` returns a list with the following elements relevant to Bartlett's test

<code>stat</code>	test statistic for Bartlett's test
<code>p.value</code>	P-value for statistical test
<code>df</code>	degrees of freedom, equal to one fewer than the number of samples

Author(s)

Gene Hunt

References

Hunt, G. 2006. Fitting and comparing models of phyletic evolution: random walks and beyond. *Paleobiology* **32**:578–601.

Sokal, R. and F. J. Rohlf 1995. *Biometry*, 3rd Ed.

Examples

```
# generate simulated data with no real variance heterogeneity
y <- sim.GRW(20, 0, 1)
bart <- test.var.het(y) # nonsignificant 95% of the time!
ys<- pool.var(y, ret.paleoTS=TRUE)
cat (ys$vv) # note all variances now the same
```

Index

- *Topic **datasets**
 - Stickleback data, 38
- *Topic **hplot**
 - plot.paleoTS, 31
- *Topic **models**
 - as.paleoTS, 3
 - as.paleoTSfit, 5
 - cat.paleoTS, 6
 - compareModels, 7
 - fit.sgs, 8
 - fit3models, 10
 - fitGpunc, 12
 - IC, 14
 - ln.paleoTS, 15
 - logL.GRW, 16
 - logL.joint.GRW, 17
 - LRI, 18
 - lynchD, 19
 - mle.GRW, 21
 - modelCurves, 22
 - opt.covTrack, 23
 - opt.GRW, 24
 - opt.GRW.shift, 26
 - opt.joint.GRW, 27
 - opt.RW.Mult, 29
 - plot.paleoTS, 31
 - sim.covTrack, 32
 - sim.GRW, 33
 - sim.OU, 34
 - sim.punc, 36
 - std.paleoTS, 37
 - sub.paleoTS, 39
 - test.var.het, 40
- *Topic **package**
 - paleoTS-package, 2
- *Topic **ts**
 - as.paleoTS, 3
 - as.paleoTSfit, 5
 - cat.paleoTS, 6
 - compareModels, 7
 - dorsal.spines (Stickleback data), 38
 - fit.sgs, 8, 13
 - fit3models, 8, 10, 19, 25, 30
 - fitGpunc, 7, 10, 12
 - IC, 14
 - akaik.wts (IC), 14
 - as.paleoTS, 3, 5, 31, 40
 - as.paleoTSfit, 5, 10, 15, 24, 25, 27, 29, 32
 - cat.paleoTS, 6
 - compareModels, 7, 15
 - compareModels, 7
 - fit.sgs, 8
 - fit3models, 10
 - fitGpunc, 12
 - IC, 14
 - ln.paleoTS, 15
 - logL.GRW, 16
 - logL.joint.GRW, 17
 - LRI, 18
 - lynchD, 19
 - mle.GRW, 21
 - modelCurves, 22
 - opt.covTrack, 23
 - opt.GRW, 24
 - opt.GRW.shift, 26
 - opt.joint.GRW, 27
 - opt.RW.Mult, 29
 - plot.paleoTS, 31
 - sim.covTrack, 32
 - sim.GRW, 33
 - sim.OU, 34
 - sim.punc, 36
 - std.paleoTS, 37
 - sub.paleoTS, 39
 - test.var.het, 40

- ln.paleoTS, 15, 38
- logL.covTrack (opt.covTrack), 23
- logL.GRW, 16, 18, 22, 25, 30
- logL.joint.GRW, 17, 29
- logL.joint.OU (logL.joint.GRW), 17
- logL.joint.punc (fitGpunc), 12
- logL.joint.Stasis (logL.joint.GRW), 17
- logL.joint.URW (logL.joint.GRW), 17
- logL.Mult (opt.RW.Mult), 29
- logL.Mult.covTrack (opt.covTrack), 23
- logL.punc (fitGpunc), 12
- logL.SameMs (opt.RW.Mult), 29
- logL.SameVs (opt.RW.Mult), 29
- logL.sgs (fit.sgs), 8
- logL.Stasis (logL.GRW), 16
- logL.URW (logL.GRW), 16
- LRI, 18
- lynchD, 19

- mle.GRW, 17, 21
- mle.Stasis (mle.GRW), 21
- mle.URW (mle.GRW), 21
- modelCurves, 22

- opt.covTrack, 23, 32
- opt.GRW, 10, 11, 13, 15, 17, 21, 22, 24, 24, 27, 29, 30, 34
- opt.GRW.shift, 7, 26, 37
- opt.joint.GRW, 10, 11, 18, 27
- opt.joint.OU, 35
- opt.joint.OU (opt.joint.GRW), 27
- opt.joint.punc (fitGpunc), 12
- opt.joint.Stasis (opt.joint.GRW), 27
- opt.joint.URW (opt.joint.GRW), 27
- opt.punc, 37
- opt.punc (fitGpunc), 12
- opt.RW.Mult, 25, 27, 29
- opt.RW.SameMs (opt.RW.Mult), 29
- opt.RW.SameVs (opt.RW.Mult), 29
- opt.sgs, 37
- opt.sgs (fit.sgs), 8
- opt.Stasis (opt.GRW), 24
- opt.URW, 20
- opt.URW (opt.GRW), 24
- ou.M (sim.OU), 34
- ou.V (sim.OU), 34

- paleoTS (paleoTS-package), 2
- paleoTS-package, 2

- pelvic.score (Stickleback data), 38
- plot.paleoTS, 4, 31
- pool.var (test.var.het), 40
- pterygiophores (Stickleback data), 38

- read.paleoTS (as.paleoTS), 3

- shift2gg (cat.paleoTS), 6
- shifts (cat.paleoTS), 6
- sim.covTrack, 32
- sim.GRW, 25, 33, 35
- sim.GRW.shift, 27
- sim.GRW.shift (sim.punc), 36
- sim.OU, 34
- sim.punc, 7, 13, 36
- sim.sgs, 10
- sim.sgs (sim.punc), 36
- sim.Stasis (sim.GRW), 33
- split4punc (cat.paleoTS), 6
- std.paleoTS, 16, 37
- Stickleback data, 38
- sub.paleoTS, 39

- test.var.het, 40