Package ‘pavo’

October 3, 2017

Description A cohesive framework for parsing, analyzing and organizing color from spectral data.

Title Perceptual Analysis, Visualization and Organization of Spectral Color Data in R

Version 1.3.1

License GPL (>= 2)

Date 2017-10-01

Author Rafael Maia [aut, cre],
Thomas White [aut],
Chad Eliason [aut],
Pierre-Paul Bitton [aut]

Maintainer Rafael Maia <rm72@zips.uakron.edu>

URL http://rafaelmaia.net/pavo/

Depends R(>= 2.10)

Imports rcdd, mapproj, geometry, pbmcapply

Suggests rgl, testthat, knitr

VignetteBuilder knitr

RoxygenNote 6.0.1

NeedsCompilation no

Repository CRAN

Date/Publication 2017-10-02 22:41:45 UTC

R topics documented:

`pavo-package` *pavo: analyzing color data in R*

Description

An R package for the perceptual analysis, visualization and organization of color data

Usage

```
.PlotTetraEnv
```

Format

An object of class `environment` of length 0.

Details

Package:	pavo
Type:	Package
Version:	0.99
Date:	2016-11-11
License:	GPL (>= 2)
LazyLoad:	yes
To learn more about pavo, take a look at the vignettes:	
browseVignettes(package = "pavo")	

Author(s)

Rafael Maia <rm72@zips.uakron.edu>, Chad Eliason <cme16@zips.uakron.edu>, Pierre-Paul Bitton <bittonp@uwindsor.ca>, Thomas White <thomas.white026@gmail.com>

Examples

```
#see functions.
```

`aggplot` *Plot aggregated reflectance spectra*

Description

Combines and plots spectra (by taking the average and the standard deviation, for example) according to an index or a vector of identities.

Usage

```
aggplot(rspecdata, by = NULL, FUN.center = mean, FUN.error = sd,
  lcol = NULL, shadecol = NULL, alpha = 0.2, legend = FALSE, ...)
```

Arguments

`rspecdata`	(required) data frame containing the spectra to be manipulated and plotted.
`by`	(required) either a single value specifying the range of spectra within the data frame to be combined (for example, by = 3 indicates the function will be applied to groups of 3 consecutive columns in the spectra data frame) or a vector containing identifications for the columns in the spectra data frame (in which case the function will be applied to each group of spectra sharing the same identification).

`FUN.center`	the function to be applied to the groups of spectra, calculating a measure of central tendency (defaults to `mean`).
`FUN.error`	the function to be applied to the groups of spectra, calculating a measure of variation (defaults to `sd`).
`lcol`	color of plotted lines indicating central tendency.
`shadecol`	color of shaded areas indicating variance measure.
`alpha`	transparency of the shaded areas.
`legend`	automatically add a legend.
`...`	additional graphical parameters to be passed to plot.

Value

Plot containing the lines and shaded areas of the groups of spectra.

Author(s)

Rafael Maia <rm72@zips.uakron.edu>, Chad Eliason <cme16@zips.uakron.edu>

References

Montgomerie R (2006) Analyzing colors. In: Hill G, McGraw K (eds) Bird coloration. Harvard University Press, Cambridge, pp 90-147.

Examples

```
## Not run:
data(sicalis)
bysic <- gsub("^ind[0-9].",'', names(sicalis)[-1])
aggplot(sicalis, bysic)
aggplot(sicalis, bysic, FUN.error=function(x) quantile(x, c(0.0275,0.975)))
aggplot(sicalis, bysic, shade = spec2rgb(sicalis), lcol = 1)
aggplot(sicalis, bysic, lcol = 1, FUN.error = function(x) sd(x)/sqrt(length(x)))

## End(Not run)
```

`aggspec` *Aggregate reflectance spectra*

Description

Combines spectra (by taking the average, for example) according to an index or a vector of identities.

Usage

```
aggspec(rspecdata, by = NULL, FUN = mean, trim = TRUE)
```

Arguments

`rspecdata`	(required) data frame, possibly of class `rspec` containing the spectra to be manipulated. If it contains a wavelength column named "wl", that column will be ignored.
`by`	(required) either a single value specifying the range of spectra within the data frame to be combined (for example, by = 3 indicates the function will be applied to groups of 3 consecutive columns in the spectra data frame); a vector containing identifications for the columns in the spectra data frame (in which case the function will be applied to each group of spectra sharing the same identification); or a list of vectors, e.g., `by = list(sex, species)`.
`FUN`	the function to be applied to the groups of spectra. (defaults to `mean`)
`trim`	logical. if `TRUE` (default), the function will try to identify and remove numbers at the end of the names of the columns in the new rspec object.

Value

A data frame of class `rspec` containing the spectra after applying the aggregating function.

Author(s)

Chad Eliason `<cme16@zips.uakron.edu>`

References

Montgomerie R (2006) Analyzing colors. In: Hill G, McGraw K (eds) Bird coloration. Harvard University Press, Cambridge, pp 90-147.

Examples

```
## Not run:
data(teal)

# Average every two spectra
teal.sset1 <- aggspec(teal, by = 2)
plot(teal.sset1)

# Create factor and average spectra by levels 'a' and 'b'
ind <- rep(c('a', 'b'), times = 6)
teal.sset2 <- aggspec(teal, by = ind)

plot(teal.sset2)

## End(Not run)
```

as.rspec *Convert data to an rspec object*

Description

Converts data frames or matrices containing spectral data to `rspec` object

Usage

```
as.rspec(object, whichwl = NULL, interp = TRUE, lim = NULL)

is.rspec(object)
```

Arguments

`object`	(required) a data frame or matrix containing spectra to process.
`whichwl`	specifies which column contains wavelengths. If NULL (default), function searches for column containing equally spaced numbers and sets it as wavelengths "wl". If no wavelengths are found or `whichwl` is not given, returns arbitrary index values.
`interp`	whether to interpolate wavelengths in 1-nm bins (defaults to TRUE).
`lim`	vector specifying wavelength range to interpolate over (e.g., `c(300, 700)`).

