

# Package ‘RSnetwork’

October 8, 2024

**Title** Ecological Networks Analysis with Reciprocal Scaling

**Version** 1.0.0

**Description** Analyze ecological networks and possibly species-environment tables using various correspondence analyses methods, and also quantify niches widths with reciprocal scaling.

**License** MIT + file LICENSE

**Encoding** UTF-8

**Roxygen** list(markdown = TRUE)

**RoxygenNote** 7.3.2

**Imports** ade4,  
adegraphics,  
dplyr,  
ggplot2,  
ggrepel,  
grid,  
lmtest,  
mvtnorm,  
stats,  
stringr,  
tibble,  
tidyr,  
tidyselect

## Contents

add_params . . . . .	2
combine_cor . . . . .	2
compas2d . . . . .	3
create_code . . . . .	5
df_to_matrix . . . . .	5
filter_matrix . . . . .	6
get_best_model . . . . .	6
get_cor . . . . .	7
get_mean_sd . . . . .	7
get_niches . . . . .	8
get_pred . . . . .	9
lm_labels . . . . .	10
matrix_to_df . . . . .	11

multiplot . . . . .	11
plot_corcircle . . . . .	13
plot_eig . . . . .	14
plot_lm_mean_sd . . . . .	14
plot_matrix . . . . .	15
plot_reciprocal . . . . .	16

<b>Index</b>	<b>18</b>
--------------	-----------

---

add_params	<i>Add parameters</i>
------------	-----------------------

---

### Description

Adds parameters value to simulation results dataframe.

### Usage

```
add_params(corsim, paramvec)
```

### Arguments

corsim	a list with the correlation values (output of <a href="#">get_cor</a> ).
paramvec	A vector of parameters (must be a 1-row dataframe)

### Value

The list corsim with added columns in each dataframe of the list. The added columns are named like the columns of paramvec and contain repeated values of paramvec

---

combine_cor	<i>Combine correlations</i>
-------------	-----------------------------

---

### Description

Combine lists formatted by [get\\_cor](#).

### Usage

```
combine_cor(a, b)
```

### Arguments

a	First list formatted by <a href="#">get_cor</a>
b	Second list formatted by <a href="#">get_cor</a>

### Value

A list as formatted by [get\\_cor](#) (see this function's "Return" section).

compas2d

*Simulate species interactions***Description**

Simulate species interactions based on the matching between 2 traits. This is based on a code that originally simulated species abundances in different sites based on their environmental niches.

**Usage**

```
compas2d(
  nconsumer = 40,
  nresource = 100,
  le_grad = 100,
  ratio_grad = 0.8,
  consumer_ab = NULL,
  resource_ab = NULL,
  min_ab = 1,
  max_ab = 100,
  ninter = 100,
  delta = 1,
  mean_tol = 2,
  sd_tol = 10,
  buffer = 1,
  col_prefix = "c",
  row_prefix = "r",
  rowvar_prefix = "tr",
  remove_zeroes = FALSE,
  return_intermediate = FALSE
)
```

**Arguments**

nconsumer	Number of consumer species
nresource	Number of resource species
le_grad	gradient length for traits. The values are comprised between 0 and le_grad.
ratio_grad	length of the second gradient as a ratio of the first one.
consumer_ab	Optional vector of abundances for consumers. If missing, abundances are taken from a uniform law between min_ab and max_ab.
resource_ab	Optional vector of abundances for resource species. If missing, abundances are taken from a uniform law between min_ab and max_ab.
min_ab	minimal abundance value for the uniform law (if consumer_ab and/or resource_ab are absent)
max_ab	maximal abundance value for the uniform law (if consumer_ab and/or resource_ab are absent)
ninter	Total number of observations in the matrix
delta	Exponent between 0 and 1 to give more (1) or less (0) weight to trait matching relatively to abundance.

mean_tol	Mean value for the distribution of the normal law in which consumers niche standard deviation are drawn (for each trait).
sd_tol	Standard deviation for the distribution of the normal law in which consumers niche standard deviation are drawn (for each trait).
buffer	buffer to allow consumer species optima to fall outside the gradient by 0 - buffer ; le_grad + buffer.
col_prefix	Prefix for the column names of the matrix (which will be prefix + index). Defaults to "c" for "consumers".
row_prefix	Prefix for the row names of the matrix (which will be prefix + index). Defaults to "r" for "resources".
rowvar_prefix	Prefix for the column names of the row variable (which will be prefix + 1 or 2). Defaults to "tr" for "resource traits".
remove_zeroes	If there are unobserved species, should we keep them in the final matrix?
return_intermediate	Return intermediate results? If yes, will return the probability of interaction based on matching only, on abundances only and the predicted mix of matching and abundances (before sampling).

