Package ‘vegan3d’

October 25, 2018

Title Static and Dynamic 3D Plots for the 'vegan' Package
Version 1.1-2
Date 2018-10-25
Depends R (>= 3.2.0), vegan (>= 2.3-0)
Imports cluster, rgl, scatterplot3d (>= 0.3-40)
Description Static and dynamic 3D plots to be used with ordination results and in diversity analysis, especially with the vegan package.
License GPL-2

BugReports https://github.com/vegandevs/vegan3d/issues
URL https://cran.r-project.org/, https://github.com/vegandevs/vegan3d
NeedsCompilation no
Author Jari Oksanen [aut, cre],
Roeland Kindt [aut],
Gavin L. Simpson [aut]
Maintainer Jari Oksanen <jhoksane@gmail.com>
Repository CRAN
Date/Publication 2018-10-25 13:30:03 UTC

R topics documented:

vegan3d-package ................................................. 2
ordiplot3d ...................................................... 3
ordirgl .......................................................... 5
orditree3d ....................................................... 8
rgl.isomap ....................................................... 9
rgl.renyiaccum .................................................. 10

Index 12
Description

The vegan3d package provides 3D plotting for all vegan ordination methods or any other ordination method that vegan scores function can handle. It can also display hclust results in 3D over a 2D plane. Dynamic 3D plots are based on the rgl package and static plots are drawn with the scatterplot3d package.

Index of help topics:

- ordiplot3d: Three-Dimensional Ordination Graphics
- ordirgl: Three-Dimensional Dynamic Ordination Graphics
- orditree3d: Draw Cluster Tree over a Plane
- rgl.isomap: Dynamic 3D plot of isomap ordination.
- rgl.renyiaccum: Dynamic Perspective Plot of Renyi Diversity Accumulation
- vegan3d-package: Dynamic and Static 3D Plotting for Ordination and Clustering

Drawing with rgl Functions

The rgl graphics are dynamic 3D plots that can be spun and zoomed by mouse. The vegan3d package provides interface to ordination and clustering objects. The functions use rgl setting and conventions and do not change the user settings. For general configuration of the plots, users should check rgl documentation. For instance, general look and feel of drawn items can be configured with rgl.material.

The rgl package may not be available in all platforms, and therefore the package is not automatically attached. If you want to use rgl functions, you must either prefix commands with rgl:: or call library(rgl) in your session.

Function ordirgl is similar as ordiplot in vegan, and any ordination result can be drawn with similar conventions. Functions with orgl prefix add items to existing plots, for instance, orglellipse is analogous to ordiellipse.

Function ordirgltree draws an hclust dendrogram over a plane. It was originally developed for 2D ordination planes, but any other plane can be used, for instance a projected map.

Functions rgl.isomap and rgl.renyiaccum provide alternative dynamic 3D plots for vegan isomap and renyiaccum functions.

Drawing with scatterplot3d Functions

The scatterplot3d package draws static 3D graphics, and vegan3d provides an interface for ordination and clustering objects. You must consult the scatterplot3d documentation for configuring your plots.

Function ordiplot3d is similar to ordirgl or ordiplot and draws a static 3D plot in the standard graphical device. It returns invisibly a plotting object which contains the projected points, and
Description

Function `ordiplot3d` displays three-dimensional ordination graphics using `scatterplot3d`. Function works with all ordination results from `vegan` and all ordination results known by `scores` function.

Usage

`ordiplot3d(object, display = "sites", choices = 1:3, col = "black", 
ax.col = "red", arr.len = 0.1, arr.col = "blue", envfit, 
xlab, ylab, zlab, ...)`

Arguments

- **object**: An ordination result or any object known by `scores`.
- **display**: Display "sites" or "species" or other ordination object recognized by `scores`.
- **choices**: Selected three axes.
- **col**: Colours of points. Can be a vector, and factors are interpreted as their internal numerical codes.
- **ax.col**: Axis colour (concerns only the crossed axes through the origin).
- **arr.len**: 'Length' (width) of arrow head passed to `arrows` function.
- **arr.col**: Colour of biplot `arrows` and centroids of environmental variables.
- **envfit**: Fitted environmental variables from `envfit` displayed in the graph.
- **xlab, ylab, zlab**: Axis labels passed to `scatterplot3d`. If missing, labels are taken from the ordination result. Set to NA to suppress labels.
- **...**: Other parameters passed to graphical functions.

