Package 'tapnet'

May 8, 2026

Title Trait Matching and Abundance for Predicting Bipartite Networks

Version 0.6

Description
Functions to produce, fit and predict from bipartite networks with abundance, trait and phylogenetic information. Its methods are described in detail in Benadi, G., Dormann, C.F., Fruend, J., Stephan, R. & Vazquez, D.P. (2021) Quantitative prediction of interactions in bipartite networks based on traits, abundances, and phylogeny. The American Naturalist, in press.

Depends R (>= 3.5.0)

Imports ape, bipartite, MPSEM, methods, phytools, stats, utils, vegan

License GPL

Encoding UTF-8

LazyData true

ByteCompile true

URL https://github.com/biometry/tapnet

Suggests knitr

VignetteBuilder knitr, utils

RoxygenNote 7.3.3

NeedsCompilation no

Author Carsten Dormann [aut, cre],
Gita Benadi [aut],
Boris Tinoco [dtc],
Ruth Stephan [ctb],
Jochen Fruend [ctb]

Maintainer Carsten Dormann <carsten.dormann@biom.uni-freiburg.de>

Repository CRAN

Date/Publication 2026-01-20 14:00:02 UTC

Contents

`fit_tapnet` *Fit the tapnet model to a network*

Description

Estimates the parameters of the tapnet model by log-likelihood based on the observed network(s)

Usage

```
fit_tapnet(
  tapnet,
  ini = NULL,
  tmatch_type_pem = "normal",
  tmatch_type_obs = "normal",
  TmatchMatrixList = NULL,
  lambda = 0,
  method = "Nelder",
  maxit = 50000,
  hessian = FALSE,
  obj_function = "multinom",
  fit.delta = FALSE
)
```

Arguments

`tapnet`	a tapnet object;
`ini`	initial parameter values for the optimization; optional;
`tmatch_type_pem`	type of trait matching function for latent traits, currently "normal" or "shiftl-norm";
`tmatch_type_obs`	type of trait matching function for observed traits, currently "normal" or "shiftl-norm";

TmatchMatrixList	list of independent trait-matching matrices (one per network);
lambda	LASSO shrinkage factor for latent trait parameters;
method	Optimization method (most derivative-based approaches will not work! SANN is a (slow) alternative to the default);
maxit	Maximum number of steps for optimization;
hessian	logical: output hessian for calculation of standard errors?
obj_function	Objective function for the optimization, either "multinom" or "sq_diff" (or "bjorn");
fit.delta	logical; should the parameter delta be fitted? It allows tapnet to down-weigh the importance of trait matching relative to abundances. Defaults to FALSE.

Details

The core function for using the tapnet approach: it fits the model to the data (= networks). Then, the estimated parameters can be used to predict to other networks (using predict_tapnet).

Value

A tapnet-fit object, containing the tapnet model parameters as entries "par_opt", the settings of the tmatch_type for PEMs and observed traits, the parameter set for lambda, the optimisation method set, along with its maxit-value, and, finally, the output of the call to optim, including the target value (the negative log-likelihood), the convergence report and the parameters as fitted *at the transformed scale*. Note that the entries under "opt" will not be the same as those under "par_opt"!

Author(s)

Gita Benadi <gita.benadi@biom.uni-freiburg.de> and Carsten Dormann <carsten.dormann@biom.uni-freiburg.de>

References

Benadi et al. in prep

Examples

```
# takes about 35 s
data(Tinoco)
tap <- make_tapnet(tree_low = plant_tree, tree_high = humm_tree, networks = networks[2:3],
       traits_low = plant_traits, traits_high = humm_traits, npems_lat = 4)
fit <- fit_tapnet(tap) # fits to networks 2 and 3 only
str(fit)
```

gof_tapnet *Goodness-of-fit of a tapnet fit*

Description

Provides various measures to describe how well the tapnet model fits the data

Usage

```
gof_tapnet(
  fit,
  tapnet = NULL,
  indices = c("connectance", "NODF", "weighted NODF", "H2"),
  nrep = 1000,
  se_refit = FALSE,
  se_nsim = 1000
)
```

Arguments

fit	results of applying fit_tapnet to the tapnet object;
tapnet	a tapnet object created with simulate_tapnet or make_tapnet; if NULL, the name stored in the attributes of fit is used to access an object of that name in the global environment; can be used e.g. in simulations, when the tapnet object is renamed relative to the one fitted;
indices	network indices to compare between observed and fitted network; calls networklevel;
nrep	Number of networks to simulate for indices comparison; these are draws from the fitted multinomial distribution;
se_refit	logical; should standard errors for the parameters be calculated using parametric bootstrap (refitting on simulated data)? Defaults to FALSE because it's very slow (i.e. takes hours).
se_nsim	number of simulations for parametric bootstrap (ignored unless the previous argument is set to TRUE).

