

Package: PRC (via r-universe)

October 15, 2026

Type Package

Title Principal Response Curves (PRC) For Displaying Multivariate Main Effects And Interactions

Version 2.0.4

Maintainer Cajo J.F. ter Braak <cajo.terbraak@wur.nl>

Description PRC is a small `\code{vegan}`- and `\code{LMMsolver}`-based package to create and display Principal Response Curves (PRC) and smooth Principal Response Curves (PRC). PRC is a special case of Redundancy Analysis (rda) for multivariate responses in designed experiments and repeated measurement designs. PRC expresses ordination scores as deviation from a reference, usually to visualize multivariate A-dependent effects (e.g time-dependent effects) of a treatment B in (un)balanced factorial experiments. The main user functions are `\code{\link{doPRC}}`, `\code{\link{plotPRC}}`, `\code{\link{plotPRC2d}}` and `\code{\link{smoothPRC}}`. The easiest example is the code demo `\code{PRC_pyrifos}`. This version includes `\code{\link{smoothPRC}}` using `\code{\link{set_smoothPRC_model}}`.

URL <https://doi.org/10.1002/etc.5620180207>,
<https://doi.org/10.1016/j.chemolab.2023.104898>

BugReports <https://github.com/CajoterBraak/PRC/issues>

Depends R (>= 4.1.0)

License GPL-3 | file LICENSE

Encoding UTF-8

LazyData true

Imports LMMsolver(>= 1.0.13), permute, vegan (>= 2.6-8), spam, dplyr, ggplot2 (>= 3.5.1), latex2exp, ggrepel, gridExtra, rlang, Rcpp

LinkingTo Rcpp

Suggests rmarkdown, knitr

NeedsCompilation yes

VignetteBuilder knitr

Config/roxygen2/version 8.1.1

Repository <https://cajoterbraak.r-universe.dev>

Date/Publication 2026-10-15 14:34:20 UTC

RemoteUrl <https://github.com/CajoterBraak/PRC>

RemoteRef HEAD

RemoteSha 2d620fef9edda262c5a49d8627569b6da76cf75f

Contents

anova.smoothPRC	3
biomonitoring	3
CPFNUTphytoplankton	4
doPRC	5
fit_full_model_columns	8
fit_full_model_rows	8
fit_null_model_columns	9
fit_null_model_rows	9
fvalues4levels	10
get_focal_and_conditioning_factors	11
initialize_pwr_alg	12
Ossenkampen	12
plot_sample_scores_cdt	13
plot_smoothPRC_cdt	15
plot_species_scores_2d	17
plot_species_scores_bk	19
plotPRC	21
plotPRC2d	24
plotsmoothPRC	28
PRC_scores	29
rdacca_taxon_stats	32
set_smoothPRC_model	33
SimData	36
smoothPRC	37
update_fit_state	40

Index	41
--------------	-----------

anova.smoothPRC	<i>Permutation Test for Smooth PRC</i>
-----------------	--

Description

Permutation Test for Smooth PRC

Usage

```
## S3 method for class 'smoothPRC'
anova(object, ..., permutations = 19, verbose = TRUE)
```

Arguments

object	an object from smoothPRC .
...	unused.
permutations	a list of control values for the permutations as returned by the function how , or the number of permutations required (default 19) or a permutation matrix where each row gives the permuted indices.
verbose	logical for printing progress of the permutations

Details

The test takes account of additional covariates in fixed of the LLMsolver model. The test is approximate without permutation of whole time series. The test is based on the first axis.

biomonitoring	<i>Biomonitoring example from van den Brink et al. (2009)</i>
---------------	---

Description

The data frame `biomonitoring` contains the design and species data from van den Brink et al.(2009). The data is on biomonitoring of macroinvertebrate species in the Rhine and Meuse rivers, near the towns of Kampen and Grave, respectively, from 1992 - 2000, sampled six times per year, except in 1998 (4x), 1999 and 2000 (7x). The first example in the demo `monitoring_EMA2009` uses reference coding of an redundancy analysis without covariates so as to emphasize the compositional difference from a reference. The second example is a more regular PRC, contrasting the compositional changes at one site without those in another.

- sample character identifier for the sample.
- year numeric year of sampling
- location location of sampling: factor with levels `c("Grave","Kampen")`
- counts of 139 macroinvertebrate species

References

van den Brink, PJ, den Besten, PJ, bij de Vaate, A and ter Braak, CJF, 2009. Principal response curves technique for the analysis of multivariate biomonitoring time series. *Environmental Monitoring and Assessment*. 152, 271-281. doi:[10.1007/s1066100803146](https://doi.org/10.1007/s1066100803146)

CPFNUTphytoplankton	<i>Phytoplankton data from the chlorpyrifos-nutrient (CPF-NUT) experiment from van den Brink & ter Braak (1998)</i>
---------------------	---

Description

The dataframe CPTNUT_phytoplankton contains the design and phytoplankton data from the microcosm example of PRC from van den Brink & ter Braak (1998). In this experiment from van Donk et al. (1995), there are three treatments: control (con), nutrients (nut) and nutrients with chlorpyrifos application (cpf-nut), each with four cosms (replications) sampled 13 times (twice before nutrients were added and six times before the additional application of chlorpyrifos). The phytoplankton data are counts and there are 26 species in total.

Format

A data frame with 156 rows and 31 columns:

- samples character identifier for the sample.
- Time factor coding for the week of sampling with respect to first application of nutrients.
- Treatment treatment factor as in van den Brink & ter Braak (1998). The levels are con, nut and cpf-nut for the four control cosms, the four cosms that receive nutrients after the first two sampling points, and the four cosms to which both nutrients and chlorpyrifos are applied, respectively. Note that the cosms of the nutrient-chlorpyrifos treatment are coded herein as nut in the weeks before the first chlorpyrifos application, as they are treated experiment-wise similarly in these weeks. But, note that the samples from con and nut are coded differently on pre-treatment times (although they are similar experiment-wise), so as to see the pre-treatment variability. See also the factor con_nut_cpf.
- Plot numeric identifier for the microcosm.
- con_nut_cpf treatment factor. Before application of nutrients or chlorpyrifos, the samples are all treated similarly, but are coded differently (as often in PRC).
- cell counts of 25 different phytoplankton taxa in 1L, and density for the 26th species, Volvox. For details, see van den Donk et al. (1995).

Source

van Donk et al. (1995), van den Brink & ter Braak (1998)

References

- van den Brink, PJ & ter Braak, CJF (1998). Multivariate analysis of stress on experimental ecosystems by principal response curves and similarity analysis. *Aquatic ecology*, 32, 163-178. doi:[10.1023/A:1009944004756](https://doi.org/10.1023/A:1009944004756)
- van Donk E, Prins H, Voogd HM, Crum SJH and Brock TCM (1995) Effects of nutrient loading and insecticide application on the ecology of Elodea – dominated freshwater microcosms. I. Responses of plankton and zooplanktivorous insects. *Arch Hydrobiol* 133: 417–439. doi:[10.1127/archiv-hydrobiol/133/1995/417](https://doi.org/10.1127/archiv-hydrobiol/133/1995/417)

doPRC

Perform a Principal Response Curve analysis (PRC)

Description

doPRC perform a Principal Response Curve analysis (PRC) using the S3 formula interface of [rda](#) or [cca](#)

Usage

```
doPRC(
  formula,
  scale = FALSE,
  referencelevel = NULL,
  rank = 2,
  flip = rep(FALSE, rank),
  scaling = "ms",
  method = "rda",
  data
)
```

Arguments

- | | |
|----------------|--|
| formula | formula specifying how the response (species data) must modeled by predictors and covariates. A typical formula is <code>Y~treatment*week+Condition(week)</code> , where Y is the global response data matrix or data frame. Typically these are community (species) data. If count-like, they probably should be log transformed prior to the analysis, because a multiplicative model has more realism for non-negative data than an additive model. |
| scale | Scales species to unit variance (like correlations) if method is "rda" (default: FALSE), else void. See rda . |
| referencelevel | numeric or character level(s) to be used as reference level or levels of the focal treatment(s); default: 1 (first level(s)). With doPRC , the focal treatment is (or treatments are) given in the resulting list at <code>result\$focal_and_conditioning_factors\$focal_factor`</code> |
| rank | number of axes to be computed; default 2. |

flip	logical value or vector; default: FALSE. Should the axes be flipped, i.e. reversed in orientation? flip can be numeric with values -1 and 1 for reverse and no reverse, respectively, i.e. multiply the scores by either -1 or 1.
scaling	character c("ms", "ss"). Scales the axes of an RDA object to mean square species scores of 1 ("ms") or a sum of squares of 1 ("ss"). The only scaling for a CCA object is "ss" (weighted mean square = 1, with weight = species total abundance). For a dbrda object, there are no species scores and the scaling is like "ss". The default scaling ("ms" for RDA and "ss" for CCA) has the advantage that the PRC treatment and sample scores measure the effect sizes of the 'average' species as their (weigthed) mean square is 1 (sd = 1); it is the scaling used in Canoco5 (www.canoco5.com). Scaling "ss" is the default in prcomp and used in ter Braak (2023).
method	character rda or cca. Default: "rda"
data	Data frame containing the variables on the right hand side of the model formula.

Details

The function performs PRC using [rda](#) or [cca](#) with the specified formula and applies [PRC_scores](#) to the result.

Value

an object as a [cca.object](#) with additional entries as output of [PRC_scores](#). The assigned classes are c("PRC", "rda", "cca").