Value

an object of class `rspec` for use in further `pavo` functions

a logical value indicating whether the object is of class `rspec`

Author(s)

Chad Eliason <`cme16@zips.uakron.edu`>

Examples

```
## Not run:

# Generate some fake reflectance data
fakedat <- data.frame(wl= c(300:700), refl1 = rnorm(401), refl2 = rnorm(401))
head(fakedat)

# Determine if is rspec object
is.rspec(fakedat)

# Convert to rspec object
fakedat2 <- as.rspec(fakedat)
is.rspec(fakedat2)
head(fakedat2)
## End(Not run)
```

axistetra *Plot reference axes in a static tetrahedral colorspace*

Description

Plots reference x, y and z arrows showing the direction of the axes in a static tetrahedral colorspace plot.

Usage

```
axistetra(x = 0, y = 1.3, size = 0.1, arrowhead = 0.05,
  col = par("fg"), lty = par("lty"), lwd = par("lwd"), label = TRUE,
  adj.label = list(x = c(0.003, 0), y = c(0.003, 0.003), z = c(0, 0.003)),
  label.cex = 1, label.col = NULL)
```

Arguments

`x, y`	position of the legend relative to plot limits (usually a value between 0 and 1, but because of the perspective distortion, values greater than 1 are possible)
`size`	length of the arrows. Can be either a single value (applied for x, y and z) or a vector of 3 separate values for each axis.
`arrowhead`	size of the arrowhead.
`col, lty, lwd`	graphical parameters for the arrows.
`label`	logical, include x, y and z labels (defaults to TRUE).
`adj.label`	position adjustment for the labels. a list of 3 named objects for x, y and z arrows, each with 2 values for x and y adjustment.
`label.cex, label.col`	graphical parameters for the labels.

Value

`axistetra` adds reference arrows showing the direction of the 3-dimensional axes in a static tetrahedral colorspace plot.

Author(s)

Rafael Maia `<rm72@zips.uakron.edu>`

`bgandilum` *Default background and illuminant data*

Description

Default background and illuminant data

Author(s)

Rafael Maia `<rm72@zips.uakron.edu>`

References

Endler, J. (1993). The Color of Light in Forests and Its Implications. Ecological Monographs, 63, 1-27.

`bootcoldist` *Bootstrap colour distance confidence intervals*

Description

Uses a bootstrap procedure to generate confidence intervals for the mean colour distance between two or more samples of colours

Usage

```
bootcoldist(vismodeldata, by, boot.n = 1000, alpha = 0.95,
  cores = getOption("mc.cores", 2L), ...)
```

Arguments

`vismodeldata`	(required) quantum catch color data. Can be the result from `vismodel`, or `colspace`. Data may also be independently calculated quantum catches, in the form of a data frame with columns representing photoreceptors.
`by`	(required) a vector containing indicating the group to wich each row from the object belongs to.
`boot.n`	number of bootstrap replicates (defaults to 1000)
`alpha`	the confidence level for the confidence intervals (defaults to 0.95)
`cores`	number of cores to be used in parallel processing. If `1`, parallel computing will not be used. Defaults to `getOption("mc.cores", 2L)`
`...`	other arguments to be passed to `coldist`. Must at minimum include `n` and `weber`. See `coldist` for details.

Value

a matrix including the empirical mean and bootstrapped confidence limits for dS (and dL if `achro = TRUE`).

Examples

```
## Not run:
 data(sicalis)
 vm <- vismodel(sicalis, achro='bt.dc')
 gr <- gsub("ind..", "", rownames(vm))
 bootcoldist(vm, gr, n=c(1,2,2,4), weber=0.1, weber.achro=0.1, cores=1)

## End(Not run)
```

`coldist` *Color distances*

Description

Calculates color distances. When data are the result of `vismodel`, it applies the receptor-noise model of Vorobyev et al. (1998) to calculate color distances with noise based on relative photoreceptor densities. It also accepts `colspace` data from the hexagon, colour-opponent-coding, categorical, segment, and cielab models, in which case euclidean distances (hexagon, cielab, categorical, segment) or manhattan distances (coc) are returned.

Usage

```
coldist(modeldata, noise = c("neural", "quantum"), subset = NULL,
  achro = FALSE, qcatch = NULL, n = c(1, 2, 2, 4), weber = 0.1,
  weber.ref = "longest", weber.achro = 0.1, v, n1, n2, n3, n4)
```

Arguments

`modeldata`	(required) quantum catch color data. Can be the result from `vismodel`, or `colspace`. Data may also be independently calculated quantum catches, in the form of a data frame with columns representing photoreceptors.
`noise`	how the noise will be calculated. (Ignored for `colspace` objects if model is not a receptor noise model (i.e. hexagon, colour-opponent-coding, categorical, segment, and cie models)): • `neural`: noise is proportional to the Weber fraction and is independent of the intensity of the signal received (i.e. assumes bright conditions). • `quantum`: noise is the sum of the neural noise and receptor noise, and is thus proportional to the Weber fraction and inversely proportional to the intensity of the signal received (the quantum catches). Note that the `quantum` option will only work with objects of class `vismodel`.

`subset`	If only some of the comparisons should be returned, a character vector of length 1 or 2 can be provided, indicating which samples are desired. The subset vector must match the labels of the input samples, but partial matching (and regular expressions) are supported.
`achro`	Logical. If `TRUE`, last column of the data frame is used to calculate the achromatic contrast, with noise based on the Weber fraction given by the argument `weber.achro`. If the data are from the hexagon model (i.e. `colspace(space = 'hexagon')`), it instead returns simple long (or 'green') receptor contrast.
`qcatch`	if the object is of class `vismodel` or `colspace`, this argument is ignored. If the object is a data frame of quantal catches from another source, this argument is used to specify what type of quantum catch is being used, so that the noise can be calculated accordingly: • `Qi`: Quantum catch for each photoreceptor • `fi`: Quantum catch according to Fechner law (the signal of the receptor channel is proportional to the logarithm of the quantum catch)
`n`	photoreceptor densities for the cones used in visual modeling. must have same length as number of columns (excluding achromatic receptor if used; defaults to the Pekin robin *Leiothrix lutea* densities: `c(1,2,2,4)`). Ignored for `colspace` objects if model is not a receptor noise model (i.e. hexagon, colour-opponent-coding, categorical, and cie models).
`weber`	The Weber fraction to be used. The noise-to-signal ratio `v` is unknown, and therefore must be calculated based on the empirically estimated Weber fraction of one of the cone classes. `v` is then applied to estimate the Weber fraction of the other cones. by default, the value of 0.1 is used (the empirically estimated value for the LWS cone from *Leiothrix lutea*). Ignored for `colspace` objects if model is not a receptor noise model (i.e. hexagon, colour-opponent-coding, categorical, segment, and cie models).
`weber.ref`	the cone class used to obtain the empirical estimate of the Weber fraction used for the `weber` argument. By default, `n4` is used, representing the LWS cone for *Leiothrix lutea*. Ignored for `colspace` objects if model is not a receptor noise model (i.e. hexagon, colour-opponent-coding, categorical, segment, and cie models).
`weber.achro`	the Weber fraction to be used to calculate achromatic contrast, when `achro = TRUE`. Defaults to 0.1. Ignored for `colspace` objects if model is not a receptor noise model (i.e. hexagon, colour-opponent-coding, categorical, segment, and cie models).
`n1, n2, n3, n4, v`	deprecated arguments. see below.