## Details

This function models species interactions as arising from the matching between their traits. Traits are drawn from a uniform law with parameters determined by `le_grad` and `ratio_grad` (for the first and second trait). The matching is based on a multivariate normal distribution with means corresponding to the difference between trait values, and variance determined by the variance of consumers' niches. It is then mixed to the abundances (using the parameter `delta`) and sampled to obtain `ninteractions`.

## Value

An object of class `compas`, which is a list with 3 components (4 if `return_intermediate`):

- `comm`: the interaction matrix of dimension `nresource` x `nconsumer`. Rows and columns are named like species (see `row_prefix` and `col_prefix`).
- `rowvar`: resource traits values. It is a `nresource` x 2 matrix. Rows are named like resource species (see `row_prefix`) and column are named like resource traits (see `rowvar_prefix`).
- `colvar`: consumer traits values. It is a `nconsumer` x 2 x 2 array. Rows are named like consumer species (see `col_prefix`). Column names in each array dimension are `mean` and `sd` (for the mean and standard deviation of the species niche, respectively).
- if `return_intermediate`, a fourth component named `intermediate` is returned. The first element is named `p_matching` and contains the matrix of probabilities of interactions based only on matching. The second `ab_neutral` contains the matrix of count of interactions as predicted by species abundances. The third `p_mix` contains the matrix of probabilities of interactions taking into account the mix of matching and abundances.

---

create_code	<i>Create code</i>
-------------	--------------------

---

**Description**

Create a code for each (unique) value of a vector

**Usage**

```
create_code(col, code_pattern = NA, name = NA)
```

**Arguments**

col	vector to code
code_pattern	first letter to add before code
name	optional column names for the output dataframe

**Value**

dataframe with 2 columns: one with original vector (named col by default), the other with the corresponding codes (named col\_code by default)

---

df_to_matrix	<i>Interaction dataframe to matrix</i>
--------------	--

---

**Description**

Transforms an interaction dataframe into an incidence matrix.

**Usage**

```
df_to_matrix(df, rows = "plant_species_code", columns = "animal_species_code")
```

**Arguments**

df	the dataframe to transform. Must have species names (columns which names are given by the parameters rows and columns), and a third column frequency with the interaction frequency/strength index.
rows	Name of the column with the species names that will be the rows of the matrix.
columns	Name of the column with the species names that will be the columns of the matrix.

**Value**

Returns an incidence matrix with named rows and columns filled with the values in frequency.

---

filter_matrix	<i>Filter matrix</i>
---------------	----------------------

---

### Description

Filter incidence matrix to keep only species that interact with more partners that a given threshold.

### Usage

```
filter_matrix(mat, thr)
```

### Arguments

mat	The incidence matrix.
thr	The minimal number of different interacting partners a species must have (a species must have $\geq$ thr different interacting partners). It is not weighted (for instance, if thr = 2 then a species interacting 100 times with a unique other partner will be removed.)

### Value

The interaction matrix but where the rows and/or columns with species that do not interact above the threshold removed.

---

get_best_model	<i>Get best model</i>
----------------	-----------------------

---

### Description

Compare 2 linear models and returns the best

### Usage

```
get_best_model(lmsimple, lm2, method = c("LRT", "AIC"), alpha = 0.05)
```

### Arguments

lmsimple	First model (if using method = "LRT", must be nested in lm2)
lm2	Second model
method	The method to use to compare models: likelihood ratio test (method = "LRT") for nested linear models comparison or AIC (method = "AIC")
alpha	Significance threshold to consider when using method = "LRT"

### Value

The best model

get\_cor

*Get correlations***Description**

Measure correlations between niche obtained from CA or reciprocal scaling and the fundamental and realized niches.