References

- ter Braak (2023) Redundancy analysis includes analysis of variance-simultaneous component analysis (ASCA) and outperforms its extensions Chemometrics and Intelligent Laboratory Systems. [doi:10.1016/j.chemolab.2023.104898](https://doi.org/10.1016/j.chemolab.2023.104898)
- ter Braak, C.J.F. & te Beest, D.E. 2022. Testing environmental effects on taxonomic composition with canonical correspondence analysis: alternative permutation tests are not equal. *Environmental and Ecological Statistics*. 29 (4), 849-868. [doi:10.1007/s10651022005454](https://doi.org/10.1007/s10651022005454)
- van den Brink, P.J. & ter Braak, C. (1998) Multivariate analysis of stress in experimental ecosystems by Principal Response Curves and similarity analysis. *Aquatic Ecology*, 32, 163-178. [doi:10.1023/A:1009944004756](https://doi.org/10.1023/A:1009944004756)
- van den Brink, P.J. & ter Braak, C.J.F. (1999) Principal Response Curves: Analysis of time-dependent multivariate responses of a biological community to stress. *Environmental Toxicology and Chemistry*, 18, 138-148. [doi:10.1002/etc.5620180207](https://doi.org/10.1002/etc.5620180207)
- van den Brink, P.J., den Besten, P., bij de Vaate, A. & ter Braak, C.J.F. (2009) Principal response curves technique for the analysis of multivariate biomonitoring time series. *Environmental Monitoring and Assessment*, 152, 271-281. [doi:10.1007/s1066100803146](https://doi.org/10.1007/s1066100803146)

See Also

[plotPRC](#), [plotPRC2d](#)

Examples

```

library(PRC)
data(pyrifos, package = "vegan") #log-transformed species data from package vegan
#the chlorpyrifos experiment from van den Brink & ter Braak 1999
Design <- data.frame(week=gl(11, 12, labels=c(-4, -1, 0.1, 1, 2, 4, 8, 12, 15, 19, 24)),
                     dose=factor(rep(c(0.1, 0, 0, 0.9, 0, 44, 6, 0.1, 44, 0.9, 0, 6), 11)),
                     ditch = gl(12, 1, length=132))

#

mod_prc <- doPRC(pyrifos ~ dose:week + Condition(week), data = Design)
mod_prc$focal_and_conditioning_factors$focal factor` # as expected: "dose"
print(mod_prc)

#               Proporti Rank
# Total          1.0000
# Conditional    0.2192   10 (between-week variation)
# Constrained    0.3346   44 (between-treatment variation across weeks)
# Unconstrained  0.4462   77 (within-week variation)

# without lines, threshold is 7
plotPRC(mod_prc)
# this threshold is about P = 0.01 for PRC1 in the regression "species_k ~ week+PRC1"
# and retains 34 species to be plotted with names
sum(mod_prc$species[, "Fratio1"] > 7) # 34
# only the PRC curves:
plotPRC(mod_prc, plot = 0, symbols_on_curves = TRUE)
# with lines and a stronger threshold
plotPRC(mod_prc, plot = "ditch", threshold = 10, width = c(4,1))

# modifying the plot
gg <- plotPRC(mod_prc, plot = "ditch", width = c(4,1), verbose = FALSE)
# PRC plot of samples (c_dt)
p1 <- gg$separateplots$treatments +
  ggplot2::ggtitle(paste("new title:", latex2exp::TeX("$c_{dt}$")))

p2 <- gg$separateplots$species +
  ggplot2::ylab("new title: loadings") # loadings of species (b_k)
# Assign these plots to symbols and use grid.arrange to produce the plot
#you like, for example:
gridExtra::grid.arrange(p1+ ggplot2::ylab("")+
  ggplot2::xlab("week since chorpyrifos application") + ggplot2::ggtitle(""),
  p2, ncol = 2, widths = c(4,1),
  top = "new title", left = "PRC curves of treaments",
  right = "PRC species loadings")

# test significance of PRC/RDA axes -----

## Ditches are randomized, we have a time series, and are only
## interested in the first axis
library(vegan)
ctrl <- how(plots = Plots(strata = Design$ditch, type = "free"),
            within = Within(type = "none"), nperm = 99)

```

```
anova(mod_prc, by = "axis",permutations= ctrl, model = "reduced",
      cutoff = 0.10)
```

fit_full_model_columns
<i>Fit the full column model (internal)</i>

Description

Returns an object with a fitted() method.

Usage

```
fit_full_model_columns(model, ...)
```

Arguments

- | | |
|-------|-----------------------|
| model | Model specification. |
| ... | Additional arguments. |

fit_full_model_rows	<i>Fit the full row model (internal)</i>
---------------------	--

Description

Returns an object with a fitted() method.

Usage

```
fit_full_model_rows(model, ...)
```

Arguments

- | | |
|-------|-----------------------|
| model | Model specification. |
| ... | Additional arguments. |

`fit_null_model_columns`*Fit the null column model (internal)*

Description

Returns an object with a fitted() method.

Usage

```
fit_null_model_columns(model, ...)
```

Arguments

<code>model</code>	Model specification.
<code>...</code>	Additional arguments.

`fit_null_model_rows` *Fit the null row model (internal)*

Description

Returns an object with a fitted() method.

Usage

```
fit_null_model_rows(model, ...)
```

Arguments

<code>model</code>	Model specification.
<code>...</code>	Additional arguments.

fvalues4levels	<i>Change numeric-like levels of a factor to a numeric</i>
----------------	--

Description

fvalues4levels changes levels of a factor to numeric variable in [set_smoothPRC_model](#) and for plotting of the PRC diagram.

Usage

```
fvalues4levels(object, fctr)
```

Arguments

object	a data frame or tibble.
fctr	name of a factor in object (character or symbol)

Value

a numeric named vector (names are levels)

Examples

```
Design <- data.frame(week=gl(11, 12, labels=c(-4, -1, 0.1, 1, 2, 4, 8, 12, 15, 19, 24)),
                    dose=factor(rep(c(0.1, 0, 0, 0.9, 0, 44, 6, 0.1, 44, 0.9, 0, 6), 11)))
fvalues4levels(Design, "week")

# fvalues4levels strips the starting k characters so as to find numbers,e.g.
Design$week2 <- factor(paste("Week", Design$week))
levels(Design$week2)
fvalues4levels(Design, "week2")
# they may not be in increasing order in general,
# but will be after sorting:
sort(fvalues4levels(Design, "week2"))
# the resulting numeric variable:
numvar <- fvalues4levels(Design, "week2")[Design[, "week2"]]
summary(numvar)
length(numvar)== nrow(Design)
```

get_focal_and_conditioning_factors

Get constraining and conditional variables from an rda or cca ordination

Description

get_focal_and_conditioning_factors derives the focal constraining variables(s) and (if present) the conditioning variables(s) from a [rda](#) or [cca](#) ordination specified via a formula for use in a PRC (with [doPRC](#) or [PRC_scores](#))

Usage

```
get_focal_and_conditioning_factors(object, factors_only = TRUE)
```

Arguments

object a result of [rda](#), [cca](#) specified via a formula (S3 method for class 'formula'), or a result of [doPRC](#) or [PRC_scores](#).

factors_only logical to exclude numeric variables from the focal factor(s). Default TRUE.

Value

A list with element ``focal factor`` and `condition`

References

ter Braak C.J.F. & Šmilauer P. (2018): Canoco reference manual and user's guide: software for ordination, version 5.1x. Microcomputer Power, Ithaca, USA, 536 pp. (<http://www.canoco5.com>)

See Also

[doPRC](#), [PRC_scores](#)

Examples

```
data(pyrifos, package = "vegan") #transformed species data from package vegan
#the chlorpyrifos experiment from van den Brink & ter Braak 1999
Design <- data.frame(
  week=gl(11, 12, labels=c(-4, -1, 0.1, 1, 2, 4, 8, 12, 15, 19, 24)),
  dose=factor(rep(c(0.1, 0, 0, 0.9, 0, 44, 6, 0.1, 44, 0.9, 0, 6), 11)))

library(vegan)
mod_rda <- rda( pyrifos ~ dose:week + Condition(week), data = Design)
species_scores= scores(mod_rda,choices =1:2, display="sp",scaling = 1)
library(PRC)
head(rdacca_taxon_stats(mod_rda)) # stats per species
get_focal_and_conditioning_factors(mod_rda) # factors from the model
```

```
species_scores= vegan::scores(mod_rda,choices =1:2, display="sp",scaling = 2)
# 1-d vertical plot of species scores (a cut of RDA1-scores at 0.5)
plot_species_scores_bk(species_scores)

# a cut of RDA1-scores at 1.0 (as scoresname is unset)
plot_species_scores_bk(species_scores, threshold = 14)

plot_species_scores_2d(species_scores)
plot_species_scores_2d(species_scores, label.repel = TRUE)
plot_species_scores_2d(species_scores, max.overlaps = 30, label.repel = TRUE)
```

initialize_pwr_alg	<i>initializes the power algorithm (internal)</i>
--------------------	---

Description

initializes the power algorithm (internal)

Usage

initialize_pwr_alg(model, Y, weights, axes)

Arguments

model	Model specification.
Y	response
weights	optional weights
axes	prior axes and covariable data

Ossenkampen	<i>Data from Ossenkampen experiment</i>
-------------	---

Description

The dataframe Ossenkampen contains the data from a long-term fertilizer experiment (1958-2007) in grassland with 98 response variables (abundances of the plant species that grew in the Ossenkampen plots, measured as counts in 100 subsamples), as analysed in ter Braak (2023).

The experiment started in 1958 with 12 plots arranged in two randomized complete blocks with six treatments: four types of fertilizer (K, P, PK or NPK), liming and no fertilizer. Eight years later (in 1966) each block was extended with two plots, one with N and another with NPK fertilization. The two limed plots behaved rather differently from the rest and are omitted here. In this data, the K and P plots are combined with the no fertilizer plots so that the control treatment (Cntrl) consisted of 6 plots, with the aim to make the variability among replicates more interesting. Three plots of

the control and one PK plot were not sampled in 1967 and two NPK samples were not sampled in 1984; these missing samples were not imputed in this data set.

