Package ‘phylosem’

October 6, 2023

Type Package
Title Phylogenetic Structural Equation Model
Version 1.1.0
Date 2023-10-03
Imports sem, ape, phylobase, phylopath
Depends TMB, R (>= 4.0.0),
Suggests semPlot, TreeTools, Rphylopars, phylolm, fishtree, phyr, knitr, rmarkdown, ggplot2, testthat
Enhances phytools, DiagrammeR
LinkingTo RcppEigen, TMB

Description
Applies phylogenetic comparative methods (PCM) and phylogenetic trait imputation using structural equation models (SEM), extending methods from Thorson et al. (2023) <doi:10.1111/2041-210X.14076>. This implementation includes a minimal set of features, to allow users to easily read all of the documentation and source code. PCM using SEM includes phylogenetic linear models and structural equation models as nested submodels, but also allows imputation of missing values. Features and comparison with other packages are described in Thorson and van der Bijl (In press).

License GPL-3
Encoding UTF-8
RoxygenNote 7.2.3
VignetteBuilder knitr
NeedsCompilation yes
Author James Thorson [aut, cre] (<https://orcid.org/0000-0001-7415-1010>), Wouter van der Bijl [ctb] (<https://orcid.org/0000-0002-7366-1868>)
Maintainer James Thorson <James.Thorson@noaa.gov>
Repository CRAN
Date/Publication 2023-10-06 20:30:02 UTC
**as_fitted_DAG**

**Convert output from package phylosem to phylopath**

**Description**

Convert phylosem to phylopath output

**Usage**

```r
as_fitted_DAG(object)
```

**Arguments**

- `object` Output from `phylosem`

**Value**

Convert output to format supplied by `est_DAG`
as_phylo4d

Convert output from package phylosem to phylo4d object

Description

Convert phylosem to phylo4d

Usage

as_phylo4d(object)

Arguments

object Output from phylosem

Value

Convert output to format supplied by phylo4d

as_sem

Convert output from package phylosem to sem

Description

Convert phylosem to sem output

Usage

as_sem(object)

Arguments

object Output from phylosem

Value

Convert output to format supplied by sem
average(x, cut_off, avg_method)

Arguments

x
output from compare_phylosem

Arguments

cut_off
threshold where any model with delta-AIC greater than this value is excluded from average

avg_method
see average_DAGs

Value

Returns an AIC-weighted average of fitted models from compare_phylosem after conversion to format from est_DAG
**best**

*Extract best fitted model*

**Description**

Extract best fitted model

**Usage**

```r
best(x)
```

**Arguments**

- `x`: output from `compare_phylosem`

**Value**

Returns best model from those fitted using `compare_phylosem`

---

**best.compare_phylosem**

*Extract best fitted model*

**Description**

Extract best fitted model

**Usage**

```r
## S3 method for class 'compare_phylosem'
best(x)
```

**Arguments**

- `x`: output from `compare_phylosem`

**Value**

Returns best model from those fitted using `compare_phylosem`
choice.compare_phylosem

------

| choice | Choose model |

- **Description**
  Choose model

- **Usage**
  
  ```
  choice(x, choice)
  ```

- **Arguments**
  
  - `x`: output from `compare_phylosem`
  - `choice`: Integer indicating model to extract

- **Value**
  
  Returns chosen model from those fitted using `compare_phylosem`
coef.phylosem

**Extract path coefficients**

**Description**

Extract path coefficients.

**Usage**

```r
## S3 method for class 'phylosem'
coef(object, standardized = FALSE, ...)
```

**Arguments**

- `object`: Output from `phylosem`
- `standardized`: Whether to standardize regression coefficients
- `...`: Not used

**Value**

Data-frame listing all path coefficients, their parameter index and estimated values

---

compare_phylosem

**Compare phylogenetic structural equation models**

**Description**

Fits several phylogenetic structural equation model for further comparison

**Usage**

```r
compare_phylosem(
  sem_set,
  tree,
  data,
  family = rep("fixed", ncol(data)),
  covs,
  estimate_ou = FALSE,
  estimate_lambda = FALSE,
  estimate_kappa = FALSE,
  ...
)
```
Arguments

sem_set A named list of structural equation model specifications, where each element will be passed as argument sem to phylosem

tree phylogenetic structure, using class as.phylo

data data-frame providing variables being modeled. Missing values are inputted as NA. If an SEM includes a latent variable (i.e., variable with no available measurements) then it still must be inputted as a column of data with entirely NA values.

family Character-vector listing the distribution used for each column of data, where each element must be fixed, normal, binomial, or poisson. family="fixed" is default behavior and assumes that a given variable is measured exactly. Other options correspond to different specifications of measurement error.