**Usage**

```
get_cor(niches)
```

**Arguments**

**niches**                      An object containing the fundamental, realized and multivariate niches (output of [get\\_niches](#))

**Value**

A list of lists. The first element (named *fundamental*) contains the correlations with the fundamental niche values and the second element (named *realized*) contains the correlations with the realized niche values. Each sublist is a list of 2 dataframes:

- **mean\_cor**: correlations of niches optima with the CA coordinates.
- **sd\_cor**: correlations of niches breadths with reciprocal scaling standard deviations.

These dataframes have the following columns:

- **axis**: correlations measured on which multivariate axis?
- **type**: the type of individuals for which the measure is (rows or columns?). Column or row codes are chosen with *rowname* and *colname* from the parameter *niches*.
- **cor**: the correlation value

get\_mean\_sd

*Get mean and standard deviation***Description**

Return the mean and standard deviation from a reciprocal scaling analysis

**Usage**

```
get_mean_sd(recscal, ax = 1:2)
```

**Arguments**

**recscal**                      A dataframe (expected to be the output of [ade4::reciprocal.coa](#)). The first columns must contain the reciprocal scaling scores and the last 3 columns are Row, Col and Weight.

**ax**                              the axes for which to compute mean and variance (the function will use the n-th first columns from the *recscal* dataframe).

**Value**

A list of 4 matrices, each containing the mean or standard deviation per axes. Each matrix has one column per axis given in ax and as many rows as there are grouping levels.

- rowstd is the standard deviation per row
- rowmean is the mean per row
- colstd is the standard deviation per column
- colmean is the mean per column

Matrices rownames are the row or column groups (species names). Column names are the same as in recscal.

---

get_niches	<i>Get niches</i>
------------	-------------------

---

**Description**

Returns the fundamental and realized niches and the niches returned by reciprocal scaling.

**Usage**

```
get_niches(
  rec,
  comm,
  consumer_niche,
  resource_traits,
  rowname = "row",
  colname = "col"
)
```

**Arguments**

rec	A dataframe (expected to be the output of <a href="#">ade4::reciprocal.coa</a> ). The first 2 columns must contain the reciprocal scaling scores (there can be more axes, but only the first 2 will be used.) The last 3 columns are Row, Col and Weight.
comm	The observed community matrix. It is used to get the resource species niche width.
consumer_niche	A nx2x2 array with n rows (n species), 2 columns corresponding to species niche optima and standard deviation (corresponding to their multivariate normal niche) and 2 traits stored in the dimension. There is 1 row per species.
resource_traits	A mx2 matrix. Each row corresponds to a species and each column to a trait value.
rowname	How to name the first element of the list (resources/rows) (e.g. could be "plants")
colname	How to name the second element of the list (resources/rows) (e.g. could be "birds")



## Details

The fundamental niche optimum of a consumer is the mean of its niche (from the `consumer_niche` table). For a resource, it corresponds to the traits from the `resource_traits` table. The fundamental niche breadths are computed for consumers only, and corresponds to the breadths of their niche (from the `consumer_niche` table).

The realized niche optima are computed as the weighted means of the traits of species' interacting partners: therefore, the realized niche of a consumer is the weighted mean of the traits of the resource species it interacts with, and reciprocally. The realized niche breadths are computed as the weighted standard deviations of the traits of a species' interacting partners. Therefore, the realized niche breadth of a consumer is the weighted standard deviation of the traits of the resource species it interacts with, and reciprocally.

Niche optima computed with multivariate methods correspond to the reciprocal scaling mean and niche breadths correspond to the reciprocal scaling standard deviation.

## Value

A list of lists of lists. The first element (named `rowname`) contains the niches for the resources and the second element (named `colname`) contains the niches for the consumers. Each element contains 2 lists:

- `mean`: niche optima
- `sd`: niche breadths

For each of these sublists, there are 3 matrices elements named:

- `fundamental`: the fundamental niche based on species traits
- `realized`: the realized niche based on interacting species traits
- `mvar`: niche parameters corresponding to the mean and standard deviations given by reciprocal scaling.

These matrices describe the value for each species (in rows) along each trait/axis (in column).

---

get\_pred

*Get the predictions of a linear model*

---

## Description

Returns the model prediction on given data range.