The data analyzed in Berendse et al. (2021) also contained the limed plots and contained, for the multivariate analyses, imputations of the few missing samples after 1966.

- Sample numeric identifier for a sample
- Plot identifier for a sample (the identifier details the fertilizer treatment and Block)
- Block numeric identifier of the block (1 or 2)
- A year of sampling
- B fertilizer treatment, condensed to four treatments ("Cntrl" is the referencelevel)
- AgrCap count out of 100 subsamples of species AgrCap, and so on for the remaining 97 species

References

Berendse, F., Geerts, R.H.E.M., Elberse, W.T., Bezemer, T.M., Goedhart, P.W., Xue, W., Noordijk, E., ter Braak, C.J.F. & Korevaar, H. (2021) A matter of time: Recovery of plant species diversity in wild plant communities at declining nitrogen deposition. *Diversity and Distributions*, 27, 1180-1193. doi:10.1111/ddi.13266

Geerts, R.H.E.M., Bufer, C., Schils, P.C., Korevaar, H., Berendse, F. & Struik, P.C. (2021) The Ossekampen Grassland Experiment: Data underlying the publication: A matter of time: recovery of plant species diversity in wild plant communities at declining nitrogen deposition. doi:10.4121/13200398

ter Braak, C.J.F. (2023) Redundancy analysis includes analysis of variance-simultaneous component analysis (ASCA) and outperforms its extensions. *Chemometrics and Intelligent Laboratory Systems*, 104898. doi:10.1016/j.chemolab.2023.104898

plot_sample_scores_cdt

PRC diagram of treatment and plot lines or points without species loading

Description

plot_sample_scores_cdt creates a PRC diagram of the treatments without species loadings.

Usage

```
plot_sample_scores_cdt(
  object,
  treatment = NULL,
  condition = NULL,
  plot = 1,
  xvals = NULL,
  score_name = "PRC1",
```

```

size = c(2, 2),
symbols_on_curves = FALSE,
verbose = TRUE
)
```

Arguments

object	a result of doPRC or PRC_scores or a list with element PRCplus (a data frame with a variables to plot, see details).
treatment	character vector of names of factors in object\$PRCplus, the first two of which set the color and linetype of treatments. Default: derived from object\$'focal_and_conditioning_f'
condition	character vector of names of factors in object\$PRCplus, the first one being used as x-axis. Default: derived from object\$'focal_and_conditioning_factors'.
plot	0 or 1 (no/yes individual plot points) or character name of a factor in object\$PRCplus indicating the plots (microcosms, subjects or objects) in a longitudinal study so as to plot lines. Default: 1 yielding points instead of lines.
xvals	numeric values for the levels of the condition factor, or name of numeric variable in object\$PRCplus to be used for the x-axis. Default: NULL for which the values are obtained using the function fvalues4levels applied to object\$PRCplus[[condition[1]]]
score_name	name of scores to use as y-axis; default: "PRC1".
size	size of symbols and symbols in the legend. default: c(2, 2). If the length is not 2 than the first entry is replicated.
symbols_on_curves	logical, draw symbols on the PRC curves at the measurement times (default FALSE). If TRUE, symbols are for the treatment, if there is one, and for the second treatment if there are two.
verbose	logical for printing the current model, score_name, condition and treatment (default TRUE).

Details

plot_sample_scores_cdt can be used to plot any ordination score or variable against the condition with treatments indicated by colors and line type by setting score_name to the name of the variable of interest and argument plot to 0. If object is not a result of [doPRC](#) or [PRC_scores](#), set treatment and condition (or object\$'focal_and_conditioning_factors') and set object to a list with element PRCplus; object\$PRCplus should contain the names indicated by the arguments treatment, condition and, optionally, plot

- treatment: a factor for the treatments,
- condition: a factor or variable condition

If plot!=0, the scores (values) for individual sample points or lines must be in the variable "PRC1_E" or, in general, paste(score_name, "_E", sep = "").

Value

a ggplot object

See Also

[PRC_scores](#), [get_focal_and_conditioning_factors](#), [plotPRC](#) and [fvalues4levels](#)

Examples

```
data(pyrifos, package = "vegan") #transformed species data from package vegan
#the chlorpyrifos experiment from van den Brink & ter Braak 1999
Design <- data.frame(week=gl(11, 12, labels=c(-4, -1, 0.1, 1, 2, 4, 8, 12, 15, 19, 24)),
                     dose=factor(rep(c(0.1, 0, 0, 0.9, 0, 44, 6, 0.1, 44, 0.9, 0, 6), 11)),
                     ditch = gl(12, 1, length=132))

mod_rda <- vegan::rda( pyrifos ~ dose:week + Condition(week), data = Design)
Design_w_PRCs <- PRC_scores(mod_rda, focal_factor_name= "dose", referencelevel = "0",
                           rank = 2, data= Design, scaling ="ms", flip = FALSE)

plot_sample_scores_cdt(Design_w_PRCs, plot = "ditch")
plot_sample_scores_cdt(Design_w_PRCs)
plot_sample_scores_cdt(Design_w_PRCs, plot = "Ditch")
```

plot_smoothPRC_cdt	<i>smooth PRC diagram of treatment and plot lines or points without species loading</i>
--------------------	---

Description

plot_smoothPRC_cdt creates a PRC diagram of the treatments without species loadings.

Usage

```
plot_smoothPRC_cdt(
  object,
  axis = 1,
  mod_prc = NULL,
  with_classical_lines = TRUE,
  with_unconstrained_scores = TRUE,
  flip = FALSE,
  sample_times_only = FALSE
)
```

Arguments

object	a result of smoothPRC .
axis	the number of the axis to plot
mod_prc	a results of doPRC , <i>i.e.</i> a classical PRC with the same data and design. Default NULL, in which the classical PRC is computed using doPRC and the data in object.

```

with_classical_lines      logical to add or suppress classical PRC lines. Default TRUE, i.e. with classical
                           PRC lines.
with_unconstrained_scores logical to add or suppress the unconstrained scores in the smooth PRC showing
                           the scatter of the individual samples (cosms). Default TRUE, i.e. with individual
                           points.
flip                      logical. Should the axis be reversed? Default FALSE. Can be numeric with -1
                           meaning: reverse, and 1 meaning do not reverse.
sample_times_only         logical to PRClines based on the sampled times only, i.e. without interpolation.
                           Default FALSE.

```

Examples

```

# Analyze example data -----

data(SimData)
# extract design
Design0 <- SimData[,c("A","B")]
Design0$A <- factor(Design0$A); Design0$B <- factor(Design0$B)
#put levels in natural order
Design0$A<- factor(Design0$A, levels=c(levels(Design0$A)[-2],levels(Design0$A)[2]))
Y0 <- as.matrix(SimData[,-(1:3)])
ids <- which(Design0$A %in% "A10" & Design0$B %in% "B5")
#design <- "Complete" #
design <- "Empty cell"
if (design == "Complete"){ Design <- Design0;Y <- Y0} else {Design <-Design0[-ids,]; Y=Y0[-ids,]}
print(with(Design,table(A,B)))
names(Design)

# Design and Y for Fig 1 of ter Braak 2023 ready -----
dim(Y)
str(Design)
names(Design) <- c("Time","Treatment")
mod_prc <- doPRC(Y ~ Treatment:Time + Condition(Time), data = Design)
#Mod_prc <- vegan::prc(Y, Design$Treatment,Design$Time)
#Mod_prc$CCA$eig# identical to doPRC
b_mod_prc <- vegan::scores(mod_prc,choices= 1, display = "sp")

x_mod_prc <- vegan::scores(mod_prc,choices= 1, display = "lc")

summary(mod_prc)

smoothPRC_model <- set_smoothPRC_model(data = Design)
smoothPRC_model$spline
smooth_PRC <- smoothPRC(Y, model = smoothPRC_model)

summary(smooth_PRC$obj[[1]])
(cc <- cor(smooth_PRC$B[,1], b_mod_prc))
plot_species_scores_bk(-smooth_PRC$B, scoresname = "B1")

```

```

plot_species_scores_bk(b_mod_prc)
plot_smoothPRC_cdt(smooth_PRC, flip= TRUE)

plotsmoothPRC(smooth_PRC, flip= TRUE, threshold = 0)

# Demonstration of the fitted model
x1 <- scale(as.numeric(Design$Time))[,1]
x2 <- scale(as.numeric(Design$Treatment))[,1]

# with the scaling to mean square of 1 of the species scores b
summary(lm(smooth_PRC$X[,1]~ x1+ x2+ x1:x2 )) # R^2 ~1!

plotPRC(mod_prc)

```

plot_species_scores_2d

Species ordination plot

Description

plot_species_scores_2d creates an ordination diagram of species with selection criterion of which species to plot with names.