covs optional: a character vector of one or more elements, with each element giving a string of variable names, separated by commas. Variances and covariances among all variables in each such string are added to the model. For confirmatory factor analysis models specified via cfa, covs defaults to all of the factors in the model, thus specifying all variances and covariances among these factors. Warning: covs="x1, x2" and covs=c("x1", "x2") are not equivalent: covs="x1, x2" specifies the variance of x1, the variance of x2, and their covariance, while covs=c("x1", "x2") specifies the variance of x1 and the variance of x2 but not their covariance.

estimate_ou Boolean indicating whether to estimate an autoregressive (Ornstein-Uhlenbeck) process using additional parameter lnalpha, corresponding to the model="OUrandomRoot" parameterization from phylolm as listed in doi:10.1093/sysbio/syu005

estimate_lambda Boolean indicating whether to estimate additional branch lengths for phylogenetic tips (a.k.a. the Pagel-lambda term) using additional parameter logitlambda

estimate_kappa Boolean indicating whether to estimate a nonlinear scaling of branch lengths (a.k.a. the Pagel-kappa term) using additional parameter lnkappa

... Additional arguments passed to phylosem

Value

An object (list) of class 'compare_phylosem', containing a list of output from phylosem

fit_tmb

Optimize a TMB model

Description

fit_tmb runs a TMB model and generates standard diagnostics
Usage

fit_tmb(
  obj,
  fn = obj$fn,
  gr = obj$gr,
  startpar = NULL,
  lower = -Inf,
  upper = Inf,
  getsd = TRUE,
  control = list(eval.max = 10000, iter.max = 10000, trace = 1),
  bias.correct = FALSE,
  bias.correct.control = list(sd = FALSE, split = NULL, nsplit = NULL, vars_to_correct = NULL),
  savedir = NULL,
  loopnum = 2,
  newtonsteps = 0,
  n = Inf,
  getReportCovariance = FALSE,
  getJointPrecision = FALSE,
  getHessian = FALSE,
  quiet = FALSE,
  start_time_elapsed = as.difftime("0:0:0"),
  ...
)

Arguments

obj The compiled TMB object
fn Function to be minimized. Must return a scalar value.
gr Optional function that takes the same arguments as objective and evaluates the gradient of objective at its first argument. Must return a vector as long as startpar
startpar Starting values for fixed effects (default NULL uses obj$par)
lower, upper vectors of lower and upper bounds, replicated to be as long as start. If unspecified, all parameters are assumed to be unconstrained.
getsd Boolean indicating whether to run standard error calculation; see sdreport for details
control A list of control parameters. For details see nlminb
bias.correct Boolean indicating whether to do epsilon bias-correction; see sdreport and fit_tmb for details
bias.correct.control tagged list of options for epsilon bias-correction, where vars_to_correct is a character-vector of ADREPORT variables that should be bias-corrected
savedir directory to save results (if savedir=NULL, then results aren’t saved)
loopnum number of times to re-start optimization (where loopnum=3 sometimes achieves a lower final gradient than loopnum=1)
newtonsteps  Integer specifying the number of extra newton steps to take after optimization (alternative to loopnum). Each newtonstep requires calculating the Hessian matrix and is therefore slow. But for well-behaved models, each Newton step will typically decrease the maximum gradient of the loglikelihood with respect to each fixed effect, and therefore this option can be used to achieve an arbitrarily low final gradient given sufficient time for well-behaved models. However, this option will also perform strangely or have unexpected consequences for poorly-behaved models, e.g., when fixed effects are at upper or lower bounds. There is no need to increase the value of newtonsteps if the model appears to be converged (i.e., has low final_gradient and none of the standard errors are NA values)

n  sample sizes (if n!=Inf then n is used to calculate BIC and AICc)

getReportCovariance  Get full covariance matrix of ADREPORTed variables?

getJointPrecision  Optional. Return full joint precision matrix of random effects and parameters?

getHessian  return Hessian for usage in later code

quiet  Boolean whether to print additional messages results to terminal

start_time_elapsed  how much time has elapsed prior to calling fit_tmb, for use, e.g., when calling fit_tmb multiple times in sequence, where start_time_elapsed = opt_previous$time_for_run

...  list of settings to pass to sdreport

Value

the standard output from nlminb, except with additional diagnostics and timing info, and a new slot containing the output from sdreport

References

For more details see doi:10.1016/j.fishres.2015.11.016

Description

list_parameters lists all fixed and random effects

Usage

list_parameters(Obj, verbose = TRUE)

Arguments

Obj  Compiled TMB object

verbose  Boolean, whether to print messages to terminal
Value

Tagged-list of fixed and random effects, returned invisibly and printed to screen

Fisheries natural mortality example

Description


Usage

data(Mlifehist_ver1_0)