Value

A data frame containing up to 4 columns. The first two (`patch1, patch2`) refer to the two colors being contrasted; `dS` is the chromatic contrast (delta S) and `dL` is the achromatic contrast (delta L). Units are JND's in the receptor-noise model, euclidean distances in the categorical and segment space, manhattan distances in the color-opponent-coding space, green-receptor contrast in the hexagon, and lightness (L) contrast in the cielab model.

Note on previous versions

previous versions of `coldist` calculated receptor noise using the arguments `v` for the individual cone noise-to-signal ratio and `n1,n2,n3,n4` for the relative cone densities. These arguments have been replaced by `weber` and `n`, which takes a vector of relative cone densities. `weber.ref` allows the user to specify which receptor to use as the reference to obtain the desired Weber fraction, and `coldist` calculates internally the value of `v` to be used when calculating the Weber fraction for the remaining cones.

This allows a more explicit choice of Weber fraction, without the need to find the right value of `v` to use in order to obtain the desired signal-to-noise ratio. Furthermore, by allowing `n` to be entered as a vector, `coldist` can now handle visual systems with more than four photoreceptors.

In addition, the achromatic noise is calculated based on the `weber.achro` argument directly, and not based on `v` and `n4` as before.

Author(s)

Rafael Maia `<rm72@zips.uakron.edu>`

References

Vorobyev, M., Osorio, D., Bennett, A., Marshall, N., & Cuthill, I. (1998). Tetrachromacy, oil droplets and bird plumage colours. Journal Of Comparative Physiology A-Neuroethology Sensory Neural And Behavioral Physiology, 183(5), 621-633.

Hart, N. S. (2001). The visual ecology of avian photoreceptors. Progress In Retinal And Eye Research, 20(5), 675-703.

Endler, J. A., & Mielke, P. (2005). Comparing entire colour patterns as birds see them. Biological Journal Of The Linnean Society, 86(4), 405-431.

Olsson, P., Lind, O., & Kelber, A. (2015) Bird colour vision: behavioural thresholds reveal receptor noise. Journal of Experimental Biology, 218, 184-193.

Lind, O. (2016) Colour vision and background adaptation in a passerine bird, the zebra finch (Taeniopygia guttata). Royal Society Open Science, 3, 160383.

Examples

```
## Not run:
# Dichromat
data(flowers)
vis.flowers <- vismodel(flowers, visual = 'canis', relative = FALSE)
didist.flowers <- coldist(vis.flowers)

# Trichromat
vis.flowers <- vismodel(flowers, visual = 'apis', relative = FALSE)
tridist.flowers <- coldist(vis.flowers)

# Trichromat, color-hexagon model (euclidean distances)
vis.flowers <- vismodel(flowers, visual = 'apis', qcatch = 'Ei',
                        relative = FALSE, vonkries = TRUE, achro = 'l', bkg = 'green')
hex.flowers <- colspace(vis.flowers, space = 'hexagon')
hexdist.flowers <- coldist(hex.flowers)
```

```
# Trichromat, color-opponent-coding model (manhattan distances)
vis.flowers <- vismodel(flowers, visual = 'apis', qcatch = 'Ei', relative = FALSE, vonkries = TRUE)
coc.flowers <- colspace(vis.flowers, space = 'coc')
hexdist.flowers <- coldist(coc.flowers)

# Tetrachromat
data(sicalis)
vis.sicalis <- vismodel(sicalis, visual = 'avg.uv', relative = FALSE)
tetradist.sicalis.n <- coldist(vis.sicalis)

# This will also work, but give you several warnings you shouldn't ignore!!
col.sicalis <- colspace(vis.sicalis)
tetradist.sicalis.n <- coldist(col.sicalis)

tetradist.sicalis.q <- coldist(vis.sicalis, noise = 'quantum')

## End(Not run)
```

colspace *Model spectra in a colorspace*

Description

Models reflectance spectra in a colorspace. For information on plotting arguments and graphical parameters, see `plot.colspace`.

Usage

```
colspace(vismodeldata, space = c("auto", "di", "tri", "tcs", "hexagon", "coc",
  "categorical", "ciexyz", "cielab", "cielch", "segment"), qcatch = NULL)
```

Arguments

`vismodeldata`	(required) quantum catch color data. Can be either the result from `vismodel` or independently calculated data (in the form of a data frame with columns representing quantum catches).
`space`	Which colorspace/model to use. Options are: • `auto`: if data is a result from `vismodel`, applies `di`, `tri` or `tcs` if input visual model had two, three or four cones, respectively. • `di`: dichromatic colourspace. See `dispace` for details. (plotting arguments) • `tri`: trichromatic colourspace (i.e. Maxwell triangle). See `trispace` for details. (plotting arguments) • `tcs`: tetrahedral colourspace. See `tcspace` for details. (plotting arguments) • `hexagon`: the trichromatic colour-hexagon of Chittka (1992). See `hexagon` for details. (plotting arguments)

- coc: the trichromatic colour-opponent-coding model of Backhaus (1991). See coc for details. (plotting arguments)
- categorical: the tetrachromatic categorical fly-model of Troje (1993). See categorical for details. (plotting arguments)
- ciexyz: CIEXYZ space. See cie for details. (plotting arguments)
- cielab: CIELAB space. See cie for details. (plotting arguments)
- cielch: CIELCh space. See cie for details. (plotting arguments)
- segment: segment analysis of Endler (1990). See segspace for details. (plotting arguments)

qcatch	Which quantal catch metric is being inputted. Only used when input data is NOT an output from vismodel. Must be Qi, fi or Ei.