Usage

```

plot_species_scores_2d(
  species_scores,
  threshold = 7,
  speciesname = NULL,
  scorenames = c("RDA1", "RDA2"),
  selectname = "Fratio2",
  label.repel = FALSE,
  withnames_only = FALSE,
  max.overlaps = 10,
  mult_expand = 0.1,
  verbose = TRUE
)

```

Arguments

species_scores a species-by-scores matrix, a data frame with rownames (species names) or a tibble with variable with name speciesname containing species names and a column or variabe with names scorenames containing the scores (default: c("RDA1", "RDA2") being the species score names from [rda](#).

threshold	species with criterion (specified by selectname) higher than the threshold are displayed. Default: 5 (which is the threshold F-ratio for testing two regression coefficients (one for each axis) at $p=0.01$ with 60 df for the error in a multiple regression of each single species onto the condition and the ordination axes). If selectname is not in the data the threshold is divided by ten, so that the default is 0.5.
speciesname	name of the variable containing the species names (default NULL uses row-names).
scorenames	names of the two columns or variables containing the species scores to be plotted (default <code>c("RDA1", "RDA2")</code>).
selectname	name of the column or variable containing the criterion for the selection of species to be displayed Default: "Fratio2"; if selectname is not found in species_scores, set to the Euclidian norm (length) of the vectors defined by row-wise combining the values defined by scorenames.
label.repel	logical (default: FALSE) for using labels in white boxes with borders using geom_label_repel instead of using plain text labels using geom_text_repel .
withnames_only	logical to plot the species that exceed the threshold only, i.e. to <i>not</i> plot points indicating species that do not reach the threshold. (Default: FALSE).
max.overlaps	exclude text labels that overlap too many things. Default: 10 (see geom_text_repel).
mult_expand	fraction of range expansion of the diagram (default = 0.1). See expansion .
verbose	logical for printing the number of species that geom_text_repel or geom_label_repel will try to name in the plot out of the total number of species (default: TRUE).

Details

The names of species are added using [geom_text_repel](#) or [geom_label_repel](#). These functions may not be able to plot the names of all selected species. This issue can be diminished by increasing the value of `max.overlaps`.

Value

a ggplot object

See Also

[doPRC](#), [plotPRC2d](#)

Examples

```
data(pyrifos, package = "vegan") #transformed species data from package vegan
#the chlorpyrifos experiment from van den Brink & ter Braak 1999
Design <- data.frame(
  week=gl(11, 12, labels=c(-4, -1, 0.1, 1, 2, 4, 8, 12, 15, 19, 24)),
  dose=factor(rep(c(0.1, 0, 0, 0.9, 0, 44, 6, 0.1, 44, 0.9, 0, 6), 11)))

library(vegan)
mod_rda <- rda( pyrifos ~ dose:week + Condition(week), data = Design)
species_scores= scores(mod_rda,choices =1:2, display="sp",scaling = 1)
```

```

library(PRC)
head(rdacca_taxon_stats(mod_rda)) # stats per species
get_focal_and_conditioning_factors(mod_rda) # factors from the model

species_scores= vegan::scores(mod_rda,choices =1:2, display="sp",scaling = 2)
# 1-d vertical plot of species scores (a cut of RDA1-scores at 0.5)
plot_species_scores_bk(species_scores)

# a cut of RDA1-scores at 1.0 (as scoresname is unset)
plot_species_scores_bk(species_scores, threshold = 14)

plot_species_scores_2d(species_scores)
plot_species_scores_2d(species_scores, label.repel = TRUE)
plot_species_scores_2d(species_scores, max.overlaps = 30, label.repel = TRUE)

```

plot_species_scores_bk

Vertical ggplot2 line plot of ordination scores

Description

plot_species_scores_bk creates a vertical line plot of ordination scores with selection criterion for which scores to plot with names.

Usage

```

plot_species_scores_bk(
  species_scores,
  ylab = latex2exp::TeX("$b_k$"),
  threshold = 7,
  y_lab_interval = 0.5,
  speciesname = NULL,
  scoresname = "RDA1",
  selectname = "Fratio1",
  speciesgroup = NULL,
  expand = 0.2,
  verbose = TRUE
)

```

Arguments

species_scores a species-by-scores matrix, a data frame with row names (species names) or a tibble with variable with name speciesname containing species names and a column or variable with name scoresname containing the scores (default: "RDA1"), e.g. species scores from library vegan and an optional factor with name specified by the argument speciesgroup.

ylab	y-axis label. Default: <code>\$b_k\$</code> .
threshold	species with criterion (specified by <code>selectname</code>) higher than the threshold are displayed. Default: 7 (which is the threshold F-ratio for testing a single regression coefficient at $p = 0.01$ with 60 df for the error in a multiple regression of each single species onto the condition and the ordination axis). If <code>selectname</code> is not in <code>species_scores</code> , the threshold is divided by 14, so that the default is 0.5.
y_lab_interval	interval of the y-axis ticks. A tick at no effect (0) is always included; default: 0.5.
speciesname	name of the variable containing the species names (default NULL uses row names).
scoresname	name of the column or variable containing the species scores to be plotted (default "RDA1").
selectname	name of the column or variable containing the criterion for the selection of species to be displayed. Default: "Fratio1"; if <code>selectname</code> is not found in <code>species_scores</code> , set to <code>scoresname</code> .
speciesgroup	name of the factor, the levels of which receive different colors in the vertical plot.
expand	amount of extension of the line plot (default 0.2).
verbose	logical for printing the number of species with names out of the total number (default: TRUE).

Details

The absolute value of the criterion values is taken before selection, so that also the species scores of the ordination can serve as a criterion (e.g. `selectname = "RDA1"`). The function has been copied from the `douconca` package at <https://cran.r-project.org/web/packages/douconca>.

The function is used in `plotPRC` and `plotsmoothPRC`.

Value

a `ggplot` object

Examples

```
# Analyze example data -----

data(SimData)
# extract design
Design0 <- SimData[,c("A","B")]
Design0$A <- factor(Design0$A); Design0$B <- factor(Design0$B)
#put levels in natural order
Design0$A<- factor(Design0$A, levels=c(levels(Design0$A)[-2],levels(Design0$A)[2]))
Y0 <- as.matrix(SimData[,-(1:3)])
ids <- which(Design0$A %in% "A10" & Design0$B %in% "B5")
#design <- "Complete" #
design <- "Empty cell"
if (design == "Complete"){ Design <- Design0;Y <- Y0} else {Design <-Design0[-ids,]; Y=Y0[-ids,]}
print(with(Design,table(A,B)))
```

```

names(Design)

# Design and Y for Fig 1 of ter Braak 2023 ready -----
dim(Y)
str(Design)
names(Design) <- c("Time", "Treatment")
mod_prc <- doPRC(Y ~ Treatment:Time + Condition(Time), data = Design)
#Mod_prc <- vegan::prc(Y, Design$Treatment, Design$Time)
#Mod_prc$CCA$eig# identical to doPRC
b_mod_prc <- vegan::scores(mod_prc, choices = 1, display = "sp")

x_mod_prc <- vegan::scores(mod_prc, choices = 1, display = "lc")

summary(mod_prc)

smoothPRC_model <- set_smoothPRC_model(data = Design)
smoothPRC_model$spline
smooth_PRC <- smoothPRC(Y, model = smoothPRC_model)

summary(smooth_PRC$obj[[1]])
(cc <- cor(smooth_PRC$B[,1], b_mod_prc))
plot_species_scores_bk(-smooth_PRC$B, scoresname = "B1")
plot_species_scores_bk(b_mod_prc)
plot_smoothPRC_cdt(smooth_PRC, flip = TRUE)

plotsmoothPRC(smooth_PRC, flip = TRUE, threshold = 0)

# Demonstration of the fitted model
x1 <- scale(as.numeric(Design$Time))[,1]
x2 <- scale(as.numeric(Design$Treatment))[,1]

# with the scaling to mean square of 1 of the species scores b
summary(lm(smooth_PRC$X[,1] ~ x1 + x2 + x1:x2)) # R^2 ~1!

plotPRC(mod_prc)

```

plotPRC

1d PRC diagram

Description

plotPRC creates a one-dimensional PRC diagram with vertical line plot of species loadings from a result of [doPRC](#) or [PRC_scores](#).

Usage

```
plotPRC(
  object,
  treatment = NULL,
  condition = NULL,
  plot = 1,
  xvals = NULL,
  axis = 1,
  size = c(2, 2),
  symbols_on_curves = FALSE,
  widths = c(4, 1),
  title = NULL,
  left = "Treatment and plot scores",
  right = "taxon scores",
  threshold = 7,
  y_lab_interval = 0.5,
  speciesname = NULL,
  selectname = "Fratio",
  verbose = TRUE
)
```

Arguments

object	a result of doPRC or PRC_scores or a list with element PRCplus (a data frame with a variables to plot, see details).
treatment	character vector of names of factors in object\$PRCplus, the first two of which set the color and linetype of treatments. Default: derived from object\$'focal_and_conditioning_f
condition	character vector of names of factors in object\$PRCplus, the first one being used as x-axis. Default: derived from object\$'focal_and_conditioning_factors'.
plot	0 or 1 (no/yes individual plot points) or character name of a factor in object\$PRCplus indicating the plots (microcosms, subjects or objects) in a longitudinal study so as to plot lines. Default: 1 yielding points instead of lines.
xvals	numeric values for the levels of the condition factor, or name of numeric variable in object\$PRCplus to be used for the x-axis. Default: NULL for which the values are obtained using the function fvalues4levels applied to object\$PRCplus[[condition[1]]]
axis	axis shown; default 1, showing the first PRC axis.
size	size of symbols and symbols in the legend. default: c(2, 2). If the length is not 2 than the first entry is replicated.
symbols_on_curves	logical, draw symbols on the PRC curves at the measurement times (default FALSE). If TRUE, symbols are for the treatment, if there is one, and for the second treatment if there are two.
widths	relative width of treatment PRC plot and the loading plot (see grid.arrange).
title	overall title.
left	left axis text (see arrangeGrob).

right	right axis text (see arrangeGrob).
threshold	species with criterion (specified by selectname) higher than the threshold are displayed. Default: 7 (which is the threshold F-ratio for testing a single regression coefficient at $p = 0.01$ with 60 df for the error in a multiple regression of each single species onto the condition and the ordination axis). If selectname is not in species_scores, the threshold is divided by 14, so that the default is 0.5.
y_lab_interval	interval of the y-axis ticks. A tick at no effect (0) is always included; default: 0.5.
speciesname	name of the variable containing the species names (default NULL uses rownames in object\$species)
selectname	name of the column or variable containing the criterion for the selection of species to be displayed without "1" or other axis number. Default: "Fratio"; if selectname is not found in object\$species_scores, set to scoresname. e.g. the "RDA1" scores.
verbose	logical for plotting and printing the current model, score_name, condition and treatment (default TRUE).