Parse path

Description

parse_path is copied from sem::parse.path

Usage

parse_path(path)

Arguments

path text to parse

Details

Copied with permission from John Fox under licence GPL (>= 2)

Value

Tagged-list defining variables and direction for a specified path coefficient
phylosem

Fit phylogenetic structural equation model

Description

Fits a phylogenetic structural equation model

Usage

phylosem(
  sem,
  tree,
  data,
  family = rep("fixed", ncol(data)),
  covs = colnames(data),
  estimate_ou = FALSE,
  estimate_lambda = FALSE,
  estimate_kappa = FALSE,
  data_labels = rownames(data),
  quiet = FALSE,
  newtonsteps = 1,
  tmb_inputs = NULL,
  run_model = TRUE,
  ...
)

Arguments

- **sem**: structural equation model structure, passed to either `specifyModel` or `specifyEquations` and then parsed to control the set of path coefficients and variance-covariance parameters
- **tree**: phylogenetic structure, using class `as.phylo`
- **data**: data-frame providing variables being modeled. Missing values are inputted as NA. If an SEM includes a latent variable (i.e., variable with no available measurements) then it still must be inputted as a column of data with entirely NA values.
- **family**: Character-vector listing the distribution used for each column of data, where each element must be `fixed`, `normal`, `binomial`, or `poisson`. `family="fixed"` is default behavior and assumes that a given variable is measured exactly. Other options correspond to different specifications of measurement error.
- **covs**: optional: a character vector of one or more elements, with each element giving a string of variable names, separated by commas. Variances and covariances among all variables in each such string are added to the model. For confirmatory factor analysis models specified via `cfa`, `covs` defaults to all of the factors in the model, thus specifying all variances and covariances among these factors. **Warning**: `covs=\"x1, x2\"` and `covs=c(\"x1\", \"x2\")` are not equivalent: `covs=\"x1,`
`x2" specifies the variance of `x1`, the variance of `x2`, and their covariance, while `covs=c("x1", "x2")` specifies the variance of `x1` and the variance of `x2` but not their covariance.

- **estimate_ou** Boolean indicating whether to estimate an autoregressive (Ornstein-Uhlenbeck) process using additional parameter `lnalpha`, corresponding to the `model="OUrandomRoot"` parameterization from `phylolm` as listed in doi:10.1093/sysbio/syu005
- **estimate_lambda** Boolean indicating whether to estimate additional branch lengths for phylogenetic tips (a.k.a. the Pagel-lambda term) using additional parameter `lnkappa`
- **estimate_kappa** Boolean indicating whether to estimate a nonlinear scaling of branch lengths (a.k.a. the Pagel-kappa term) using additional parameter `lnkappa`
- **data_labels** For each row of data, listing the corresponding name from `tree$tip.label`. Default pulls `data_labels` from `rownames(data)`
- **quiet** if `FALSE`, the default, then the number of input lines is reported and a message is printed suggesting that `specifyEquations` or `cfa` be used.
- **newtonsteps** Integer specifying the number of extra newton steps to take after optimization (alternative to `loopnum`). Each newtonstep requires calculating the Hessian matrix and is therefore slow. But for well-behaved models, each Newton step will typically decrease the maximum gradient of the loglikelihood with respect to each fixed effect, and therefore this option can be used to achieve an arbitrarily low final gradient given sufficient time for well-behaved models. However, this option will also perform strangely or have unexpected consequences for poorly-behaved models, e.g., when fixed effects are at upper or lower bounds. There is no need to increase the value of `newtonsteps` if the model appears to be converged (i.e., has low `final_gradient` and none of the standard errors are NA values)
- **tmb_inputs** optional tagged list that overrides the default constructor for TMB inputs (use at your own risk)
- **run_model** Boolean indicating whether to estimate parameters (the default), or instead to return the model inputs and compiled TMB object without running;

... Additional parameters passed to `fit_tmb`

**Details**

Note that parameters `logitlambda`, `lnkappa`, and `lnalpha` if estimated are each estimated as having a single value that applies to all modeled variables. This differs from default behavior in `phylolm`, where these parameters only apply to the "response" and not "predictor" variables. This also differs from default behavior in `phylopath`, where a different value is estimated in each call to `phylolm` during the d-separation estimate of path coefficients. However, it is consistent with default behavior in `Rphylopars`, and estimates should be comparable in that case. These additional parameters are estimated with unbounded support, which differs somewhat from default bounded estimates in `phylolm`, although parameters should match if overriding `phylolm` defaults to use unbounded support. Finally, `phylosem` allows these three parameters to be estimated in any combination, which is expanded functionality relative to the single-option functionality in `phylolm`.