Author(s)

Rafael Maia <rm72@zips.uakron.edu>

Thomas White <thomas.white026@gmail.com>

References

Smith T, Guild J. (1932) The CIE colorimetric standards and their use. Transactions of the Optical Society, 33(3), 73-134.

Westland S, Ripamonti C, Cheung V. (2012). Computational colour science using MATLAB. John Wiley & Sons.

Chittka L. (1992). The colour hexagon: a chromaticity diagram based on photoreceptor excitations as a generalized representation of colour opponency. Journal of Comparative Physiology A, 170(5), 533-543.

Chittka L, Shmida A, Troje N, Menzel R. (1994). Ultraviolet as a component of flower reflections, and the colour perception of Hymenoptera. Vision research, 34(11), 1489-1508.

Troje N. (1993). Spectral categories in the learning behaviour of blowflies. Zeitschrift fur Naturforschung C, 48, 96-96.

Stoddard, M. C., & Prum, R. O. (2008). Evolution of avian plumage color in a tetrahedral color space: A phylogenetic analysis of new world buntings. The American Naturalist, 171(6), 755-776.

Endler, J. A., & Mielke, P. (2005). Comparing entire colour patterns as birds see them. Biological Journal Of The Linnean Society, 86(4), 405-431.

Kelber A, Vorobyev M, Osorio D. (2003). Animal colour vision - behavioural tests and physiological concepts. Biological Reviews, 78, 81 - 118.

Backhaus W. (1991). Color opponent coding in the visual system of the honeybee. Vision Research, 31, 1381-1397.

Endler, J. A. (1990) On the measurement and classification of color in studies of animal color patterns. Biological Journal of the Linnean Society, 41, 315-352.

Examples

```
## Not run:
data(flowers)

# Dichromat
vis.flowers <- vismodel(flowers, visual = 'canis')
di.flowers <- colspace(vis.flowers, space = 'di')

# Colour hexagon
vis.flowers <- vismodel(flowers, visual = 'apis', qcatch = 'Ei', relative = FALSE,
                        vonkries = TRUE, achro = 'l', bkg = 'green')
hex.flowers <- colspace(vis.flowers, space = 'hexagon')

# Trichromat
vis.flowers <- vismodel(flowers, visual = 'apis')
tri.flowers <- colspace(vis.flowers, space = 'tri')
plot(tri.flowers)

# Tetrachromat
vis.flowers <- vismodel(flowers, visual = 'bluetit')
tcs.flowers <- colspace(vis.flowers, space = 'tcs')

# Categorical
vis.flowers <- vismodel(flowers, visual = 'musca', achro = 'md.r1')
cat.flowers <- colspace(vis.flowers, space = 'categorical')

## End(Not run)
```

`explorespec` *Plot spectral curves*

Description

Plots one or multiple spectral curves in the same graph to rapidly compare groups of spectra.

Usage

```
explorespec(rspecdata, by = NULL, scale = c("equal", "free"),
  legpos = "topright", ...)
```

Arguments

`rspecdata`	(required) a data frame, possibly an object of class `rspec` that has wavelength range in the first column, named 'wl', and spectral measurements in the remaining columns.
`by`	number of spectra to include in each graph (defaults to 1)
`scale`	defines how the y-axis should be scaled. `'free'`: panels can vary in the range of the y-axis; `'equal'`: all panels have the y-axis with the same range.

`legpos`	legend position control. Either a vector containing `x` and `y` coordinates or a single keyword from the list: `"bottomright"`, `"bottom"`, `"bottomleft"`, `"left"`, `"topleft"`, `"top"`, `"topright"`, `"right"` and `"center"`.
`...`	additional parameters to be passed to plot

Value

Spectral curve plots

Note

Number of plots presented per page depends on the number of graphs produced.

Author(s)

Pierre-Paul Bitton `<bittonp@uwindsor.ca>`

Examples

```
## Not run:
data(sicalis)
explorespec(sicalis, 3)
explorespec(sicalis, 3, ylim = c(0, 100), legpos = c(500, 80))
## End(Not run)
```

`flowers` *Reflectance spectra from a suite of native Australian flowers, collected around Cairns, Queensland.*

Description

dataset containing reflectance measurements from 36 native Australian angiosperm species, indicated by column names.

Author(s)

Thomas White `<thomas.white026@gmail.com>`

getspec *Import spectra files*

Description

Finds and imports spectra files from a folder. Currently works for reflectance files generated in Ocean Optics SpectraSuite (USB2000, USB4000 and Jaz spectrometers), CRAIC software (after exporting) and Avantes (after exporting).

Usage

```
getspec(where = getwd(), ext = "txt", lim = c(300, 700), decimal = ".",
  sep = NULL, subdir = FALSE, subdir.names = FALSE, fast = FALSE,
  cores = getOption("mc.cores", 2L))
```

Arguments

`where`	(required) folder in which files are located.
`ext`	file extension to be searched for, without the "." (defaults to "txt").
`lim`	a vector with two numbers determining the wavelength limits to be considered (defaults to 300 and 700).
`decimal`	character to be used to identify decimal plates (defaults to ".").
`sep`	column delimiting characters to be considered in addition to the default (which are: tab, space, and ";")
`subdir`	should subdirectories within the `where` folder be included in the search? (defaults to `FALSE`).
`subdir.names`	should subdirectory path be included in the name of the spectra? (defaults to `FALSE`).
`fast`	logical. if `TRUE`, will try a fast algorithm that assumes all spectra were produced using the same software configuration (defaults to `FALSE`).
`cores`	Number of cores to be used. If greater than 1, import will use parallel processing (not available in Windows).

Value

A data frame, of class `rspec`, containing individual imported spectral files as columns. Reflectance values are interpolated to the nearest wavelength integer.

Author(s)

Rafael Maia `<rm72@zips.uakron.edu>`

References

Montgomerie R (2006) Analyzing colors. In: Hill G, McGraw K (eds) Bird coloration. Harvard University Press, Cambridge, pp 90-147.

Examples

```
## Not run:
getspec('examplespec/', lim = c(400, 900))
getspec('examplespec/', ext = 'ttt')
## End(Not run)
```

`irrad2flux` *Converts between irradiance and photon (quantum) flux*

Description

Some spectrometers will give illuminant values in units of irradiance (uWatt * cm^-2), but physiological models require illuminants in units of photon (quantum) flux (umol * s^-1 * m^-2). The functions `irrad2flux` and `flux2irrad` allows for easy conversion of `rspec` objects between these units.

