Package: ggord (via r-universe)

August 21, 2024

Type Package

Title Ordination Plots with ggplot2

Version 1.1.8

Date 2024-07-16

Author Marcus W. Beck [aut, cre]

Maintainer Marcus W. Beck <mbafs2012@gmail.com>

Description Create ordination biplots with ggplot2. Multiple methods are available including principal components analysis, correspondence analysis, nonmetric multidimensional scaling, multiple correspondence analysis, and linear discriminant analysis. Limited triplot methods are also available.

BugReports https://github.com/fawda123/ggord/issues

License CC0

LazyData true

Imports ggplot2, ggrepel, grid, plyr

Depends R (>= 3.1.1)

Suggests knitr, ade4, adephylo, ape, ca, FactoMineR, MASS, phylobase, vegan, rmarkdown

RoxygenNote 7.2.3

VignetteBuilder knitr

Repository https://fawda123.r-universe.dev

RemoteUrl https://github.com/fawda123/ggord

RemoteRef HEAD

RemoteSha 1caae694eb64f804b6b14e027b9a08cbf27e56e4

Contents

`ggord` *Ordination plots with ggplot2*

Description

Create an ordination biplot using ggplot2 including options for selecting axes, group color aesthetics, and selection of variables to plot.

Usage

```
ggord(...)

## Default S3 method:
ggord(
  obs,
  vecs,
  axes = c("1", "2"),
  grp_in = NULL,
  cols = NULL,
  facet = FALSE,
  nfac = NULL,
  addpts = NULL,
  obslab = FALSE,
  ptslab = FALSE,
  ellipse = TRUE,
  ellipse_pro = 0.95,
  poly = TRUE,
  polylntyp = "solid",
  hull = FALSE,
  arrow = 0.4,
  labcol = "black",
  veccol = "black",
  vectyp = "solid",
  veclsz = 0.5,
  ext = 1.2,
  repel = FALSE,
  vec_ext = 1,
  vec_lab = NULL,
  size = 4,
  sizelab = NULL,
  addsize = size/2,
  addcol = "blue",
  addpch = 19,
  txt = 4,
  alpha = 1,
  alpha_el = 0.4,
  xlims = NULL,
  ylims = NULL,
  var_sub = NULL,
  coord_fix = TRUE,
  parse = TRUE,
  grp_title = "Groups",
  force = 1,
  max.overlaps = 10,
  exp = c(0, 0),
  ...
)

## S3 method for class 'PCA'
ggord(ord_in, grp_in = NULL, axes = c("1", "2"), ...)

## S3 method for class 'MCA'
ggord(ord_in, grp_in = NULL, axes = c("1", "2"), ...)

## S3 method for class 'mca'
ggord(ord_in, grp_in = NULL, axes = c("1", "2"), ...)

## S3 method for class 'acm'
ggord(ord_in, grp_in = NULL, axes = c("1", "2"), ...)

## S3 method for class 'prcomp'
ggord(ord_in, grp_in = NULL, axes = c("1", "2"), ...)

## S3 method for class 'princomp'
ggord(ord_in, grp_in = NULL, axes = c("1", "2"), ...)

## S3 method for class 'metaMDS'
ggord(ord_in, grp_in = NULL, axes = c("1", "2"), ...)

## S3 method for class 'lda'
ggord(ord_in, grp_in = NULL, axes = c("1", "2"), ...)

## S3 method for class 'pca'
ggord(ord_in, grp_in = NULL, axes = c("1", "2"), ...)

## S3 method for class 'coa'
ggord(ord_in, grp_in = NULL, axes = c("1", "2"), ...)

## S3 method for class 'ca'
ggord(ord_in, grp_in = NULL, axes = c("1", "2"), ...)

## S3 method for class 'ppca'
ggord(ord_in, grp_in = NULL, axes = NULL, ...)

## S3 method for class 'rda'
```

```
ggord(ord_in, grp_in = NULL, axes = c("1", "2"), ...)

## S3 method for class 'capscale'
ggord(ord_in, grp_in = NULL, axes = c("1", "2"), ...)

## S3 method for class 'dbrda'
ggord(ord_in, grp_in = NULL, axes = c("1", "2"), ...)

## S3 method for class 'cca'
ggord(ord_in, grp_in = NULL, axes = c("1", "2"), ...)

## S3 method for class 'dpcoa'
ggord(ord_in, grp_in = NULL, axes = c("1", "2"), ...)
```