Details

The function creates the treatment PRC diagram using [plot_sample_scores_cdt](#) and the vertical line plot of loadings using [plot_species_scores_bk](#). These ggplot2-objects are combined in a single object using [arrangeGrob](#).

Value

a list of a gridExtra object (plot using [arrangeGrob](#)) and a list of two ggplot2 objects (the treatment PRC-diagram and the vertical line plot).

See Also

[doPRC](#), [plotPRC2d](#), [plot_sample_scores_cdt](#), [plot_species_scores_bk](#) and [grid.arrange](#)

Examples

```
library(PRC)
data(pyrifos, package = "vegan") #log-transformed species data from package vegan
#the chlorpyrifos experiment from van den Brink & ter Braak 1999
Design <- data.frame(week=gl(11, 12, labels=c(-4, -1, 0.1, 1, 2, 4, 8, 12, 15, 19, 24)),
                     dose=factor(rep(c(0.1, 0, 0, 0.9, 0, 44, 6, 0.1, 44, 0.9, 0, 6), 11)),
                     ditch = gl(12, 1, length=132))

#

mod_prc <- doPRC(pyrifos ~ dose:week + Condition(week), data = Design)
mod_prc$focal_and_conditioning_factors$focal factor` # as expected: "dose"
print(mod_prc)

#               Proporti Rank
# Total          1.0000
```

```

# Conditional      0.2192   10 (between-week variation)
# Constrained      0.3346   44 (between-treatment variation across weeks)
# Unconstrained    0.4462   77 (within-week variation)

# without lines, threshold is 7
plotPRC(mod_prc)
# this threshold is about P = 0.01 for PRC1 in the regression "species_k ~ week+PRC1"
# and retains 34 species to be plotted with names
sum(mod_prc$species[, "Fratio1"] > 7) # 34
# only the PRC curves:
plotPRC(mod_prc, plot = 0, symbols_on_curves = TRUE)
# with lines and a stronger threshold
plotPRC(mod_prc, plot = "ditch", threshold = 10, width = c(4,1))

# modifying the plot
gg <- plotPRC(mod_prc, plot = "ditch", width = c(4,1), verbose = FALSE)
# PRC plot of samples (c_dt)
p1 <- gg$separateplots$treatments +
  ggplot2::ggtitle(paste("new title:", latex2exp::TeX("$c_{dt}$")))

p2 <- gg$separateplots$species +
  ggplot2::ylab("new title: loadings") # loadings of species (b_k)
# Assign these plots to symbols and use grid.arrange to produce the plot
# you like, for example:
gridExtra::grid.arrange(p1+ ggplot2::ylab("")+
  ggplot2::xlab("week since chorpyrifos application") + ggplot2::ggtitle(""),
  p2, ncol = 2, widths = c(4,1),
  top = "new title", left = "PRC curves of treatments",
  right = "PRC species loadings")

# test significance of PRC/RDA axes -----

## Ditches are randomized, we have a time series, and are only
## interested in the first axis
library(vegan)
ctrl <- how(plots = Plots(strata = Design$ditch, type = "free"),
  within = Within(type = "none"), nperm = 99)

anova(mod_prc, by = "axis", permutations = ctrl, model = "reduced",
  cutoff = 0.10)

```

plotPRC2d

2d PRC diagram

Description

plotPRC2d creates a two-dimensional PRC diagram with species loadings from a result of [doPRC](#) or [PRC_scores](#)

Usage

```
plotPRC2d(
  object,
  treatment = NULL,
  condition = NULL,
  plot = 1,
  xvals = NULL,
  axes = c(1, 2),
  threshold = 7,
  title = NULL,
  left = NULL,
  right = NULL,
  speciesname = NULL,
  selectname = "Fratio",
  xylab = NULL,
  label.repel = FALSE,
  withnames_only = FALSE,
  max.overlaps = 10,
  mult_expand = 0.1,
  widths = c(3, 2),
  verbose = TRUE
)
```

Arguments

object	a result of doPRC or PRC_scores or a list with element PRCplus (a data frame with a variables to plot, see details).
treatment	character vector of names of factors in object\$PRCplus, the first two of which set the color and linetype of treatments. Default: derived from object\$'focal_and_conditioning_f
condition	character vector of names of factors in object\$PRCplus, the first one being used as x-axis. Default: derived from object\$'focal_and_conditioning_factors'.
plot	0 or 1 (no/yes individual plot points) or character name of a factor in object\$PRCplus indicating the plots (microcosms, subjects or objects) in a longitudinal study so as to plot lines. Default: 1 yielding points instead of lines.
xvals	numeric values for the levels of the condition factor, or name of numeric variable in object\$PRCplus to be used for the x-axis. Default: NULL for which the values are obtained using the function fvalues4levels applied to object\$PRCplus[[condition[1]]]
axes	axes shown; default c(1, 2) showing the first two PRC axes.
threshold	species with criterion (specified by selectname) higher than the threshold are displayed. Default: 5 (which is the threshold F-ratio for testing two regression coefficients (one for each axis) at $p=0.01$ with 60 df for the error in a multiple regression of each single species onto the condition and the ordination axes). If selectname is not in the data the threshold is divided by ten, so that the default is 0.5.
title	overall title.
left	left axis text (see arrangeGrob).

<code>right</code>	right axis text (see arrangeGrob).
<code>speciesname</code>	name of the variable containing the species names (default NULL uses row-names).
<code>selectname</code>	name of the column or variable containing the criterion for the selection of species to be displayed Default: "Fratio"; if <code>selectname</code> is not found in <code>object\$species_scores</code> , set to the Euclidian norm (length) of the vectors defined by row-wise combining the values defined by <code>scorenames</code> .
<code>xylab</code>	character vector of length 2 defining x- and y-axis title of the species ordination so as to overwrite the defaults axis labels
<code>label.repel</code>	logical (default: FALSE) for using labels in white boxes with borders using geom_label_repel instead of using plain text labels using geom_text_repel .
<code>withnames_only</code>	logical to plot the species that exceed the threshold only, i.e. to <i>not</i> plot points indicating species that do not reach the threshold. (Default: FALSE).
<code>max.overlaps</code>	exclude text labels that overlap too many things. Default: 10 (see geom_text_repel).
<code>mult_expand</code>	fraction of range expansion of the diagram (default = 0.1). See expansion .
<code>widths</code>	relative width of treatment PRC plot and the loading plot (see grid.arrange).
<code>verbose</code>	logical for printing the number of species that geom_text_repel or geom_label_repel will try to name in the plot out of the total number of species (default: TRUE).

Details

The function creates the treatment PRC diagrams for the axes using [plot_sample_scores_cdt](#), with diagrams for the $x=y$ and $x=-y$ directions, and the 2d species loading plot using [plot_species_scores_2d](#). These ggplot2-objects are combined in a single object using [arrangeGrob](#).

Value

a list of a `gridExtra` object (plot using [arrangeGrob](#)) and a list of five ggplot2 objects, arranged in a list of two: `treatments` = the PRC1, PRC2-diagrams with line $x=y$ and $x=-y$ diagrams and `species` = the 2d species ordination plot).

See Also

[doPRC](#), [plot_sample_scores_cdt](#), [plot_species_scores_2d](#) and [grid.arrange](#)

Examples

```
library(PRC)
data("CPFNUTphytoplankton")

# Read example data -----

names(CPFNUTphytoplankton)[1:6]
Design <- CPFNUTphytoplankton[,c("Time", "Treatment", "Plot", "con_nut_cpf")]
summary(Design)
print(with(Design, table(Time, Treatment)))
```

```

# extract response variables
Counts <- as.matrix(CPFNUTphytoplankton[,-(1:5)]); rownames(Counts)<-CPFNUTphytoplankton$sample
# transform as a multiplicative model is more logical than an additive one.
# But need to add pseudocount to avoid log(0) = infinite

Y <- log(Counts+1)
# Remark: this transformation reproduces the results in
#         van den Brink & ter Braak 1998
#         apart from scaling (but the product b_k*c_dt is identical) and
#         apart from the PRC1 score at week 4, which is erroneous in the paper.
# PRC via vegan -----

mod_prc <- doPRC( Y~ Time * Treatment + Condition(Time), rank = 3,
                  data= Design, flip= c(FALSE, TRUE, FALSE))
round(mod_prc$percExp, 0)

plotPRC(mod_prc, plot = "Plot", width = c(2,1), symbols_on_curves = TRUE)

plotPRC(mod_prc, plot = "Plot", width = c(2,1), axis = 2)
# note that the default selection criterion "Fratio2" also selects species
#strongly reacting to axis 1

plotPRC2d(mod_prc, plot = 0, threshold = 1,
           title = "PRC Phytoplankton van den Brink & ter Braak (1998)")

# modifying the plot
# Assign the plots to symbols and use grid.arrange to produce the plot you like,
#for example:
gg <- plotPRC2d(mod_prc, plot = 0, verbose = FALSE)
pl_species <- gg$separateplots$species # species ordination
pl <- gg$separateplots$treatments
names(pl)

layout<- rbind(c(1,2), c(3,4),c(3,5))
gridExtra::grid.arrange(pl[["PRC1"]]+ggplot2::ylab("PRC1")+ggplot2::ggtitle(""),
                        pl[["PRC2"]]+ggplot2::ylab("PRC2")+ggplot2::xlab("")+
                        ggplot2::theme(legend.position="none")+ggplot2::ggtitle(""),
                        pl_species,
                        pl[["line1"]]+ggplot2::ylab("scores x = y")+
                        ggplot2::xlab("")+ggplot2::theme(legend.position="none")+
                        ggplot2::ggtitle(""),
                        pl[["line1-1"]]+ggplot2::ylab("scores x = -y")+
                        ggplot2::theme(legend.position="none")+
                        ggplot2::ggtitle(""),
                        layout_matrix = layout, widths = c(3,2),
                        top = "mytitle", left = "", right = "")

```

plotsmoothPRC	<i>smooth PRC diagram</i>
---------------	---------------------------

Description

Creates a smooth PRC diagram with treatment curves and a vertical species-loading plot.