Usage

```
irrad2flux(rspecdata)

flux2irrad(rspecdata)
```

Arguments

`rspecdata`	(required) a rspec object containing illuminant values.

Value

a converted `rspec` object.

a converted `rspec` object

Author(s)

Rafael Maia `<rm72@zips.uakron.edu>`

jnd2xyz *Convert JND distances into perceptually-corrected Cartesian coordinates*

Description

Converts a `coldist` output into Cartesian coordinates that are perceptually-corrected (i.e. Euclidean distances = JND distances)

Usage

```
jnd2xyz(coldistres, rotate = TRUE, center = c("mean", "achro"),
  ref1 = "l", ref2 = "u", axis1 = c(1, 1, 0), axis2 = c(0, 0, 1))
```

Arguments

`coldistres`	(required) the output from a `coldist` call.
`rotate`	logical indicating if the data should be rotated (defaults to `TRUE`).
`center`	should the vectors for rotation be centered in the achromatic center ("achro") or the data centroid ("mean", the default)?
`ref1`	the cone to be used as a the first reference. May be NULL (for no first rotation in the 3-dimensional case) or must match name in the original data that was used for `coldist`. Defaults to 'l'.
`ref2`	the cone to be used as a the second reference.May be NULL (for no first rotation in the 3-dimensional case) or must match name in the original data that was used for `coldist`. Defaults to 'u'. (only used if data has 3 dimensions).
`axis1`	A vector of length 3 composed of 0's and 1's, with 1's representing the axes (x,y,z) to rotate around. Defaults to c(1,1,0), such that the rotation aligns with the xy plane (only used if data has 2 or 3 dimensions). Ignored if `ref1` is NULL (in 3-dimensional case only)
`axis2`	A vector of length 3 composed of 0's and 1's, with 1's representing the axes (x,y,z) to rotate around. Defaults to c(0,0,1), such that the rotation aligns with the z axis (only used if data has 3 dimensions). Ignored if `ref2` is NULL (in 3-dimensional case only)

Author(s)

Rafael Maia `<rm72@zips.uakron.edu>`

References

Pike, T.W. (2012). Preserving perceptual distances in chromaticity diagrams. Behavioral Ecology, 23, 723-728.

Examples

```
## Not run:
data(flowers)
vis.flowers <- vismodel(flowers)
cd.flowers <- coldist(vis.flowers)
jnd2xyz(cd.flowers)

## End(Not run)
```

`legendtetra` *Add legend to a static tetrahedral colorspace*

Description

Adds a legend to a static tetrahedral colorspace plot.

Usage

```
legendtetra(x = 0.8, y = 1.2, ...)
```

Arguments

`x, y`	position of the legend relative to plot limits (usually a value between 0 and 1, but because of the perspective distortion, values greater than 1 are possible)
`...`	additional arguments passed to `legend`.

Value

`legendtetra` adds a legend to a static tetrahedral colorspace plot. for additional information on which arguments are necessary and how they are used, see `legend`.

Author(s)

Rafael Maia <`rm72@zips.uakron.edu`>

merge.rspec *Merge two rspec objects*

Description

Merges two `rspec` or `data.frame` objects into a single `rspec` object.

Usage

```
## S3 method for class 'rspec'
merge(x, y, by = "wl", ...)
```

Arguments

`x, y`	(required) two data frames (or `rspec` objects) to merge.
`by`	wavelength column name (defaults to `"wl"`).
`...`	additional class arguments.

Value

an object of class `rspec` for use with pavo functions.

Author(s)

Chad Eliason <`cme16@zips.uakron.edu`>

See Also

`as.rspec`, `aggspec`

Examples

```
## Not run:

# Load and split dataset into 2 sections
data(teal)
teal1 <- teal[, c(1, 3:5)]
teal2 <- teal[, c(1, 2, 6:12)]
teal.mer <- merge(teal1, teal2, by = 'wl')
head(teal.mer)
par(mfrow = c(1, 2))
plot(teal.mer)
plot(teal)


## End(Not run)
```

`pavo-deprecated` *Deprecated function(s) in the pavo package*

Description

These functions are provided for compatibility with older version of the pavo package. They may eventually be completely removed.

Usage

```
tcs(...)
```

Arguments

`...`	Parameters to be passed to the modern version of the function

Details

`tcs`	now a synonym for `colspace`
`segclass`	now a synonym for `vismodel(..., visual = "segment")`

`peakshape` *Peak shape descriptors*

Description

Calculates height, location and width of peak at the reflectance midpoint (FWHM). Note: bounds should be set wide enough to incorporate all minima in spectra. Smoothing spectra using `procspec` is also recommended.

Usage

```
peakshape(rspecdata, select = NULL, lim = NULL, plot = TRUE,
  ask = FALSE, absolute.min = FALSE, ...)
```

Arguments

`rspecdata`	(required) a data frame, possibly an object of class `rspec`, with a column with wavelength data, named 'wl', and the remaining column containing spectra to process.
`select`	specification of which spectra to plot. Can be a numeric vector or factor (e.g., `sex == 'male'`).

`lim`	a vector specifying the wavelength range to analyze.
`plot`	logical. Should plots indicating calculated parameters be returned? (Defaults to `TRUE`).
`ask`	logical, specifies whether user input needed to plot multiple plots when number of spectra to analyze is greater than 1 (defaults to `FALSE`).
`absolute.min`	logical. If `TRUE`, full width at half maximum will be calculated using the absolute minimum reflectance of the spectrum, even if that value falls outside the range specified by `lim`. (defaults to `FALSE`)
`...`	additional arguments to be passed to plot.

Value

a data frame containing column names (id); peak height (max value, B3), location (hue, H1) and full width at half maximum (FWHM), as well as half widths on left (HWHM.l) and right side of peak (HWHM.r). Incl.min column indicates whether user-defined bounds incorporate the actual minima of the spectra. Function will return a warning if not.

Author(s)

Chad Eliason <`cme16@zips.uakron.edu`>, Rafael Maia <`rm72@zips.uakron.edu`>

See Also

`procspec`

Examples

```
## Not run:
data(teal)
peakshape(teal, select = 3)
peakshape(teal, select = 10)

# Use wavelength bounds to narrow in on peak of interest
peakshape(teal, select = 10, lim=c(400, 550))

## End(Not run)
```

`plot.colspace` *Plot spectra in a colourspace*

Description

Plots reflectance spectra in the appropriate colorspace.