Also note that `phylopath` by default uses standardized coefficients. To achieve matching parameter estimates between `phylosem` and `phylopath`, standardize each variable to have a standard deviation of 1.0 prior to fitting with `phylosem`. 
Value

An object (list) of class 'phylosem'. Elements include:

- **data** Copy of argument data
- **SEM_model** SEM model parsed from sem using `specifyModel` or `specifyEquations`
- **obj** TMB object from `MakeADFun`
- **tree** Copy of argument tree
- **tmb_inputs** The list of inputs passed to `MakeADFun`
- **opt** The output from `fit_tmb`
- **report** The output from `obj$report()`
- **parhat** The output from `obj$env$parList()` containing maximum likelihood estimates and empirical Bayes predictions

References

**Introducing the package, its features, and comparison with other software (to cite when using phylosem):**

*Statistical methods for phylogenetic structural equation models*

*Earlier development of computational methods, originally used for phlogenetic factor analysis:*


*Earlier development of phylogenetic path analysis:*

*Interface involving SEM ‘arrow notation’ is repurposed from:*

*Coercing output to phylo4d depends upon:*

*Laplace approximation for parameter estimation depends upon:*

Examples

```r
# Load data set
data(rhino, rhino_tree, package="phylopath")

# Run phylosem
model = "
  DD -> RS, p1
  BM -> LS, p2
  BM -> NL, p3
  NL -> DD, p4"
psem = phylosem( sem = model, 
                   data = rhino[,c("BM","NL","DD","RS","LS")],
                   tree = rhino_tree )

# Convert and plot using phylopath
library(phylopath)
my_fitted_DAG = as_fitted_DAG(psem)
coef_plot( my_fitted_DAG )
plot( my_fitted_DAG )

# Convert to phylo4d
my_phylo4d = as_phylo4d(psem)

# Convert to sem and plot
library(sem)
my_sem = as_sem(psem)
pathDiagram( model = my_sem,
             style = "traditional",
             edge.labels = "values" )
effects( my_sem )

# Plot using semPlot
if( require(semPlot) ){
  myplot = semPlotModel( my_sem )
  semPaths( my_sem,
            nodeLabels = myplot@Vars$name )
}
```
**print.phylosem**

*Print parameter estimates*

**Description**

Print parameter estimates and standard errors.

**Usage**

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

**Arguments**

- `x`  
  Output from *phylosem*

- `...`  
  Not used

**Value**

prints (and invisibly returns) output from `fit_tmb`

---

**summary.phylosem**

*Summarize phylosem*

**Description**

summarize phylosem

**Usage**

```r
## S3 method for class 'phylosem'
summary(object, ...)
```

**Arguments**

- `object`  
  Output from *phylosem*

- `...`  
  Not used

**Value**

Data-frame containing all estimated intercepts, path coefficients, and variance-covariance parameters as well as their standard errors
**TMBAIC**

*Calculate marginal AIC for a fitted model*

**Description**

TMBAIC calculates AIC for a given model fit

**Usage**

TMBAIC(opt, k = 2, n = Inf)

**Arguments**

- `opt` the output from `nlminb` or `optim`
- `k` the penalty on additional fixed effects (default=2, for AIC)
- `n` the sample size, for use in AICc calculation (default=Inf, for which AICc=AIC)

**Value**

AIC, where a parsimonious model has a AIC relative to other candidate models

---

**vcov.phylosem**

*Extract Variance-Covariance Matrix*

**Description**

extract the covariance of fixed effects, or both fixed and random effects.

**Usage**

```r
## S3 method for class 'phylosem'
vcov(object, which = c("fixed", "random", "both"), ...)
```

**Arguments**

- `object` output from `phylosem`
- `which` whether to extract the covariance among fixed effects, random effects, or both
- `...` ignored, for method compatibility
Index

* data
  Mlifehist_ver1_0, 11
  as.phylo, 8, 12
  as_fitted_DAG, 2
  as_phylo4d, 3
  as_sem, 3
  average, 4
  average.compare_phylosem, 4
  average_DAGs, 4
  best, 5
  best.compare_phylosem, 5
  choice, 6
  choice.compare_phylosem, 6
  coef.phylosem, 7
  compare_phylosem, 4–6, 7
  est_DAG, 2, 4
  fit_tmb, 8, 9, 13, 14, 16
  list_parameters, 10
  MakeADFun, 14
  Mlifehist_ver1_0, 11
  nlminb, 9, 10
  parse_path, 11
  phylo4d, 3
  phylosem, 2, 3, 7, 8, 12, 16
  print.phylosem, 16
  sdreport, 9, 10
  sem, 3
  specifyEquations, 12, 14
  specifyModel, 12, 14
  summary.phylosem, 16
  TMBAIC, 17
  vcov.phylosem, 17