Details

Function `ordiplot3d` plots static three-dimensional scatter diagrams using `scatterplot3d`. Function uses most default settings of underlying graphical functions, and you must consult their help pages to change graphics to suit your taste (see `scatterplot3d`).

Function returns invisibly an object of class `ordiplot3d` which inherits from `ordiplot`. The result object contains the projected coordinates of plotted items and functions to convert 3D data to 2D (see `scatterplot3d`). Function will display only one selected set of `scores`, typically either
"sites" or "species". Examples show how to use the invisible return object to add another set of points to the projected plot.

In constrained ordination (cca, rda, capscale), biplot arrows and centroids are always displayed similarly as in two-dimensional plotting function plot.cca. Alternatively, it is possible to display fitted environmental vectors or class centroids from envfit. These are displayed similarly as the results of constrained ordination, and they can be shown only for non-constrained ordination. The user must remember to specify at least three axes in envfit if the results are used with these functions.

The function has a scores method to extract the projected coordinates from the invisible return object. Standard vegan functions can be used with the returned object. You can use any function from the ordihull and ordiarrows families (see Examples).

Value

Function ordiplot3d returns invisibly an object of class "ordiplot3d" inheriting from ordiplot. The return object will contain the coordinates projected onto two dimensions for points, and the projected coordinates of origin, and possibly the projected coordinates of the heads of arrows and centroids of environmental variables. The result will also contain the object returned by scatterplot3d, including function xyz.convert which projects three-dimensional coordinates onto the plane used in the current plot (see Examples). In addition, there is a function envfit.convert that projects a three-dimensional envfit object to the current plot.

Warning

Please note that scatterplot3d sets internally some graphical parameters (such as mar for margins) and does not honour default settings. It is advisable to study carefully the documentation and examples of scatterplot3d.

Author(s)

Jari Oksanen

See Also

scatterplot3d, ordiplot, ordiarrows, ordihull.

Examples

```r
### Default 'ordiplot3d'
data(dune, dune.env)
ord <- cca(dune ~ A1 + Moisture, dune.env)
ordiplot3d(ord)
### A boxed 'pin' version
ordiplot3d(ord, type = "h")
### More user control
pl <- ordiplot3d(ord, scaling = "symmetric", angle=15, type="n")
points(pl, "points", pch=16, col="red", cex = 0.7)
### identify(pl, "arrows", col="blue") would put labels in better positions
text(pl, "arrows", col="blue", pos=3)
text(pl, "centroids", col="blue", pos=1, cex = 1)
```
### Add species using `xyz.convert` function returned by `ordiplot3d`

```r
sp <- scores(ord, choices=1:3, display="species", scaling="symmetric")
text(pl1$xyz.convert(sp), rownames(sp), cex=0.7, xpd=TRUE)
```

### Two ways of adding fitted variables to ordination plots

```r
ord <- cca(dune)
ef <- envfit(ord ~ Moisture + A1, dune.env, choices = 1:3)
```

1. Use argument 'envfit'

```r
ordiplot3d(ord, envfit = ef)
```

2. Use returned `envfit.convert` function for better user control

```r
pl3 <- ordiplot3d(ord)
plot(pl3$envfit.convert(ef), at = pl3$origin)
```

`envfit.convert()` also handles different 'choices' of axes

```r
pl3 <- ordiplot3d(ord, choices = c(1,3,2))
plot(pl3$envfit.convert(ef), at = pl3$origin)
```

### vegan::ordiXXX functions can add items to the plot

```r
ord <- cca(dune)
pl4 <- with(dune.env, ordiplot3d(ord, col = Management, pch=16))
with(dune.env, ordiellipse(pl4, Management, draw = "poly", col = 1:4,
  alpha = 0.6))
with(dune.env, ordispider(pl4, Management, col = 1:4, label = TRUE))
```

---

**ordirgl**  
*Three-Dimensional Dynamic Ordination Graphics*

**Description**

Function `ordirgl` displays three-dimensional dynamic ordination graphs which can be rotated and zoomed. This function works with all ordination results from `vegan` and all ordination results known by the `scores` function. The `orgl`-prefixed functions add elements to the `ordirgl` graph similarly as `ordi`-prefixed functions in `vegan`.