Details

This is a function particularly interesting for simulated data, when the true parameters are known. In this case, GOF for fitted latent traits and so forth are computed.

Value

A list of goodness-of-fit measures: bc_sim_web are the Bray-Curtis similarities between fitted and observed network; cor_web are Spearman correlations between fitted and observed; and net_indices compute the selected network indices for fitted and observed networks. Also outputs the fitted I_mat for users to do other gymnastics with it. If more than one network is used for fitting, all these measures are returned for all networks (as vector or list under the respective label). See example.

Author(s)

Gita Benadi <gita.benadi@biom.uni-freiburg.de> and Carsten Dormann <carsten.dormann@biom.uni-freiburg.de>

References

Benadi et al. in prep

Examples

```
data(Tinoco)
tap <- make_tapnet(tree_low = plant_tree, tree_high = humm_tree, networks = networks[2:3],
       traits_low = plant_traits, traits_high = humm_traits, npems_lat = 4)
fit <- fit_tapnet(tap) # uses networks 2 and 3 for fitting!
gof_tapnet(fit)
```

`internalFunctions` *Helper functions for tapnet*

Description

Lower-level, non-exported functions to be called by the main tapnet functions

Usage

```
internalFunctions()

bjornloglik(web, P)

fit_abund(tapnet)

pems_from_tree(tree)

select_relevant_pems(tree, species)

tmatch(delta_t, type = "normal", width = 1, shift = 0, xi = 1, err = 1e-05)

param_vec2list(params, n, m, fit.delta = FALSE)

loglik_tapnet(
  params,
  networks,
  tmatch_type_pem,
  tmatch_type_obs,
  TmatchMatrixList = NULL,
```

```
  lambda = 0,
  obj_function = "multinom",
  fit.delta = TRUE
)

latent_cor(true_pars, fitted_pars, pems_low, pems_high)

web_indices(web, web_dim, indices)

refit_params(tapnet, fit, fitted_I_mat)
```

Arguments

web	data for an interaction network in vector form, e.g. from predict_tapnet;
P	matrix of same size as observed web, summing to 1, representing something like the probability of selecting a link; typically constructed as part of tapnet, i.e. the I-mat of fit_tapnet;
tapnet	a tapnet object;
tree	phylogenetic tree in phylo format;
species	a named vector of species, representing (some of) the tips of the phylogenetic tree;
delta_t	vector of pairwise trait differences (higher - lower);
type	trait matching function: either "normal" or "shiftlnorm";
width	width parameter of trait matching function, similar to sd in the normal;
shift	shift parameter (optimum trait distance), currently ignored in fitting;
xi	penalty for increasing the width of the (unstandardised) trait-matching function; defaults to 1 (implying a quadratically increasing weight of width in the optimisatino function, i.e. strongly acting against large values of sigma)
err	"baseline" probability of match, even if traits do not match at all;
params	parameter vector with setting for tapnet simulation;
n	number of latent trait linear combination parameters (lower level);
m	number of latent trait linear combination parameters (higher level);
fit.delta	logical; should the trait-weighting exponent delta be fitted?
networks	the "networks" part of a tapnet object;
tmatch_type_pem	type of trait matching function for latent traits;
tmatch_type_obs	type(s) of trait matching functions for observed traits; can be a vector as long as there are traits;
TmatchMatrixList	list of independent trait-matching matrices (one per network);
lambda	LASSO shrinkage parameter to avoid collinearity issues when observed traits are phylogenetically correlated;

obj_function	objective function, either "multinom" or "least squares" (leads to OLS fitting) or "bjorn" (leading to use of a somewhat weird but in some opinion the correct way to compute the likelihood);
true_pars	parameters used for simulating the network;
fitted_pars	parameters estimated by fit_tapnet;
pems_low	phylogenetic eigenvectors for the lower trophic level;
pems_high	phylogenetic eigenvectors for the higher trophic level;
web_dim	vector of two numbers indicating the rows and column of the network matrix;
indices	vector of names of network indices to compute; see networklevel for what is available;
fit	a fitted tapnet;
fitted_I_mat	the fitted I-matrix of a fitted tapnet object.