## Usage

```
get_pred(
  dat_predict,
  lmpred,
  by = 0.001,
  level = 0.95,
  interval = c("confidence", "prediction")
)
```

**Arguments**

<code>dat_predict</code>	The data to predict
<code>lmpred</code>	The model. It must have only one explanatory variable named mean.
<code>by</code>	The step to use for the range of <code>dat_predict</code> values
<code>level</code>	Confidence level used for the prediction. Defaults to 0.95.
<code>interval</code>	The type of interval to use: the argument is used in <code>stats::predict.lm</code> and has the same interpretation. <code>confidence</code> gives the confidence interval around the mean of the observations whereas <code>prediction</code> gives the confidence interval of predicted values.

**Value**

The model's prediction as a dataframe with columns:

- `fit`: the predicted value
- `lwr`: lower bound of the confidence interval (see arguments `level` and `interval`)
- `upr`: upper bound of the confidence interval (see arguments `level` and `interval`)
- `x`: the explanatory variable

---

`lm_labels`
*Labels of a linear model*


---

**Description**

Get the labels of a linear model to display on a plot.

**Usage**

```
lm_labels(mod, a)
```

**Arguments**

<code>mod</code>	The linear model. It is expected to have 2 or 3 coefficients of the form $y = ax + b$ or $y = ax + cx^2 + b$ .
<code>a</code>	The axis to consider. Used for the subscript of the variables.

**Details**

Inspired from <https://r-graphics.org/recipe-annotate-facet>

**Value**

A dataframe with columns `formula` and `r2` Containing respectively the model equation and coefficient of determination. The formulas are written to be parsed later in the plot. The variables' names are  $s_a$  for the predicted value and  $m_a$  for the predictor.

---

matrix_to_df	<i>Interaction matrix to dataframe</i>
--------------	--

---

### Description

Transform interaction matrix (incidence matrix) to a dataframe

### Usage

```
matrix_to_df(mat, colnames = "animals", rownames = "plants", tofactor = TRUE)
```

### Arguments

mat	Interaction matrix. The function assumes this matrix has row and column names, that are then used to fill the output dataframe columns.
colnames	The name to give to the column containing the matrix column names (defaults to "animals").
rownames	The name to give to the column containing the matrix row names (defaults to "plants").
tofactor	Should the final columns colnames and rownames be of type factor?

### Value

A dataframe with 3 columns:

- colnames containing the initial matrix column names
- rownames containing the initial matrix row names
- value containing the matrix values

---

multiplot	<i>Plot bi, tri or quadriplots</i>
-----------	------------------------------------

---

### Description

Plot bi, tri or quadriplots from a multivariate analysis

### Usage

```
multiplot(
  indiv_row = NULL,
  indiv_col = NULL,
  indiv_row_lab = rownames(indiv_row),
  indiv_col_lab = rownames(indiv_col),
  var_row = NULL,
  var_col = NULL,
  var_row_lab = rownames(var_row),
  var_col_lab = rownames(var_col),
  row_color = "black",
```

```

    col_color = "black",
    eig = NULL,
    x = 1,
    y = 2,
    xlim = NULL,
    ylim = NULL,
    grad = 4,
    mult = 1,
    title = NULL,
    max.overlaps = 20,
    labsize = 3,
    alphapoints = 0.8
  )

```

### Arguments

<code>indiv_row</code>	Matrix of individuals coordinates for rows (type <code>dudi\$li</code> )
<code>indiv_col</code>	Matrix of individuals coordinates for columns (type <code>dudi\$co</code> )
<code>indiv_row_lab</code>	Labels for row individuals
<code>indiv_col_lab</code>	Labels for columns individuals
<code>var_row</code>	Matrix of variables coordinates for rows (type <code>dudi\$corR</code> )
<code>var_col</code>	Matrix of variables coordinates for columns (type <code>dudi\$corQ</code> )
<code>var_row_lab</code>	Labels for row variables
<code>var_col_lab</code>	Labels for columns variables
<code>row_color</code>	Color for the row individuals or variables
<code>col_color</code>	Color for the columns individuals or variables
<code>eig</code>	Eigenvalues vector
<code>x</code>	Axis to use for the x-axis
<code>y</code>	Axis to use for the y-axis
<code>xlim</code>	x-axis limits. Defaults to data range if not specified
<code>ylim</code>	y-axis limits. Defaults to data range if not specified
<code>grad</code>	graduations for the major.grid
<code>mult</code>	factor to multiply the vectors (arrows on the plot)
<code>title</code>	plot title
<code>max.overlaps</code>	<code>max.overlaps</code> argument for <code>ggrepel::geom_text_repel</code>
<code>labsize</code>	text size for point labels
<code>alphapoints</code>	Transparency value for points