**Usage**

```r
ordirgl(object, display = "sites", choices = 1:3, type = "p", col = "black",
ax.col = "red", arr.col = "yellow", radius, text, envfit, ...)
orglpoints(object, display = "sites", choices = 1:3, radius, col = "black", ...)
orgltext(object, text, display = "sites", choices = 1:3, adj = 0.5,
col = "black", ...)
orglsegments(object, groups, order.by, display = "sites", choices = 1:3,
col = "black", ...)
orglspider(object, groups, display = "sites", w = weights(object, display),
choices = 1:3, col = "black", ...)
orgllellipse(object, groups, display = "sites", w = weights(object, display),
kind = c("sd", "se", "ehull"), conf, choices = 1:3, alpha = 0.3,
col = "red", ...)
orglspanmtree(object, spantree, display = "sites", choices = 1:3,
col = "black", ...)
orglcluster(object, cluster, prune = 0, display = "sites", choices = 1:3,
col = "black", ...)
```
Arguments

object      An ordination result or any object known by \texttt{scores}.
display    Display "sites" or "species" or other ordination object recognized by \texttt{scores}.
choices    Selected three axes.
type         The type of plots: "p" for points or "t" for text labels.
ax.col      Axis colour (concerns only the crossed axes through the origin).
arr.col     Colour of biplot \texttt{arrows} and centroids of environmental variables.
radius      Size of points in the units of ordination scores.
text        Text to override the default with type = "t".
envfit       Fitted environmental variables from \texttt{envfit} displayed in the graph. Use envfit = NA to suppress display of environmental variables in constrained ordination.
adj          Text justification passed to \texttt{rgl.texts}.
groups      Factor giving the groups for which the graphical item is drawn.
order.by    Order points by this variable within groups.
w          Weights used to find the average within group. Weights are used automatically for \texttt{cca} and \texttt{decorana} results, unless undone by the user. w=NULL sets equal weights to all points.
kind         Draw ellipse for standard deviations of points ("sd") or standard deviations of their averages ("se") or an ellipsoid hull enclosing all points in the group ("ehull").
conf         Confidence limit for ellipses, e.g., 0.95. If not given, sd or se ellipses are drawn.
col          Colour of items. This can be a vector and factors are interpreted as their internal numerical values. If the function has a groups argument, vector \texttt{col} is used for each of these, and for other functions it is matched to points in \texttt{ordirgl} (see Details below).
alpha       Transparency of colour between 0.0 (fully transparent) and 1.0 (non-transparent).
spantree     A minimum spanning tree object from \texttt{vegan} \texttt{spantree}.
cluster     Result of hierarchic cluster analysis, such as \texttt{hclust} or \texttt{agnes}.
prune        Number of upper levels hierarchies removed from the tree. If prune > 0, tree will be cut into prune + 1 disconnected trees.
...          Other parameters passed to graphical functions.

Details

Function \texttt{ordirgl} plots dynamic graphics using OpenGL with the \texttt{rgl} package. It clears the graphics device and starts a new plot. The function was designed for ordination methods in the \texttt{vegan} package, but it can handle any method known to \texttt{vegan} \texttt{scores} function, or to any three column matrix. The \texttt{orgl}-prefixed functions add items to the opened \texttt{rgl} graphics device.

Function \texttt{ordirgl} uses most default settings of underlying graphical functions in \texttt{rgl}. It plots only one set of points, but functions \texttt{orglpoints} and \texttt{orgltext} can add new items to an existing plot. The points are plotted using \texttt{rgl.spheres} and the text using \texttt{rgl.texts} which both have their own configuration switches and their general look and feel can be modified with \texttt{rgl.material}. The
point size is directly defined by radius argument in the units of ordination scores in \texttt{rgl.spheres}, but \texttt{ordirgl} uses a default size of 1% of the length of the longest axis, and this can be further modified by the \texttt{cex} multiplier.

In constrained ordination (\texttt{cca}, \texttt{rda}, \texttt{capscale}), biplot arrows and centroids are always displayed similarly as in two-dimensional plotting function \texttt{plot.cca}. Alternatively, it is possible to display fitted environmental vectors or class centroids from \texttt{envfit} in both graphs. These are displayed similarly as the results of constrained ordination, and they can be shown only for non-constrained ordination. The user must remember to specify at least three axes in \texttt{envfit} if the results are used with these functions.

Function \texttt{orglsegments} is similar to \texttt{vegan ordisegments} and connects points by line segments. This can be useful for regular transects. The colour of segments can be a vector which corresponds to the groups and will be recycled.