Arguments

`...`	arguments passed to or from other methods
`obs`	matrix or data frame of axis scores for each observation
`vecs`	matrix or data frame of axis scores for each variable
`axes`	chr string indicating which axes to plot
`grp_in`	vector of grouping objects for the biplot, must have the same number of observations as the original matrix used for the ordination
`cols`	chr string of optional colors for `grp_in`
`facet`	logical indicating if plot is faceted by groups in `grp_in`
`nfac`	numeric indicating number of columns if `facet = TRUE`
`addpts`	optional matrix or data.frame of additional points if constrained ordination is used (e.g., species locations in cca, rda)
`obslab`	logical if the row names for the observations in `obs` are plotted rather than points
`ptslab`	logical if the row names for the additional points (`addpts`) in constrained ordination are plotted as text
`ellipse`	logical if confidence ellipses are shown for each group, method from the ggbiplot package, at least one group must have more than two observations
`ellipse_pro`	numeric indicating confidence value for the ellipses
`poly`	logical if confidence ellipses are filled polygons, otherwise they are shown as empty ellipses
`polylntyp`	chr string for line type of polygon outlines if `poly = FALSE`, options are `twodash`, `solid`, `longdash`, `dotted`, `dotdash`, `dashed`, `blank`, or alternatively the grouping vector from `grp_in` can be used
`hull`	logical if convex hull is drawn around points or groups if provided
`arrow`	numeric indicating length of the arrow heads on the vectors, use `NULL` to suppress arrows
`labcol`	chr string for color of text labels on vectors
`veccol`	chr string for color of vectors

`vectyp`	chr string for line type of vectors, options are `twodash`, `solid`, `longdash`, `dotted`, `dotdash`, `dashed`, `blank`
`veclsz`	numeric for line size on vectors
`ext`	numeric indicating scalar distance of the labels from the arrow ends
`repel`	logical if overlapping text labels on vectors use `geom_text_repel` from the ggrepel package
`vec_ext`	numeric indicating a scalar extension for the ordination vectors
`vec_lab`	list of optional labels for vectors, defaults to names from input data. The input list must be named using the existing variables in the input data. Each element of the list will have the desired name change.
`size`	numeric indicating size of the observation points or a numeric vector equal in length to the rows in the input data
`sizelab`	chr string indicating an alternative legend title for size
`addsize`	numeric indicating size of the species points if addpts is not `NULL`
`addcol`	numeric indicating color of the species points if addpts is not `NULL`
`addpch`	numeric indicating point type of the species points if addpts is not `NULL`
`txt`	numeric indicating size of the text labels for the vectors, use `NULL` to suppress labels
`alpha`	numeric transparency of points and ellipses from 0 to 1
`alpha_el`	numeric transparency for confidence ellipses, also applies to filled convex hulls
`xlims`	two numeric values indicating x-axis limits
`ylims`	two numeric values indicating y-axis limits
`var_sub`	chr string indcating which labels to show. Regular expression matching is used.
`coord_fix`	logical indicating fixed, equal scaling for axes
`parse`	logical indicating if text labels are parsed
`grp_title`	chr string for legend title
`force`	numeric passed to `force` argument in `geom_text_repel` from the ggrepel package
`max.overlaps`	numeric passed to `max.overlaps` argument in `geom_text_repel` from the ggrepel package
`exp`	numeric of length two for expanding x and y axes, passed to `scale_y_continuous` and `scale_y_continuous`
`ord_in`	input ordination object

Details

Explained variance of axes for triplots are constrained values.