Details

They do roughly the following:

bjornloglik computes likelihood of obtaining the observed interaction matrix, given some expected matrix of interaction propabilities (P). In contrast to the multinomial, this function assumes that the marginal totals of P constrain the probabilities. Hence, if all observations for one column (or row) have been evaluated for their probabilities, any new observation cannot come from this column (or row) anymore. This means, the probabilities in that "depleted" column (or row) have to be proportionally distributed over the other rows (or columns). There are ongoing discussions among the authors, when this is the right approach. Function written by Björn Reineking, ISRAE Grenoble (many thanks!), hence the name.

fit_abund computes expected P-matrix of observed interactions based only on abundances; if no external abundances are provided in the tapnet-object, it will use the marginal totals of the network(s) instead.

pems_from_tree computes phylogenetic eigenvectors from a phylogenetic tree;

select_relevant_pems identifies those phylogenetic eigenvectors (PEMs) of the full tree most relevant for a network containing only a subset of species;

tmatch calculates interaction probabilities based on trait matching;

param_vec2list converts a vector of parameters (for trait matching and latent trait combinations) into a named list;

loglik_tapnet the (negative!) log-likelihood function for fitting the tapnet model; actually quite an important function, easy to break, so not for the user to easily access;

latent_cor computes correlation of fitted latent with true constructed traits for simulated data;

web_indices computes the specified network indices for the provided network, after turning the prediction vector into a matrix;

refit_params Simulate new networks from a fitted tapnet object, re-fit on the simulated network and output the parameter values.

Author(s)

Gita Benadi <gita.benadi@biom.uni-freiburg.de>, Carsten Dormann <carsten.dormann@biom.uni-freiburg.de> and Jochen Fründ <jochen.fruend@biom.uni-freiburg.de>

References

Benadi et al. in prep

`make_tapnet` *Constructs an object of type "tapnet"*

Description

Collates networks, traits and phylogenies into a consistent data structure used for all other tapnet functions

Usage

```
make_tapnet(
  tree_low,
  tree_high,
  networks,
  abun_low = NULL,
  abun_high = NULL,
  traits_low = NULL,
  traits_high = NULL,
  npems_lat = NULL,
  use.all.pems = FALSE,
  empty = TRUE
)
```

Arguments

`tree_low`	phylogenetic tree of lower trophic level; required;
`tree_high`	phylogenetic tree of higher trophic level; required;
`networks`	a single or list of interaction network (as matrix); required;
`abun_low`	named abundance vector(s) for lower trophic level (single vector or list of vectors); optional;
`abun_high`	named abundance vector(s) for higher trophic level; optional;
`traits_low`	lower trophic level traits (species x traits matrix with row and column names); optional;
`traits_high`	higher trophic level traits (species x traits matrix with row and column names; optional;
`npems_lat`	number of phylogenetic eigenvectors to be used to construct latent traits. If NULL, all eigenvectors will be used.
`use.all.pems`	option to force the function to use all phylogenetic eigenvectors, not only those useful for describing the specific network's species.
`empty`	logical; should networks be emptied of all-0 rows or columns? Defaults to TRUE.

Details

Tapnet objects are the starting point for almost all other tapnet functions. They contain the information on the species and the (quantitative) interaction network data.

Value

A tapnet object, i.e. an thoroughly organised list with the inputs as entries. If multiple networks are provided, each has its own list entry, with PEMs, traits and abundances given for each network separately, in addition to the overall phylogenetic eigenvectors across all networks. See example for, well, for an example.

Author(s)

Gita Benadi <gita.benadi@biom.uni-freiburg.de>

References

Benadi et al. in prep

Examples

```
data(Tinoco)
tapnet_web1 <- make_tapnet(tree_low = plant_tree, tree_high = humm_tree, networks = networks[1],
               traits_low = plant_traits, traits_high = humm_traits, npems_lat = 4)
str(tapnet_web1) # show tapnet structure
```

---

predict_tapnet *Predict from fitted tapnet to new data*

Description

Function allows direct use of data prepared for tapnet analysis by other statistical methods, e.g. regression approaches

Usage

```
predict_tapnet(fit, abuns, tapnet = NULL)
```

Arguments

fit	results of applying fit_tapnet to the tapnet object;
abuns	named list of two entries ("low" and "high"), containing a species-named vector of abundances of the new network
tapnet	optional name of a tapnet object containing traits, phylogeny etc.; these are not stored in 'fit', but rather it is assumed that a tapnet object with the name stored in 'fit' is available in the global environment. That may not be the case, e.g. when simulating networks. In this case, 'tapnet' provides the required tapnet-object.

Details

The fitted tapnet object contains the estimated parameters, describing how traits, abundance and phylogeny play together to produce the network(s) used for fitting. This information is now used to predict interaction probabilities for a new network. Accordingly, we need to know this new network's species abundances (an input to the function), PEMs and traits. The latter are computed based on the information contained in the tapnet object (which is linked in by attribute reference). For new species, their PEMs are computed and the network is simulated, using `simnetfromtap`, for the information provided.