Function \texttt{orglspider} is similar as \texttt{vegan ordispider}: it connects points to their weighted centroid within "groups", and in constrained ordination it can connect "wa" or weighted averages scores to corresponding "1c" or linear combination scores if "groups" is missing. Function \texttt{orglellipse} is similar as \texttt{vegan ordiellipse} and draws ellipsoids of standard deviance, standard error or confidence regions for groups. At least four points are needed to define an ellipsoid in 3D, and even these will fail if all points are strictly on 2D. The \texttt{col} argument for both of these functions can be a vector corresponding to the groups.

Function \texttt{orglspantrtree} adds a minimum spanning tree from \texttt{vegan spantrtree}. This a 3D equivalent of \texttt{lines.spantrtree}. Function \texttt{orglcluster} adds a hierarchic cluster tree from \texttt{hclust} or related functions. This is a 3D equivalent of \texttt{ordicluster}. The \texttt{col} argument for both of these functions can be a vector corresponding to the connected points. In \texttt{orglspantrtree} the line colour is a mixture of colours of joined points, and in \texttt{orglcluster} it is a mixture of all points in the cluster.

\textbf{Value}

Function \texttt{ordirgl} returns nothing.

\textbf{Warning}

Function \texttt{ordirgl} uses OpenGL package \texttt{rgl} which may not be functional in all platforms.

\textbf{Author(s)}

Jari Oksanen

\textbf{See Also}

\texttt{rgl, rgl.spheres, rgl.texts, rgl.viewpoint, envfit}. These are 3D dynamic variants of \texttt{vegan} functions \texttt{ordiplot, ordisegments, ordispider and ordiellipse, ordicluster and lines.spantrtree}.

\textbf{Examples}

\begin{verbatim}
if (interactive() && require(rgl, quietly = TRUE)) {
  data(mite, mite.env)
  ord <- rda(decostand(mite, "hellinger"))
}
\end{verbatim}
ordirgl(ord, size=4, col = "yellow")
orgltext(ord, display = "species")
## show groups of Shrub abundance
## ordirgl: col by points
with(mite.env, ordirgl(ord, col = as.numeric(Shrub), scaling = "sites"))
## orglspider & orglellipse: col by groups
with(mite.env, orglspider(ord, Shrub, col = 1:3, scaling = "sites"))
with(mite.env, orglellipse(ord, Shrub, col = 1:3, kind = "se", conf = 0.95,
scaling = "sites"))

---

orditree3d

**Draw Cluster Tree over a Plane**

**Description**

Function draws a 3D plot where ordination result is at the bottom plane and a *hclust* dendrogram is drawn above the plane.

**Usage**

orditree3d(ord, cluster, prune = 0, display = "sites", choices = c(1, 2),
col = "blue", text, type = "p", ...)  
ordirgltree(ord, cluster, prune = 0, display = "sites", choices = c(1, 2),
col = "blue", text, type = "p", ...)

**Arguments**

- **ord** An ordination object or an *ordiplot* object or any other structure defining a 2D plane.
- **cluster** Result of hierarchic cluster analysis, such as *hclust* or *agnes* or any other clustering that can be coerced to a compliant format by as.hclust.
- **prune** Number of upper levels hierarchies removed from the tree. If prune > 0, tree will be cut into prune + 1 disconnected trees.
- **choices** Choice of ordination axes.
- **display** Ordination scores displayed.
- **col** Colour of tree. The colour can be a vector and it is used for the points, text and terminal branches. The colour of internal branches is a mixture of connected leaves.
- **text** Text to replace the default of item labels when type = "t".
- **type** Display of leaves: "p" for points, "t" for text, and "n" for no display.
- **...** Arguments passed to *scores* and graphical functions.
Details

orditree3d uses `scatterplot3d` package to draw a static 3D plot of the dendrogram over the ordination, and ordirgltree uses `rgl` to make a dynamic, spinnable plot. The functions were developed to plot a cluster dendrogram over a 2D ordination plane, but any other plane can be used, for instance, a map.

Value

Function orditree3d returns invisibly a `scatterplot3d` result object amended with items `points` and `internal` that give the projected coordinates of ordination scores and internal nodes, and `col.points` and `col.internal` that give their colours. All matrix-like objects can be accessed with scores.

Function ordirgltree returns nothing.

Author(s)

Jari Oksanen.

See Also

`orglcluster` and `ordicluster` (in `vegan`).

Examples

```r
data(dune, dune.env)
d <- vegdist(dune)
m <- metaMDS(d)
c1 <- hclust(d, "aver")
orditree3d(m, c1, pch=16, col=cutree(cl, 3))
```

rgl.isomap

*Dynamic 3D plot of isomap ordination.*

Description

Function displays a dynamic 3D plot from `isomap` ordination.