Usage

```
plotsmoothPRC(
  object,
  axis = 1,
  mod_prc = NULL,
  with_classical_lines = TRUE,
  with_unconstrained_scores = TRUE,
  flip = FALSE,
  widths = c(4, 1),
  title = NULL,
  left = "Treatment curves",
  right = "taxon scores",
  threshold = 7,
  y_lab_interval = 0.5,
  speciesname = NULL,
  selectname = "Fratio",
  sample_times_only = FALSE,
  verbose = TRUE
)
```

Arguments

object	a result of smoothPRC .
axis	the number of the axis to plot
mod_prc	a results of doPRC , <i>i.e.</i> a classical PRC with the same data and design. Default NULL, in which the classical PRC is computed using doPRC and the data in object.
with_classical_lines	logical to add or suppress classical PRC lines. Default TRUE, <i>i.e.</i> with classical PRC lines.
with_unconstrained_scores	logical to add or suppress the unconstrained scores in the smooth PRC showing the scatter of the individual samples (cosms). Default TRUE, <i>i.e.</i> with individual points.
flip	logical. Should the axis be reversed? Default FALSE. Can be numeric with -1 meaning: reverse, and 1 meaning do not reverse.
widths	relative width of treatment PRC plot and the loading plot (see grid.arrange).

title	overall title.
left	left axis text (see arrangeGrob).
right	right axis text (see arrangeGrob).
threshold	species with criterion (specified by selectname) higher than the threshold are displayed. Default: 7 (which is the threshold F-ratio for testing a single regression coefficient at $p = 0.01$ with 60 df for the error in a multiple regression of each single species onto the condition and the ordination axis). If selectname is not in species_scores, the threshold is divided by 14, so that the default is 0.5.
y_lab_interval	interval of the y-axis ticks. A tick at no effect (0) is always included; default: 0.5.
speciesname	name of the variable containing the species names (default NULL uses row names).
selectname	name of the column or variable containing the criterion for the selection of species to be displayed. Default: "Fratio1"; if selectname is not found in species_scores, set to scoresname.
sample_times_only	logical to PRClines based on the sampled times only, <i>i.e.</i> without interpolation. Default FALSE.
verbose	logical for printing the number of species with names out of the total number (default: TRUE).

Details

sample_times_only = TRUE can be useful for comparison with classical PRC. It avoids interpolation and may work when the default fails to produce a plot.

PRC_scores	<i>Calculate principal response curve (PRC) scores from vegan::rda, cca, dbrda or capscale objects</i>
------------	--

Description

PRC_scores calculates scores to display principal response curves with scores for individual plots from a vegan constrained ordination object ([cca.object](#)).

Usage

```
PRC_scores(
  object,
  focal_factor_name = NULL,
  referencelevel = NULL,
  rank = 2,
  flip = rep(FALSE, rank),
  scaling = "ms",
  data
)
```

Arguments

<code>object</code>	a vegan cca.object , a result of rda , cca , dbrda or capscale with model specified via a formula.
<code>focal_factor_name</code>	name of one or two focal factors (the treatment(s) of interest); if unspecified, derived from the formula in <code>object</code> via the function get_focal_and_conditioning_factors .
<code>referencelevel</code>	numeric or character level(s) to be used as reference of the <code>focal_factor_name(s)</code> ; default: 1.
<code>rank</code>	number of axes to be computed; default 2.
<code>flip</code>	logical value or vector; default: FALSE. Should the axes be flipped, i.e. reversed in orientation? <code>flip</code> can be numeric with values -1 and 1 for reverse and no reverse, respectively, i.e. multiply the scores by either -1 or 1.
<code>scaling</code>	character <code>c("ms", "ss")</code> . Scales the axes of an RDA object to mean square species scores of 1 ("ms") or a sum of squares of 1 ("ss"). The only scaling for a CCA object is "ss" (weighted mean square = 1, with weight = species total abundance). For a dbrda object, there are no species scores and the scaling is like "ss". The default scaling ("ms" for RDA and "ss" for CCA) has the advantage that the PRC treatment and sample scores measure the effect sizes of the 'average' species as their (weighed) mean square is 1 (sd = 1); it is the scaling used in Canoco5 (www.canoco5.com). Scaling "ss" is the default in prcomp and used in ter Braak (2023).
<code>data</code>	data frame of the experimental design used in the call to rda or cca

Details

PRC_scores uses [scores](#) to calculate the loadings (species scores), if obtainable from the object, and the constrained and unconstrained sample scores of the units in data. The PRC treatment and sample scores are obtained by subtracting the reference scores from the constrained and unconstrained scores, respectively. The reference scores are calculated by applying [predict](#) with `type = "lc"` on `newdata`, which equals `data`, except that the level of the focal factor of each unit is changed to the reference level. By this simple method, prc scores can be applied with and without terms in `Condition()` in the formula of the call to [rda](#), [cca](#), [dbrda](#) or [capscale](#). There can thus be additional (perhaps quantitative) covariates, `A:B+ Condition(A+other_covariates)` where the focal factor is B, and also with three (instead of two factors) e.g. with formula `A*B*C + Condition(A*C)` so as to display the A*C-dependent effects of focal factor B on the multivariate response in the call to [rda](#), [cca](#), [dbrda](#) or [capscale](#). The element `object$'focal_and_conditioning_factors'` in value is set by applying the function [get_focal_and_conditioning_factors](#) to the ordination model specified in `object`.

CCA-based PRC is useful in data with many zeroes and which are strictly compositional, i.e. have row sums that have no meaning for the question of interest (ter Braak and te Beest, 2022). A layman's example is that the weight of a cake is not important for its taste; taste depends on its composition only. CCA can be applied without log-transforming the data and thereby avoids the issue of replacement of zeroes (ter Braak and te Beest, 2022).

Value

A named list:

- **PRCplus**: a data frame that combines the design in data with the PRC and RDA/CCA scores with (`_E`) and without error (the unconstrained and constrained scores (sites and LC), respectively), and, finally, the reference scores. Note that $PRC1_E = RDA1_E - RDA1_Ref$, etc for other axes
- **species**: a matrix with the loadings with respect to the PRC scores; NULL if the object lacks loadings.
- **reference_scores**: a matrix with the scores of each unit in data with the level of the focal factor replaced by the reference level
- **focal_factor_name**: name of the focal factor in the call to `PRC_scores`
- **referencelevel**: reference level in the call to `PRC_scores`
- **coefficients**: list of PRC-coefficients for each axis (matrix if rank is equal to 1)
- **focal_and_conditioning_factors**: list of focal and conditioning factors; obtained from `get_focal_and_conditioning_` if argument `focal_factor_name` in the call is unset.
- **terms**: `object$terms`. See [cca.object](#).
- **terminfo**: `object$terminfo`. See [cca.object](#).
- **method**: `object$method`. See [cca.object](#).
- **percExp**: percentage variance of each axis out of the variance that can be explained by the predictors and covariates.

See Also

[doPRC](#), [plotPRC](#), [plotPRC2d](#)

Examples

```
data(pyrifos, package = "vegan") #transformed species data from package vegan
#the chlorpyrifos experiment from van den Brink & ter Braak 1999
Design <- data.frame(week=gl(11, 12, labels=c(-4, -1, 0.1, 1, 2, 4, 8, 12, 15, 19, 24)),
                    dose=factor(rep(c(0.1, 0, 0, 0.9, 0, 44, 6, 0.1, 44, 0.9, 0, 6), 11)),
                    ditch = gl(12, 1, length=132))

mod_rda <- vegan::rda( pyrifos ~ dose:week + Condition(week), data = Design)
Design_w_PRCs <- PRC_scores(mod_rda, focal_factor_name= "dose", referencelevel = "0",
                           rank = 2, data= Design, scaling ="ms", flip = FALSE)

plot_sample_scores_cdt(Design_w_PRCs, plot = "ditch")
plot_sample_scores_cdt(Design_w_PRCs)
plot_sample_scores_cdt(Design_w_PRCs, plot = "Ditch")
```

rdacca_taxon_stats	<i>Taxon fit statistics of an rda or cca ordination</i>
--------------------	---

Description

rdacca_taxon_stats calculates fit statistics for taxa in an [rda](#) or [cca](#) ordination.

Usage

```
rdacca_taxon_stats(object, rank = 1)
```

Arguments

object	a result of rda or cca .
rank	number of axes for which fits are calculated (1:rank). Default: 1.

Details

The code uses [ordiYbar](#) and [scores](#) and also works for unconstrained ordination using [rda](#) and [cca](#). The software Canoco 5 (ter Braak & Šmilauer, 2018) contains Cfit and a rescaled version of the column Var but not the columns Fratio.k.

Value

A matrix with taxa in rows and taxon specific statistics in columns:

Var sample variance of the taxon.

FitCE R squared: fraction of the sample variance explained by the predictors and covariates.