Value

A matrix of predicted interaction probabilities, summing to 1. This would need to be multiplied by the total number of interactions in the new network to be comparable to the observations.

Author(s)

Ruth Stephan, Gita Benadi and Carsten Dormann <carsten.dormann@biom.uni-freiburg.de>

References

Benadi et al. in prep

Examples

```
data(Tinoco)
tap <- make_tapnet(tree_low = plant_tree, tree_high = humm_tree, networks = networks[1:2],
       traits_low = plant_traits, traits_high = humm_traits, npems_lat = 4)
fit <- fit_tapnet(tap) # uses two networks for fitting!
gof_tapnet(fit)
# predict to omitted forest network's abundances:
pred1 <- predict_tapnet(fit, abuns=list("low"=plant_abun[[3]], "high"=humm_abun[[3]] ))
cor(as.vector(pred1*sum(networks[[3]])), as.vector(networks[[3]]))
```

`simnetfromtap` *Simulates a network from parameters, abundances, traits and phylogeny provided*

Description

The workhorse function of this package, called by various other functions to construct a bipartite interaction network

Usage

```
simnetfromtap(
  traits,
  abuns,
  paramsList,
  pems,
  tmatch_type_pem,
  tmatch_type_obs
)
```

Arguments

traits	a named ("low"/"high") list of species-named trait data matrices for lower and higher trophic level;
abuns	a named ("low"/"high") list of species-named abundance vectors for lower and higher trophic level;
paramsList	a list of parameter values with six elements: [[1]] and [[2]]: two vectors of linear combination parameters (importance values, one vector for each trophic level); [[3]]: a single shift parameter added to linear combination of higher trophic level PEMs; [[4]]: a single trait matching parameter for the PEMs; [[5]]: a vector of trait matching parameters for observed traits; [[6]]: a non-negative scalar delta to weight the importance of abundances
pems	a named ("low"/"high") list of two species-named data frames (PEMs of lower and higher trophic level);
tmatch_type_pem	type of trait matching function for latent traits (any name accepted by tmatch, currently "normal", "shiftlnorm" and "no"); "no" means no PEMs are matched;
tmatch_type_obs	type of trait matching function for observed traits (see previous argument).

Details

Details in here!

Value

A named interaction matrix.

Author(s)

Gita Benadi <gita.benadi@biom.uni-freiburg.de> and Carsten Dormann <carsten.dormann@biom.uni-freiburg.de>

References

Benadi et al. in prep

`simulate_tapnet` *Creates a simulated along with all parameters, abundances, traits and phylogeny used*

Description

Simulation function to produce a tapnet object, i.e. one or more networks along with their descriptors, abundances, traits, phylogenetic information

Usage

```
simulate_tapnet(
  nlower,
  nhigher,
  ntraits_nopem,
  ntraits_pem,
  pem_noise = 0.5,
  abuns = "lognormal",
  meanlog = 0,
  sdlog = 1,
  tmatch_type_pem = "normal",
  tmatch_type_obs = "normal",
  npems_lat = NULL,
  lat_low = 1,
  lat_high = 1,
  tmatch_width_pem = 1,
  pem_shift = 0,
  tmatch_width_obs = 1,
  Nwebs = 1,
  prop_species = 1,
  new_abuns = FALSE,
  Nobs = 1000
)
```

Arguments

`nlower`	species number in lower trophic level;
`nhigher`	species number in higher trophic level;
`ntraits_nopem`	number of phylogenetically uncorrelated traits (for each level);
`ntraits_pem`	number of phylogenetically correlated traits (for each level);
`pem_noise`	noise (sd of normal dist) to put on PEMs;
`abuns`	abundances set to "lognormal", "equal" or a list of two abundance vectors;
`meanlog`	parameters of the log-normal distribution for drawing abundances;
`sdlog`	same as before, but width;

`tmatch_type_pem`
: type of trait matching function for latent traits;

`tmatch_type_obs`
: type of trait matching function for observed traits (can be a vector as long as there are traits);

`npems_lat`
: number of phylogenetic eigenvectors to be used to construct latent traits. If NULL, all eigenvectors will be used;

`lat_low`
: vector of PEM linear combination parameters for lower trophic level; if 1, all values will be set to 1, if "random", values will be drawn from a uniform dist;

`lat_high`
: same for higher trophic level;

`tmatch_width_pem`
: width of trait matching function for latent (PEM-based) traits;

`pem_shift`
: shift parameter for latent trait matching;

`tmatch_width_obs`
: width of trait matching function for observed traits, can be a single value or a vector of length ntraits_nopem + ntraits_pem;

`Nwebs`
: number of webs to be simulated;

`prop_species`
: proportion of species in the phylogeny that appear in each web (species are drawn randomly). With multiple networks, this allows to create networks with partly overlapping species composition.