Usage

```
## S3 method for class 'colspace'
plot(x, ...)
```

Arguments

`x`	(required) an object of class `colspace`.
`...`	additional graphical options, which vary by modeled `space`. Refer to their individual documentation: • `diplot`: dichromat space • `triplot`: trichromat space • `tetraplot`: tetrahedral space • `catplot`: categorical space • `hexplot`: colour hexagon • `cocplot`: colour-opponent-coding space • `cieplot`: cie spaces • `segplot`: segment analysis space Also see `par`.

Value

A 2D colorspace plot appropriate to the input data.

Author(s)

Rafael Maia <`rm72@zips.uakron.edu`>

Thomas White <`thomas.white026@gmail.com`>

Chad Eliason <`cme16@zips.uakron.edu`>

References

Smith T, Guild J. (1932) The CIE colorimetric standards and their use. Transactions of the Optical Society, 33(3), 73-134.

Westland S, Ripamonti C, Cheung V. (2012). Computational colour science using MATLAB. John Wiley & Sons.

Chittka L. (1992). The colour hexagon: a chromaticity diagram based on photoreceptor excitations as a generalized representation of colour opponency. Journal of Comparative Physiology A, 170(5), 533-543.

Chittka L, Shmida A, Troje N, Menzel R. (1994). Ultraviolet as a component of flower reflections, and the colour perception of Hymenoptera. Vision research, 34(11), 1489-1508.

Troje N. (1993). Spectral categories in the learning behaviour of blowflies. Zeitschrift fur Naturforschung C, 48, 96-96.

Stoddard, M. C., & Prum, R. O. (2008). Evolution of avian plumage color in a tetrahedral color space: A phylogenetic analysis of new world buntings. The American Naturalist, 171(6), 755-776.

Endler, J. A., & Mielke, P. (2005). Comparing entire colour patterns as birds see them. Biological Journal Of The Linnean Society, 86(4), 405-431.

Kelber A, Vorobyev M, Osorio D. (2003). Animal colour vision - behavioural tests and physiological concepts. Biological Reviews, 78, 81 - 118.

Backhaus W. (1991). Color opponent coding in the visual system of the honeybee. Vision Research, 31, 1381-1397.

See Also

plot

Examples

```
## Not run:
data(flowers)
data(sicalis)

# Dichromat
vis.flowers <- vismodel(flowers, visual = 'canis')
di.flowers <- colspace(vis.flowers, space = 'di')
plot(di.flowers)

# Colour hexagon
vis.flowers <- vismodel(flowers, visual = 'apis', qcatch = 'Ei', relative = FALSE,
                        vonkries = TRUE, achro = 'l', bkg = 'green')
hex.flowers <- colspace(vis.flowers, space = 'hexagon')
plot(hex.flowers, sectors = 'coarse')

# Tetrahedron (static)
vis.sicalis <- vismodel(sicalis, visual = 'avg.uv')
tcs.sicalis <- colspace(vis.sicalis, space = 'tcs')
plot(tcs.sicalis)

# Tetrahedron (interactive)
vis.sicalis <- vismodel(sicalis, visual = 'avg.uv')
tcs.sicalis <- colspace(vis.sicalis, space = 'tcs')
tcsplot(tcs.sicalis, size = 0.005)

## Add points to interactive tetrahedron
patch <- rep(c('C','T','B'), 7)
tcs.crown <- subset(tcs.sicalis, 'C')
tcs.breast <- subset(tcs.sicalis, 'B')
tcsplot(tcs.crown, col ='blue')
tcspoints(tcs.breast, col ='red')

## Plot convex hull in interactive tetrahedron
tcsplot(tcs.sicalis, col = 'blue', size = 0.005)
tcsvol(tcs.sicalis)

## End(Not run)
```

---

plot.rspec *Plot spectra*

Description

Plots reflectance spectra in different arrangements.

Usage

```
## S3 method for class 'rspec'
plot(x, select = NULL, type = c("overlay", "stack",
  "heatmap"), varying = NULL, n = 100, ...)
```

Arguments

`x`	(required) a data frame, possibly an object of class `rspec`, with a column with wavelength data, named 'wl', and the remaining column containing spectra to plot.
`select`	specification of which spectra to plot. Can be a numeric vector or factor (e.g., `sex=='male'`)
`type`	what type of plot should be drawn. Possibilities are: • `overlay` (default) for plotting multiple spectra in a single panel with a common y-axis. • `stack` for plotting multiple spectra in a vertical arrangement. • `heatmap` for plotting reflectance values by wavelength and a third variable (`varying`).
`varying`	a numeric vector giving values for y-axis in `heatplot`.
`n`	number of bins with which to interpolate colors and `varying` for the heatplot.
`...`	additional arguments passed to plot (or image for `'heatmap'`).

Author(s)

Chad Eliason <`cme16@zips.uakron.edu`>

See Also

`spec2rgb`, `image`, `plot`

Examples

```
## Not run:
data(teal)
plot(teal, type = 'overlay')
plot(teal, type = 'stack')
plot(teal, type = 'heatmap')
## End(Not run)
```

plotsmooth *Plot loess smoothed curves*

Description

Plots curves with various levels of loess smoothing to help determine what loess parameters are best for the data.

Usage

```
plotsmooth(rspecdata, minsmooth = 0.05, maxsmooth = 0.2, curves = 5,
  specnum = 0, ask = TRUE)
```

Arguments

`rspecdata`	(required) a data frame, possibly of class `rspec`, which contains a column containing a wavelength range , named 'wl', and spectra data in remaining columns.
`minsmooth`	the minimum f value of the loess function to visualize (defaults to `0.05`).
`maxsmooth`	the maximum f value of the loess function to visualize (defaults to `0.20`).
`curves`	the number of curves to display on the same plot (defaults to `5`).
`specnum`	the number of spectral curves, from the data frame, to visualize (defaults to `ALL`).
`ask`	logical. if `TRUE`, asks for user input before changing plot pages

Value

Series of plot with curves processed with varying level of loess smoothing

Author(s)

Pierre-Paul Bitton <`bittonp@uwindsor.ca`>

Examples

```
## Not run:
data(sicalis)
plotsmooth(sicalis,0.05,0.1,7,6)

## End(Not run)
```

`points.colspace` *Plot points in a colorspace*

Description

Add points to a colorspace plot

Usage

```
## S3 method for class 'colspace'
points(x, ...)
```

Arguments

`x`	(required) an object of class `colspace`.
`...`	additional graphical options. See `par`.