### Value

a `ggplot` representing row and/or column individuals as points, possibly including row and/or column variables as vectors

---

plot_corcircle	<i>Plot correlation circle</i>
----------------	--------------------------------

---

## Description

Plot a correlation circle

## Usage

```
plot_corcircle(
  cor,
  label = rownames(cor),
  col = "black",
  lty = "solid",
  cor2 = NULL,
  label2 = rownames(cor2),
  col2 = "black",
  lty2 = "solid",
  xax = 1,
  yax = 2,
  xlab = NULL,
  ylab = NULL,
  eig = NULL,
  col_bg = "grey30",
  mar = 0.01,
  labsize = 3.88,
  lim = 1,
  clip = "off"
)
```

## Arguments

cor	dataframe containing the coordinates of the variables for each axis in columns
label	label to display on the arrows for the first set of variables
col	color for the first set of arrows
lty	linetype for the first set of variables
cor2	optional second set of variables (for double constrained analyses)
label2	label to display on the arrows for the second set of variables
col2	color for the second set of arrows
lty2	linetype for the second set of variables
xax	index of the column of the matrix to use for the x-axis
yax	index of the column of the matrix to use for the y-axis
xlab	Custom x-label
ylab	Custom y-label
eig	Eigenvalues vector (to display percentage of variation explained by each axis)
col_bg	color to use for background elements (circle and $y = 0$ and $x = 0$ lines)
mar	margin between arrow tips and label

labsize	Size for arrow labels.
lim	limits for the x and y axes: a unique number that gives the distance between the plot limit and zero (so that the circle is centered)
clip	constrain the labels to be inside the plot? If yes, change to "on".

**Value**

A ggplot with the correlation circle where one or two sets of variables are represented as arrows inside the circle of radius 1.

---

plot_eig	<i>Plot eigenvalues</i>
----------	-------------------------

---

**Description**

Barplot of eigenvalues

**Usage**

```
plot_eig(eigenvalues, showrank = FALSE)
```

**Arguments**

eigenvalues	The eigenvalues
showrank	Should the x-axis display the rank of the eigenvalues?

**Value**

A ggplot with the eigenvalues displayed as a barplot.

---

plot_lm_mean_sd	<i>Plot linear model</i>
-----------------	--------------------------

---

**Description**

Plot the linear models that predicts standard deviation with mean of the reciprocal scaling.

**Usage**

```
plot_lm_mean_sd(
  df,
  dat_pred,
  col = "black",
  ylab = "Standard deviation",
  xlab = "Mean",
  lab = NULL,
  text_size = 3,
  points_size = 1,
  facet = NULL,
```

```

    facet_dir = c("h", "v"),
    strip.position = "top",
    nudge_x = NULL,
    max.overlaps = 5
  )

```

### Arguments

df	The dataframe containing the true data. It must have columns mean (used for the x-axis) and sd (y-axis).
dat_pred	The dataframe containing the predicted data (assumed to be over the true data range). It is assumed to be obtained with <a href="#">get_pred</a> . It must have columns x, fit, lwr, upr.
col	Color of the data points and prediction line. If there are facets, it should have length 2 (one color per facet).
ylab	Label of the y-axis
xlab	Label of the x-axis
lab	A label to add to the graph. If provided, will be added in the top-left corner. It must be a dataframe with at least one column R2 (to create this dataframe, see <a href="#">lm_labels</a> ). If facet is not NULL, it must have a column type too. If it has a column formula, the formula will be displayed along with the coefficient of determination.
text_size	Size of the text (points labels and R squared). Other text (axes titles, panels) is scaled to be bigger.
points_size	Size of the points
facet	A column to use for facetting. Defaults to NULL (no facets).
facet_dir	Direction of the facet grid: "h" for "horizontal and "v" for "vertical" (useful only if facet is not NULL)
strip.position	Strip position (useful only if facet is not NULL). See the documentation of <a href="#">ggplot2::facet_wrap</a> (argument strip.position).
nudge_x	Nudge to use for top left corner labels. If NULL, defaults to 1 % of the range.
max.overlaps	max.overlaps argument for <a href="#">ggrepel::geom_text_repel</a>