Usage

```r
rgl.isomap(x, web = "white", ...)
```
rgl.renyiaccum

Arguments

- `x` Result from `isomap`.
- `web` Colour of the web. If this is a vector matching the number of points, the colour of links is a mixture of joined points. NA skips drawing the web.
- `...` Other parameters passed to `ordirgl` and `scores`.

Details

Function `rgl.isomap` displays dynamic 3D plots that can be rotated on the screen. The functions is based on `ordirgl`, but it adds the connecting lines. The function passes extra arguments to `scores` or `ordirgl` functions so that you can select axes, or define colours and sizes of points.

Value

Function returns nothing.

Note

This is a support function for `isomap` ordination in the `vegan` package.

Author(s)

Jari Oksanen.

See Also

`isomap`, `ordirgl`, `scores`.

Examples

```r
if (interactive() && require(rgl, quietly = TRUE)) {
  data(BCI)
  dis <- vegdist(BCI)
  ## colour points and links by the dominant species
  dom <- factor(make.cepnames(names(BCI))[apply(BCI, 1, which.max)])
  ord <- isomap(dis, k=3)
  rgl.isomap(ord, col = as.numeric(dom), web = as.numeric(dom), lwd=2)
}
```

rgl.renyiaccum  Dynamic Perspective Plot of Renyi Diversity Accumulation

Description

Function `rgl.renyiaccum` displays a dynamic 3D plot of the result of `renyiaccum` function in the `vegan` package. Function `persp.renyiaccum` (in `vegan`) produces similar static plots.
rgl.renyiaccum

Usage

rgl.renyiaccum(x, rgl.height = 0.2, ...)

Arguments

x A renyiaccum result.
rgl.height Vertical scaling of the plot.
... Other arguments passed to the function (ignored).

Details

This is a graphical support function to renyiaccum in vegan. Similar static plots can be produced by persp.renyiaccum.

Value

Function returns nothing.

Author(s)

Roeland Kindt.

See Also

renyiaccum, persp.renyiaccum, rgl.

Examples

if (interactive() && require(rgl, quietly = TRUE)){
  data(BCI)
  mod <- renyiaccum(BCI[1:12,])
  persp(mod)
  rgl.renyiaccum(mod)
}
Index

*Topic dynamic
  ordirgl, 5
  rgl.isomap, 9
  rgl.renyiaccum, 10
  vegan3d-package, 2

*Topic hplot
  ordiplot3d, 3
  ordirgl, 5
  orditree3d, 8
  rgl.isomap, 9
  rgl.renyiaccum, 10
  vegan3d-package, 2

*Topic multivariate
  vegan3d-package, 2

*Topic package
  vegan3d-package, 2

agnes, 6, 8
arrows, 3, 6
as.hclust, 8
capscale, 4, 7
cca, 4, 6, 7
decorana, 6
evfilt, 3, 4, 6, 7
hclust, 2, 3, 6–8
isomap, 2, 9, 10
lines.spantree, 7
ordiarrows, 4
ordicluster, 7, 9
ordiellipse, 2, 3, 7
ordihull, 4
ordiplot, 2–4, 7, 8
ordiplot3d, 2, 3
ordirgl, 2, 5, 10
ordirgltree, 2, 3
ordirgltree (orditree3d), 8
ordisegments, 7
ordispider, 7
orditree3d, 3, 8
orglcluster, 9
orglcluster (ordirgl), 5
orgllellipse, 2
orgllellipse (ordirgl), 5
orglpoints (ordirgl), 5
orglsegments (ordirgl), 5
orglspantree (ordirgl), 5
orglspider (ordirgl), 5
orgltext (ordirgl), 5
persp.renyiaccum, 10, 11
plot.cca, 4, 7
rda, 4, 7
renyiaccum, 2, 10, 11
rgl, 2, 6, 7, 11
rgl.isomap, 2, 9
rgl.material, 2, 6
rgl.renyiaccum, 2, 10
rgl.spheres, 6, 7
rgl.texts, 6, 7
rgl.viewpoint, 7
scatterplot3d, 2–4, 9
scores, 2, 3, 5, 6, 8, 10
scores.ordiplot3d (ordiplot3d), 3
spantree, 6, 7
vegan3d (vegan3d-package), 2
vegan3d-package, 2