`new_abuns`
: If abuns = "lognormal" and Nwebs > 1, should new abundances be drawn for each web?

`Nobs`
: number of observed interactions per web

Details

This function was written to explore the fitting of networks by different methods. Hence, we first have to simulate such networks. It is a very nice starting point for simulations, but irrelevant for tapnet-based analyses. The function internally sets up all the parameters and then calls `simnetfromtap` for the simulation of the actual network. Lots of options means, regrettably, lots of decisions for the user.

Value

A tapnet object, with a highly nested structure. There are six entries at the top level: trees, traits_all, networks, sim_params and the two tmatch types. Within networks, there is a list of 5 entries (for each of the 'Nweb' networks): abundances, traits, PEMs, web and I_mat. "I_mat" is the actual output from simnetfromtap, while "web" is a single draw from a multinomial distribution with I_mat as probabilities and 'Nobs' as size.

Author(s)

Gita Benadi <gita.benadi@biom.uni-freiburg.de>, Jochen Fründ <jochen.fruend@biom.uni-freiburg.de> and Carsten Dormann <carsten.dormann@biom.uni-freiburg.de>

References

Benadi et al. in prep

Examples

```
tapnet <- simulate_tapnet(nlower=10, nhigher=20, ntraits_nopem=2, ntraits_pem=0)
# a minimal call of simulate_tapnet
str(tapnet, 1) # the structure at the first level
str(tapnet, 2) # the structure at the first and second level
```

---

tapnet2df *Convert tapnet object into data.frame*

Description

Function allows direct use of data prepared for tapnet analysis by other statistical methods, e.g. regression approaches

Usage

```
tapnet2df(tapnetObject)
```

Arguments

tapnetObject	results of applying fit_tapnet to the tapnet object;

Details

This function simply puts all data into a data.frame, with each row an entry in the network matrix.

Value

A data.frame containing network observations, PEMs, traits and abundances for regression-type analysis.

Author(s)

Carsten Dormann <carsten.dormann@biom.uni-freiburg.de>

References

Benadi et al. in prep

Examples

```
ex <- simulate_tapnet(nlower=10, nhigher=50, ntraits_pem=3, ntraits_nopem=2, Nwebs = 3)
df <- tapnet2df(ex)
head(df)
## Not run:
  library(ranger)
  frf <- ranger(interactions ~ ., data=df[, -c(1:2)], importance="impurity")
  sort(importance(frf), decreasing=TRUE)

## End(Not run)
```

Tinoco *Hummingbird-flower networks*

Description

An example dataset for tapnet analysis

Usage

```
data(Tinoco)
```

Format

An object of class `matrix` (inherits from `array`) with 14 rows and 1 columns.

Details

These data are from the supplement of Tinoco et al. (2017) and contain three observed networks (forest at Mazan, shrubland at Llaviuco, cattle farm at Nero, all in Ecuador), along with traits of flowers and birds (corolla and beak length, respectively) as well as phylogenies and external abundances for all species. These data are in several ways special, but most of all because of the very high sampling effort that went into the networks.

For sake of clarity, we provide the data as separate objects. So when "Tinoco" is called, it will load seven objects: networks, humm_traits, humm_tree, humm_abun, plant_traits, plant_tree and plant_abun. To combine them into a useable tapnet object, use `make_tapnet`. Phylogenetic trees are of class "phylo" (as used/produced by **phytools**). Abundance data were provided independently of the other data directly by Boris (the other data are on dryad doi:10.5061/dryad.j860v). For the external abundances of hummingbirds, 12 point counts were performed in the same habitats where hummingbird - plant interactions were observed. "Abundance were obtained by averaging the abundance of each species per point count across the study period." "Plant abundances are averages across the study period." Many thanks to Boris for making his data freely available!

Author(s)

Boris A. Tinoco Molina <btinoco@uazuay.edu.ec> collected the data; Carsten F. Dormann <carsten.dormann@biom.uni-
packaged them

References

Tinoco, B. A.; Graham, C. H.; Aguilar, J. M. & Schleuning, M. Effects of hummingbird morphology on specialization in pollination networks vary with resource availability. *Oikos* **126**, 52-–60

Examples

```
ls()
data(Tinoco)
ls() # adds seven objects!
```

Index