### Value

A ggplot object showing standard deviation vs mean, along with the predicted values of the linear model

---

plot\_matrix

*Plot matrix*

---

### Description

Plot an interaction matrix

**Usage**

```
plot_matrix(
  mat,
  tile_fill = "white",
  col = "black",
  alpha = FALSE,
  max_size = max(mat) * 0.7,
  base_size = 12,
  xlab = "Animals",
  ylab = "Plants",
  trans = "identity",
  breaks = waiver(),
  legend_title = "Frequency"
)
```

**Arguments**

mat	The matrix. It represents the interaction frequency of species interactions (it is assumed to be plants x animals).
tile_fill	Color fill of tiles. Use NA for no fill.
col	Color of the points
alpha	If true, will make less abundant interactions more transparent
max_size	The maximum circle area to be used in scale_size_area. Defaults to 0.7 times the actual maximum.
base_size	Base text size
xlab	Title of the x axis
ylab	Title of the y axis
trans	Transformation to apply to the scaling of points.
breaks	Breaks to use for the legend of the points size.
legend_title	Title of the legend

**Value**

A ggplot representing the interaction matrix as a rectangle with points whose sizes correspond to the interaction frequencies

---

plot_reciprocal	<i>Plot reciprocal scaling</i>
-----------------	--------------------------------

---

**Description**

Plot reciprocal scaling



**Usage**

```
plot_reciprocal(
  dudi,
  recscal,
  xax = 1,
  yax = 2,
  labsize = 1,
  psize = 1,
  xlab = NULL,
  ylab = NULL,
  group = c("li", "co"),
  col = "cornflowerblue",
  alpha = 0.2,
  plot_arrows = TRUE,
  plot_points = TRUE,
  plot_labels = TRUE,
  ...
)
```

**Arguments**

dudi	An object of class <code>ade4::dudi</code> to pair with the analysis. Used for the eigenvalues (always) and for the arrows of explanatory variables if required. It must be of class <code>coa</code> , <code>pcaiv</code> or <code>dpcaiv</code> (obtained with the new function <code>dpcaiv2</code> ).
recscal	A dataframe (expected to be the output of <code>reciprocal.coa</code> function). The first columns must contain the reciprocal scaling scores and the last 3 columns are Row, Col and Weight.
xax	The index of the multivariate axis to plot on the x-axis (the column <code>xax</code> from the <code>recscal</code> dataframe).
yax	The index of the multivariate axis to plot on the y-axis (the column <code>yax</code> from the <code>recscal</code> dataframe).
labsize	Size of the ellipses labels
psize	Size of the points
xlab	Custom x-label
ylab	Custom y-label
group	The group to use for the ellipses ( <code>co</code> or <code>li</code> )
col	The ellipses colors
alpha	The ellipses transparency
plot_arrows	Whether to plot the arrows for the variables
plot_points	Plot the data points?
plot_labels	Plot ellipses labels?
...	Additional parameters passed to <code>s.class</code>

**Value**

A plot with the points grouped by ellipses.

# Index

add\_params, [2](#)  
ade4::dudi, [17](#)  
ade4::reciprocal.coa, [7](#), [8](#)  
  
combine\_cor, [2](#)  
compas2d, [3](#)  
create\_code, [5](#)  
  
df\_to\_matrix, [5](#)  
  
filter\_matrix, [6](#)  
  
get\_best\_model, [6](#)  
get\_cor, [2](#), [7](#)  
get\_mean\_sd, [7](#)  
get\_niches, [7](#), [8](#)  
get\_pred, [9](#), [15](#)  
ggplot2::facet\_wrap, [15](#)  
ggrepel::geom\_text\_repel, [15](#)  
  
lm\_labels, [10](#), [15](#)  
  
matrix\_to\_df, [11](#)  
multiplot, [11](#)  
  
plot\_corcircle, [13](#)  
plot\_eig, [14](#)  
plot\_lm\_mean\_sd, [14](#)  
plot\_matrix, [15](#)  
plot\_reciprocal, [16](#)  
  
stats::predict.lm, [10](#)