Value

`points.colspace` adds points to a colorspace plot. When `space = 'tcs'`, it creates 3D points in a tetrahedral color space plot using functions of the package `rgl`, based on openGL capabilities.

Author(s)

Rafael Maia <`rm72@zips.uakron.edu`>

Thomas White <`thomas.white026@gmail.com`>

`procspec` *Process spectra*

Description

Applies normalization and/or smoothing to spectra for further analysis or plotting.

Usage

```
procspec(rspecdata, opt = c("none", "smooth", "maximum", "minimum", "bin",
  "sum", "center"), fixneg = c("none", "addmin", "zero"), span = 0.25,
  bins = 20, ...)
```

Arguments

`rspecdata`	(required) a data frame, possibly an object of class `rspec`, with a column with wavelength data, named 'wl', and the remaining column containing spectra to process.
`opt`	what type of processing options to apply. User can select multiple options by providing a vector. Possibilities are: • `"none"` does not perform any processing (default). • `"smooth"` applies LOESS smoothing to each spectrum using `loess.smooth`. Optimal smoothing parameter can be assessed by using `plotsmooth`. • `"minimum"` subtracts the minimum from each individual spectra. • `"maxmimum"` divides each spectrum by its maximum value. • `"sum"` divides each spectrum by summed values. • `"bin"` bins each spectrum into specified wavelength ranges. User should specify. • `"center"` centers individual spectra by subtracting mean reflectance from all values.
`fixneg`	how to handle negative values. Possibilities are: • `"none"` does not perform negative value correction (default). • `"zero"` sets all negative values to zero. • `"addmin"` adds the absolute value of the maximally negative values of each spectra to the reflectance at all other wavelengths (setting the minimum value to zero, but scaling other values accordingly).
`span`	sets the smoothing parameter used by `loess.smooth`.
`bins`	sets the number of equally sized wavelength bins for `opt = "bin"`.
`...`	ignored.

Value

A data frame of class `rspec` with the processed data.

Author(s)

Chad Eliason <`cme16@zips.uakron.edu`>

References

Cuthill, I., Bennett, A. T. D., Partridge, J. & Maier, E. 1999. Plumage reflectance and the objective assessment of avian sexual dichromatism. The American Naturalist, 153, 183-200.

Montgomerie R. 2006. Analyzing colors. In Hill, G.E, and McGraw, K.J., eds. Bird Coloration. Volume 1 Mechanisms and measurements. Harvard University Press, Cambridge, Massachusetts.

See Also

`loess.smooth`

Examples

```
## Not run:
data(teal)
plot(teal, select = 10)

# Smooth data to remove noise
teal.sm <- procspec(teal, opt = 'smooth', span = 0.25)
plot(teal.sm, select = 10)

# Normalize to max of unity
teal.max <- procspec(teal, opt = c('max'), span = 0.25)
plot(teal.max, select = 10)

## End(Not run)
```

---

projplot *Hue projection plot*

Description

Produces a 2D projection plot of points in a color space

Adds points to a tetrahedral colorspace projection

Usage

```
projplot(tcsdata, ...)

projpoints(tcsres, ...)
```

Arguments

`tcsdata`	(required) color space coordinates, possibly a result from the `tcs` function, containing values for the 'h.theta' and 'h.phi' coordinates as columns (labeled as such).
`...`	additional parameters to be passed to the plotting of data points.
`tcsres`	(required) color space coordinates, possibly a result from the `tcs` function, containing values for the 'h.theta' and 'h.phi' coordinates as columns (labeled as such).

Value

`projplot` creates a 2D plot of color points projected from the tetrahedron to its encapsulating sphere, and is ideal to visualize differences in hue.

`projpoints` creates points in a projection color space plot produced by `projplot`.

Note

`projplot` uses the Mollweide projection, and not the Robinson projection, which has been used in the past. Among other advantages, the Mollweide projection preserves area relationships within latitudes without distortion.

Author(s)

Rafael Maia <rm72@zips.uakron.edu>

References

Stoddard, M. C., & Prum, R. O. (2008). Evolution of avian plumage color in a tetrahedral color space: A phylogenetic analysis of new world buntings. The American Naturalist, 171(6), 755-776.

Endler, J. A., Westcott, D., Madden, J., & Robson, T. (2005). Animal visual systems and the evolution of color patterns: Sensory processing illuminates signal evolution. Evolution, 59(8), 1795-1818.

Examples

```
## Not run:
data(sicalis)
vis.sicalis <- vismodel(sicalis, visual = 'avg.uv')
tcs.sicalis <- colspace(vis.sicalis, space = 'tcs')
projplot(tcs.sicalis, pch = 16, col = setNames(rep(1:3, 7), rep(c('C', 'T', 'B'), 7)))

## End(Not run)
```

`segspace` *Segment classification*

Description

Calculates segment classification measures as defined in Endler (1990).

Usage

```
segspace(vismodeldata)
```

Arguments

`vismodeldata`	(required) quantum catch color data. Can be either the result from `vismodel` or independently calculated data (in the form of a data frame with columns named ’S1’, ’S2’, ’S3’, ’S4’, and, optionally, ’lum’, representing a generic ’tetrachromatic’ viewer).

Value

A data frame of class `colspace` consisting of the following columns:

`S1`, `S2`, `S3`, `S4`: the relative reflectance at each of the four segments.

`LM`, `MS`: segment scores

`C`, `H`, `B`: ’chroma’, ’hue’ (degrees), and ’brightness’ in the segment classification space

Author(s)

Thomas White `<thomas.white026@gmail.com>`

Pierre-Paul Bitton `<bittonp@uwindsor.ca>`

References

Endler, J. A. (1990) On the measurement and classification of color in studies of animal color patterns. Biological Journal of the Linnean Society, 41, 315-352.

Examples

```
## Not run:
data(sicalis)
vis.sic <- vismodel(sicalis, visual = 'segment', achromatic = 'all')
seg.sic <- colspace(vis.sic, space = 'segment')

## End(Not run)
```

`sensdata` *Retrieve or plot in-built data*

Description

Retrieve (as an rspec object) or plot pavo’s in-built spectral data.