Value

A `ggplot` object that can be further modified

See Also

ggplot

Examples

```
library(ggplot2)

# principal components analysis with the iris data set
# prcomp
ord <- prcomp(iris[, 1:4])

p <- ggord(ord, iris$Species)
p

p <- ggord(ord, iris$Species, cols = c('purple', 'orange', 'blue'))
p

p + scale_shape_manual('Groups', values = c(1, 2, 3))
p + theme_classic()
p + theme(legend.position = 'top')

# change the vector labels with vec_lab
new_lab <- list(Sepal.Length = 'SL', Sepal.Width = 'SW', Petal.Width = 'PW',
 Petal.Length = 'PL')
p <- ggord(ord, iris$Species, vec_lab = new_lab)
p

# faceted by group
p <- ggord(ord, iris$Species, facet = TRUE, nfac = 3)
p

# principal components analysis with the iris dataset
# princomp
ord <- princomp(iris[, 1:4])

ggord(ord, iris$Species)

# principal components analysis with the iris dataset
# PCA
library(FactoMineR)

ord <- PCA(iris[, 1:4], graph = FALSE)

ggord(ord, iris$Species)

# principal components analysis with the iris dataset
# dudi.pca
library(ade4)

ord <- dudi.pca(iris[, 1:4], scannf = FALSE, nf = 4)

ggord(ord, iris$Species)
```

```
# multiple correspondence analysis with the tea dataset
# MCA
data(tea, package = 'FactoMineR')
tea <- tea[, c('Tea', 'sugar', 'price', 'age_Q', 'sex')]

ord <- MCA(tea[, -1], graph = FALSE)

ggord(ord, tea$Tea, parse = FALSE) # use parse = FALSE for labels with non alphanumeric characters

# multiple correspondence analysis with the tea dataset
# mca
library(MASS)

ord <- mca(tea[, -1])

ggord(ord, tea$Tea, parse = FALSE) # use parse = FALSE for labels with non alphanumeric characters

# multiple correspondence analysis with the tea dataset
# acm
ord <- dudi.acm(tea[, -1], scannf = FALSE)

ggord(ord, tea$Tea, parse = FALSE) # use parse = FALSE for labels with non alphanumeric characters

# nonmetric multidimensional scaling with the iris dataset
# metaMDS
library(vegan)
ord <- metaMDS(iris[, 1:4])

ggord(ord, iris$Species)

# linear discriminant analysis
# example from lda in MASS package
ord <- lda(Species ~ ., iris, prior = rep(1, 3)/3)

ggord(ord, iris$Species)

# correspondence analysis
# dudi.coa
ord <- dudi.coa(iris[, 1:4], scannf = FALSE, nf = 4)

ggord(ord, iris$Species)

# correspondence analysis
library(ca)
ord <- ca(iris[, 1:4])

ggord(ord, iris$Species)

# double principle coordinate analysis (DPCoA)
library(ade4)
data(ecomor)
grp <- rep(c("Bu", "Ca", "Ch", "Pr"), each = 4)    # sample groups
```

```
dtaxo <- dist.taxo(ecomor$taxo)                    # taxonomic distance between species
ord <- dpcoa(data.frame(t(ecomor$habitat)), dtaxo, scan = FALSE, nf = 2)

ggord(ord, grp_in = grp, ellipse = FALSE, arrow = 0.2, txt = 3)

# phylogenetic PCA
# ppca

library(adephylo)
library(phylobase)
library(ape)

data(lizards)

# example from help file, adephylo::ppca
# original example from JOMBART ET AL 2010

# build a tree and phylo4d object
liz.tre <- read.tree(tex=lizards$hprA)
liz.4d <- phylobase::phylo4d(liz.tre, lizards$traits)

# remove duplicated populations
liz.4d <- phylobase::prune(liz.4d, c(7,14))

# correct labels
lab <- c("Pa", "Ph", "Ll", "Lmca", "Lmcy", "Phha", "Pha",
         "Pb", "Pm", "Ae", "Tt", "Ts", "Lviv", "La", "Ls", "Lvir")
tipLabels(liz.4d) <- lab

# remove size effect
dat <- tdata(liz.4d, type="tip")
dat <- log(dat)
newdat <- data.frame(lapply(dat, function(v) residuals(lm(v~dat$mean.L))))
rownames(newdat) <- rownames(dat)
tdata(liz.4d, type="tip") <- newdat[,-1] # replace data in the phylo4d object

# create ppca
liz.ppca <- ppca(liz.4d,scale=FALSE,scannf=FALSE,nfposi=1,nfnega=1, method="Abouheif")

# plot
ggord(liz.ppca)

# distance-based redundancy analysis
# dbrda from vegan
data(varespec)
data(varechem)

ord <- dbrda(varespec ~ N + P + K + Condition(Al), varechem, dist = "bray")

ggord(ord)

######
# triplots
# redundancy analysis
# rda from vegan
ord <- rda(varespec, varechem)

ggord(ord)

# distance-based redundancy analysis
# capscale from vegan
ord <- capscale(varespec ~ N + P + K + Condition(Al), varechem, dist = "bray")

ggord(ord)

# canonical correspondence analysis
# cca from vegan
ord <- cca(varespec, varechem)

ggord(ord)

# species points as text
# suppress site points
ggord(ord, ptslab = TRUE, size = NA, addsize = 5, parse = TRUE)
```

Index