FitE fraction of the sample variance explained by the predictors after adjustment for the covariates.

Cfitk fraction of the sample variance explained by the first k axes (the axes are constrained in a constrained ordination and unconstrained otherwise).

Fratiok F-ratio of the regression model with k axes (as predictors) and the model without axes (i.e. with covariates or the intercept only).

References

ter Braak C.J.F. & Šmilauer P. (2018): Canoco reference manual and user's guide: software for ordination, version 5.1x. Microcomputer Power, Ithaca, USA, 536 pp. (<http://www.canoco5.com>)

Examples

```
data(pyrifos, package = "vegan") #transformed species data from package vegan
#the chlorpyrifos experiment from van den Brink & ter Braak 1999
Design <- data.frame(
  week=gl(11, 12, labels=c(-4, -1, 0.1, 1, 2, 4, 8, 12, 15, 19, 24)),
  dose=factor(rep(c(0.1, 0, 0, 0.9, 0, 44, 6, 0.1, 44, 0.9, 0, 6), 11)))
```

```

library(vegan)
mod_rda <- rda( pyrifos ~ dose:week + Condition(week), data = Design)
species_scores= scores(mod_rda,choices =1:2, display="sp",scaling = 1)
library(PRC)
head(rdacca_taxon_stats(mod_rda)) # stats per species
get_focal_and_conditioning_factors(mod_rda) # factors from the model

species_scores= vegan::scores(mod_rda,choices =1:2, display="sp",scaling = 2)
# 1-d vertical plot of species scores (a cut of RDA1-scores at 0.5)
plot_species_scores_bk(species_scores)

# a cut of RDA1-scores at 1.0 (as scoresname is unset)
plot_species_scores_bk(species_scores, threshold = 14)

plot_species_scores_2d(species_scores)
plot_species_scores_2d(species_scores, label.repel = TRUE)
plot_species_scores_2d(species_scores, max.overlaps = 30, label.repel = TRUE)

```

set_smoothPRC_model	<i>Define the model for smooth PRC using LMMsolver (Boer 2023) and smoothPRC</i>
---------------------	--

Description

Define the model for smooth PRC using LMMsolver (Boer 2023) and smoothPRC

Usage

```

set_smoothPRC_model(
  spline = NULL,
  splineH0 = ~spl1D(time, nseg = 6, degree = 3, pord = 2),
  fixed = y ~ 1,
  fixedH0 = fixed,
  random = NULL,
  residual = NULL,
  weights = NULL,
  randomH0 = random,
  treatment.coding = c("quantitative", "qualitative"),
  pord = c(time = 2, dose = 2, Treatment = 2),
  nseg = c(time = 6, dose = 6),
  degree = 3,
  start_time = 0,
  referencelevel = NULL,
  method = "rda",
  scaling = "ms",
  data
)

```

Arguments

spline	A formula for the spline part of the model. Should be of the form "<code>~ spl1D()</code>", "<code>~ spl2D()</code>" or "<code>~ spl3D()</code>". Generalized Additive Models (GAMs) can also be used, for example "<code>~ spl1D() + spl2D()</code>". Default: <code>NULL</code>, resulting in "<code>~ spl2D(time, dose, nseg = c(6, 6), degree = 3, pord = 2)</code>" when <code>treatment.coding</code> is <code>TRUE</code> and in, <i>e.g.</i> <pre>spl1D(time, nseg = 6, degree = 3, pord = 2) + spl1D(D1, nseg = 6, degree = 3, pord = 2) + spl1D(D2, nseg = 6, degree = 3, pord = 2) + spl1D(D3, nseg = 6, degree = 3, pord = 2)</pre> with <code>treatment.coding</code> is <code>FALSE</code>, and the factor <code>Treatment</code> has four levels. When both <code>spline</code> and <code>splineH0</code> are <code>NULL</code>, the resulting formula is <code>NULL</code>, for both.
splineH0	As <code>spline</code> but for the null model. Default: a one dimensional smooth in time: <code>~ spl1D(time, nseg = 6, degree = 3, pord = 2)</code>
fixed	Formula for the fixed part of the full model. Default <code>y~1</code> . NB: the response variable needs to be <code>y</code> , a variable internally generated from the abundance table <code>Y</code> and the loading (species) vector, that is optimized. See LMMsolve .
fixedH0	Formula for the fixed part of the null model. The default is equal to <code>fixed</code> , <i>i.e.</i> <code>y~1</code> .
random	Formula of the form " <code>~ pred</code> " for the random term in the full model fit. The default: <code>NULL</code> for no random term in the model.
residual	A formula for the residual part of the model. Should be of the form " <code>~ pred</code> ".
weights	A character string identifying the column of data to use as relative weights in the fit. Default value <code>NULL</code> , weights are all equal to one.
randomH0	The default is equal to <code>random</code> . Random term in the null model fit. Can be <code>NULL</code> for no random term in the null model.
treatment.coding	character: "quantitative" or "qualitative". Default "quantitative" for a quantitative analysis using a 2d spline in time and dose. Set <code>treatment.coding = "qualitative"</code> for a qualitative analysis in which the default spline model is replaced by one containing 1d splines, one 1d-spline for each level of the factor <code>Treatment</code> .
pord	named integer vector of the difference order of the spline penalty. Default: <code>c(time = 2, dose = 2)</code> . No effect if <code>spline</code> is not <code>NULL</code> .
nseg	named integer vector of number of segments in P-splines, can be large. Default <code>c(time = 6, dose = 6)</code> . No effect if <code>spline</code> is not <code>NULL</code> .
degree	degree of the B-splines. Default 3. No effect if <code>spline</code> is not <code>NULL</code> .
start_time	numeric time of the start of the treatment application. Default: 0.
referencelevel	name of the reference treatment. Default: first level of <code>factor(data\$Treatment)</code> .
method	character <code>rda</code> or <code>cca</code> . Default: " <code>rda</code> "
scaling	The default gives mean square species scores of 1, otherwise the sum of squares of the species scores is 1. See doPRC . Default: " <code>ms</code> ".
data	A <code>data.frame</code> containing the modeling data.

Details

An atypical aspect of the function is that the time and treatment factors need to have the names Time and Treatment in data, *i.e.* set_smoothPRC_model assumes that the data contain two factor or character variables Time and Treatment that can be converted to the numeric variables time and dose, respectively, using [fvalues4levels](#). The values of dose are set to 0 for negative values of time and time == 0, after subtraction of start_time. In this case, the default spline model is: ~ spl2D(time, dose, seg = c(6, 6), degree = 3, pord = 2). To enable Treatment to be treated as a factor (instead of as a quantitative variable), the function generates variables with names D1, ..., Dk, with k the number of treatments beyond the reference treatment, if treatment.coding is FALSE. These variables generated in the function, contain the time since the start of the experiment for each treatment. D1, D2, ... should therefore not be variable names in data, nor should time or dose be variable names in data; these are reserved names in smoothPRC. An example of the formula for spline when treatment.coding is FALSE and the number of levels of Treatment is 4, is

```
spl1D(time, nseg = 6, degree = 3, pord = 2) +
spl1D(D1,   nseg = 6, degree = 3, pord = 2) +
spl1D(D2,   nseg = 6, degree = 3, pord = 2) +
spl1D(D3,   nseg = 6, degree = 3, pord = 2)
```

This is the default model if treatment.coding is FALSE. The default spline model assumes that the Treatment factor has a quantitative basis, so that the response to Time and Treatment can be analyzed using a 2D spline model.

References

Boer, Martin P. 2023. Tensor Product P-Splines Using a Sparse Mixed Model Formulation. Statistical Modelling 23 (5-6): 465–79. doi:[10.1177/1471082X231178591](https://doi.org/10.1177/1471082X231178591)

See Also

[smoothPRC](#)

Examples

```
data("Ossenkampen")
names(Ossenkampen)[4:5]
names(Ossenkampen)[4:5] <- c("Time", "Treatment")
test <- FALSE
if (test){
  years <- sort(unique(Ossenkampen$Time))
  years<- years[c(-(1:9))]
  years <- years[-which(years%in% c(1984))]
  years
  ids <- which(Ossenkampen$Time %in% years)
}else ids <- 1:nrow(Ossenkampen)

Y <- Ossenkampen[ids,-(1:5)]
Design <- Ossenkampen[ids,1:5]
Design$Block <- factor(Design$Block)
```

```

# in doPRC and smoothPRC Time must be a factor, not a quantitative variable
Design$Time <- factor(Design$Time)
with(Design, table(Time, Treatment))
# unbalanced!!! if test <- FALSE

mod_prc <- doPRC(Y ~ Time:Treatment + Condition(Time+Block), data = Design)

b_mod_prc <- vegan::scores(mod_prc, choices = 1, display = "sp")

smoothPRC_model <-
  set_smoothPRC_model(data = Design,
    treatment.coding = "quali", start_time = 1958)

smoothPRC_model$fixed
smoothPRC_model$spline

smooth_PRC <- smoothPRC(Y, model = smoothPRC_model, formula_covariates = ~Block)
smooth_PRC$iter
if (1==0){
  # start the iterations from previous
  opt <- smooth_PRC$options_iter
  opt$init <- smooth_PRC$X
  smooth_PRC2 <- smoothPRC(Y, model = smoothPRC_model, formula_covariates = ~Block,
    options_iter = opt)
  smooth_PRC2$iter
  round(diag(cor(smooth_PRC$X, smooth_PRC2$X)), 6)
  round(diag(cor(smooth_PRC$B[, 1:2], smooth_PRC2$B[, 1:2])), 6)
}
summary(smooth_PRC$obj[[1]])
cor(smooth_PRC$B[, 1], b_mod_prc)

plotsmoothPRC(smooth_PRC, flip = TRUE)

```

SimData

Simulated data from Fig. 1 of ter Braak (2023)

Description

Simulated data with a 10x5 factorial design with two replications. The true PRC curves are diverging straight lines as given in FigS2 of ter Braak (2023) with an interaction effect that is half the main effect of the second factor. The data and design as analysed in ter Braak (2023) has one empty cell (cell A10, B5).

The data has 100 unit and 10 columns as follows:

- case identifier for a sample e.g. A2B3.1 the replicate of cell for the levels 2 and 3 of the factors A and B, respectively.
- A factor A, taken as Time variable in the classical PRC of ter Braak (2023)

- B factor B, taken as Treatment variable in the classical PRC of ter Braak (2023)
- V1 value of response variable 1 and so on for the remaining 6 response variables

References

ter Braak, C.J.F. (2023) Redundancy analysis includes analysis of variance-simultaneous component analysis (ASCA) and outperforms its extensions. *Chemometrics and Intelligent Laboratory Systems*, 104898. doi:[10.1016/10.1016/j.chemolab.2023.104898](https://doi.org/10.1016/j.chemolab.2023.104898)

smoothPRC	<i>Smooth PRC using LMMsolver (Boer, 2023)</i>
-----------	--

Description

Smooth PRC using LMMsolver (Boer, 2023)

Usage

```
smoothPRC(
  response,
  model,
  options_iter = list(init = NULL, k = 2, mA = 4, tol = 1e-08, maxiter = 50),
  n_axes = 2,
  formula_covariates = NULL,
  weights = NULL,
  n_species = 1
)
```

Arguments

response	matrix or data frame of the abundance data or a distance matrix or object of class <code>dist</code> The abundance data has dimension <i>sites</i> x <i>species</i> .
model	a named list. Usually the result of <code>set_smoothPRC_model</code> The model contains at least one element and one or more optional elements used for fitting the model using the functions <code>fit_full_model</code> and <code>fit_null_model</code> (see Details): • method: character. Default: "dbrda" for distance-based smooth PRC, when response is a distance-like object. The default is "rda" for rda-based smooth PRC, when response is an abundance matrix, leading to models of the principal components family (pca, princomp, rda, dc-pca). Any other string then leads to CCA-based smooth PRC. • data: an object, usually a data frame with the predictor data for rows (sites) • formula: one or more optional formulas specifying the model of the predictor data (if any) to be fitted to a response vector.
options_iter	list of five elements

	• <code>init</code>: initial values of sample scores. It must be a vector of length the number of samples or a matrix with axis-specific initial values in columns. If response is an abundance matrix, initialization can be via species loadings. Default NULL, resulting in a start for each eigenvector from the model $\text{Treatment} \times \text{Time} + \text{Condition}(\text{Time})$ with (if any) the current covariables and (smooth) axes that are already extracted. • <code>k</code>: number of cycles in block-power updating to better separate the first eigen vector from the second. Default 2. • <code>mA</code>: number of vectors in Anderson acceleration. Default 4. • <code>tol</code>: stopping criterion, tolerance. Default $1e-4$. • <code>maxiter</code>: maximum number of iterations. Default 20.
<code>n_axes</code>	the number of axes to extract. Default 2.
<code>formula_covariates</code>	one-sided formula with row covariates (sample-specific predictors) that the model needs to adjust for. Default NULL, resulting in ~ 1 . The formula specifies what is called the Condition in <i>vegan</i> , covariates in <i>Canoco</i> and covariables in <i>ter Braak & Prentice</i> (1988). The covariates must be in <code>model\$data</code> .
<code>weights</code>	a list of user-defined weights for columns (species) first and, second, rows (sites) Default NULL for method-defined weights, e.g. row totals of response in CCA, and uniform weights in RDA. Zero weights are not allowed; near zero weights may lead to instability; this is unchecked currently. Ignored when response is a distance.
<code>n_species</code>	number of species in response. Default 1. Ignored if response is an abundance matrix. This parameter is included for testing purposes, in particular, to get same scaling irrespective when response is an abundance matrix or a 'euclidean' distance matrix of that abundance matrix.

Details

An atypical aspect of the function is that the time and treatment factors need to have the names `Time` and `Treatment` in data.

Classical PRC is a reduced rank model of rank 1 (or 2) with model $\sim \text{Treatment} \times \text{Time}$ with the `Time` factor is a concomitant variable. In *smoothPRC* the factor `Time` is taken quantitatively and the main effect of time is not an un-ordered set of values but a smooth P-spline model. Also, the time-dependent effects of the treatments are smooth P-splines. As the main and interaction effects are splines, the main effect of time cannot be adjusted for in the usual way. It is taken out via a separate spline fit using the formulas in `splineH0` and `fixedH0` of [set_smoothPRC_model](#).

The algorithm solves a penalized reduced rank model for the dominant eigen vector (*i.e.* rank 1). The model is a P-spline model and the penalty parameters are obtained by REML using [LMMsolve](#) (Boer 2023). The model is linear in, possibly tensor, B-splines, but through the B-splines non-linear in the original quantitative variables, here `time` or `dose` and thus non-linear from the perspective of the user. With quadratic ridge-type difference penalties on the coefficients, the B-splines become P-splines (penalized B-splines). For an initial vector `b` of species scores, the unconstrained sites are calculated, which are then used as response vector in [LMMsolve](#) to give a constrained site scores `x` by the P-spline model specified in `model`. This is part of a power algorithm with block and Anderson acceleration to solve the dominant eigen vector of the underlying eigen problem while optimizing simultaneously the penalties using [LMMsolve](#).

The smoothPRC algorithm uses one and two-sided power iterations. It thus iterates through the transition formulas of the method with (P-spline) regression fits. It uses two acceleration methods: blocking and Anderson iteration.

The response is taken as a distance object if it inherits class `dist` or if it is a matrix-like object with equal dimensions using `dim` and all zero diagonal using `sum(diag(response))==0`. Initialization of the power algorithm expands the `dist` object in response to a square symmetric matrix. This expansion can be avoided by explicit initialization using `options_iter$init`.

If response is a distance object and the maximum distance exceeds 4 then the distance is divided by $n-1$ with n the number of samples. This choice is taken from `vegan::dbrda` and usually makes the eigenvalues equal to those of `vegan::rda` if the distance is 'euclidean'.

Value

A list.

eig eigenvalues.

B matrix with response variable (species) scores B 's with `Fratio`'s with attached axis number, e.g. `B1`.

X matrix with the eigen vectors (i.e. the constrained sample scores) in columns.

X_star matrix with the unconstrained sample scores.

PRC matrix with the PRC scores, X scores compared to the X -scores of the reference level in Treatment in data

PRC_star matrix with the unconstrained PRC scores.

weights names list of columns weights and row weights. Names are `K`, `R`, `sqrtK`, `sqrtR`. The first two are column (species) and row (site) totals in with `method != "rda"`. They not normalized; none sums to 1. All 1 in `method == "rda"`.

obj a named list of `LMMsolve` objects, one for each axis.

objH0 a named list of `LMMsolve` objects of the null model, one for each axis.

iter the number of iterations of the accelerated power algorithm.

Y the response matrix.

model copy of the `model` argument to the `powerSVD`.

options_iter copy of the `options_iter` argument to the `smoothPRC`.

axes an internal matrix: Intercept, optionally covariates and previous axes.

n_covariates the number of user-defined row covariates using `formula_covariates` with one for the "(Intercept)".

References

Boer, Martin P. 2023. Tensor Product P-Splines Using a Sparse Mixed Model Formulation. *Statistical Modelling* 23 (5-6): 465–79. doi:[10.1177/1471082X231178591](https://doi.org/10.1177/1471082X231178591)

update_fit_state	<i>Updates the parameters of solver (internal)</i>
------------------	--

Description

Updates the parameters of solver (internal)

Usage

```
update_fit_state(object, state)
```

Arguments

object	Object resulting for the fit.
state	Optional list of arguments for warm starts of the solver

Index

* **datasets**
 CPFNUTphytoplankton, 4

anova.smoothPRC, 3
arrangeGrob, 22, 23, 25, 26, 29

biomonitoring, 3

capscale, 30
cca, 5, 6, 11, 30, 32
cca.object, 6, 29–31
CPFNUTphytoplankton, 4

dbrda, 30
diag, 39
dim, 39
dist, 37, 39
doPRC, 5, 5, 11, 14, 15, 18, 21–26, 28, 31, 34

expansion, 18, 26

fit_full_model_columns, 8
fit_full_model_rows, 8
fit_null_model_columns, 9
fit_null_model_rows, 9
fvalues4levels, 10, 14, 15, 22, 25, 35

geom_label_repel, 18, 26
geom_text_repel, 18, 26
get_focal_and_conditioning_factors, 11, 15, 30, 31
grid.arrange, 22, 23, 26, 28

how, 3

initialize_pwr_alg, 12

LMMsolve, 34, 38, 39

ordiYbar, 32
Ossenkampen, 12

plot_sample_scores_cdt, 13, 23, 26
plot_smoothPRC_cdt, 15
plot_species_scores_2d, 17, 26
plot_species_scores_bk, 19, 23
plotPRC, 6, 15, 20, 21, 31
plotPRC2d, 6, 18, 23, 24, 31
plotsmoothPRC, 20, 28
PRC_scores, 6, 11, 14, 15, 21, 22, 24, 25, 29
prcomp, 6, 30
predict, 30

rda, 5, 6, 11, 17, 30, 32
rdacca_taxon_stats, 32

scores, 30, 32
set_smoothPRC_model, 10, 33, 37, 38
SimData, 36
smoothPRC, 3, 15, 28, 35, 37

update_fit_state, 40