Usage

```
sensdata(visual = c("none", "all", "avg.uv", "avg.v", "bluetit", "star",
  "pfowl", "apis", "canis", "cie2", "cie10", "musca"), achromatic = c("none",
  "all", "bt.dc", "ch.dc", "st.dc", "md.r1"), illum = c("none", "all",
  "bluesky", "D65", "forestshade"), trans = c("none", "all", "bluetit",
  "blackbird"), bkg = c("none", "all", "green"), plot = FALSE, ...)
```

Arguments

`visual` visual systems. Options are:

- `'none'`: no visual sensitivity data.
- `'all'`: all visual sensitivity data.
- `'apis'`: Honeybee *Apis mellifera* visual system.
- `'avg.uv'`: average avian UV system.
- `'avg.v'`: average avian V system.
- `'bluetit'`: Blue tit *Cyanistes caeruleus* visual system.
- `'canis'`: Canid *Canis familiaris* visual system.
- `'cie2'`: 2-degree colour matching functions for CIE models of human colour vision. Functions are linear transformations of the 2-degree cone fundamentals of Stockman & Sharpe (2000), as ratified by the CIE (2006).
- `'cie10'`: 10-degree colour matching functions for CIE models of human colour vision. Functions are linear transformations of the 10-degree cone fundamentals of Stockman & Sharpe (2000), as ratified by the CIE (2006).
- `'musca'`: Housefly *Musca domestica* visual system.
- `'pfowl'`: Peafowl *Pavo cristatus* visual system.
- `'star'`: Starling *Sturnus vulgaris* visual system.

`achromatic` the sensitivity data used to calculate luminance (achromatic) receptor stimulation. Options are:

- `'none'`: no achromatic sensitivity data.
- `'all'`: all achromatic sensitivity data.
- `'bt.dc'`: Blue tit *Cyanistes caeruleus* double cone.
- `'ch.dc'`: Chicken *Gallus gallus* double cone.
- `'st.dc'`: Starling *Sturnus vulgaris* double cone.
- `'md.r1'`: Housefly *Musca domestica* R1-6 photoreceptor.

`illum` illuminants. Options are:

- `'none'`: no illuminant data.
- `'all'`: all background spectral data.
- `'bluesky'` open blue sky.
- `'D65'`: standard daylight.
- `'forestshade'` forest shade.

`trans` Ocular transmission data. Options are:

- `'none'`: no transmission data.
- `'all'`: all transmission data.
- `'bluetit'`: blue tit *Cyanistes caeruleus* ocular transmission (from Hart et al. 2000).
- `'blackbird'`: blackbird *Turdus merula* ocular transmission (from Hart et al. 2000).

`bkg` background spectra. Options are:

- `'none'`: no background spectral data.

	• `'all'`: all background spectral data. • `'green'`: green foliage.
`plot`	should the spectral data be plotted, or returned instead (defaults to `FALSE`)?
`...`	additional graphical options passed to `plot.rspec` when `plot = TRUE`.

Value

An object of class `rspec` (when `plot = FALSE`), containing a wavelength column `'wl'` and spectral data binned at 1 nm intervals from 300-700 nm.

Author(s)

Thomas White `<thomas.white026@gmail.com>`

Rafael Maia `<rm72@zips.uakron.edu>`

Examples

```
## Not run:
# Plot the honeybee's receptors
sensdata(visual = 'apis', ylab = 'Absorbance', plot = TRUE)

# Plot the vverage UV vs V avian receptors
sensdata(visual = c('avg.v', 'avg.uv'), ylab = 'Absorbance', plot = TRUE)

# Retrieve the CIE colour matching functions as an rspec object
ciedat <- sensdata(visual = c('cie2', 'cie10'))

## End(Not run)
```

`sensmodel` *modeling spectral sensitivity*

Description

Models spectral sensitivity (with oil droplets; optional) based on peak cone sensitivity according to the models of Govardovskii et al. (2000) and Hart & Vorobyev (2005).

Usage

```
sensmodel(peaksens, range = c(300, 700), lambdacut = NULL, Bmid = NULL,
  oiltype = NULL, beta = TRUE, om = NULL, integrate = TRUE)
```

Arguments

`peaksens`	(required) a vector with peak sensitivities for the cones to model.
`range`	a vector of length 2 for the range over which to calculate the spectral sensitivities (defaults to 300nm to 700nm).
`lambdacut`	a vector of same length as peaksens that lists the cut-off wavelength value for oil droplets. Needs either `Bmid` or `oiltype` to also be entered. See Hart and Vorobyev (2005).
`Bmid`	a vector of same length as peaksens that lists the gradient of line tangent to the absorbance spectrum of the oil droplets. See Hart and Vorobyev (2005).
`oiltype`	a list of same length as peaksens that lists the oil droplet types (currently accepts only "T", C", "Y", "R", "P") when Bmid is not known. Calculates Bmid based on the regression equations found in Hart ad Vorobyev (2005).
`beta`	logical. If `TRUE` the sensitivities will include the beta peak See Govardovskii et al.(2000) (defaults to `TRUE`).
`om`	a vector of same length as `range1-range2` that contains ocular media transmission data. If included, cone sensitivity will be corrected for ocular media transmission. Currently accepts "bird" using values from Hart et al. (2005), or user-defined values.
`integrate`	logical. If `TRUE`, each curve is transformed to have a total area under the curve of 1 (best for visual models; defaults to `TRUE`). NOTE: integration is applied before any effects of ocular media are considered, for compatibility with visual model procedures.

Value

A data frame of class `rspec` containing each cone model as a column.

Author(s)

Pierre-Paul Bitton `<bittonp@uwindsor.ca>`, Chad Eliason `<cme16@zips.uakron.edu>`

References

Govardovskii VI, Fyhrquist N, Reuter T, Kuzmin DG and Donner K. 2000. In search of the visual pigment template. Visual Neuroscience 17:509-528

Hart NS, and Vorobyev M. 2005. modeling oil droplet absorption spectra and spectral sensitivities of bird cone photoreceptors. Journal of Comparative Physiology A. 191: 381-392

Hart NS, Partridge JC, Cuthill IC, Bennett AT (2000) Visual pigments, oil droplets, ocular media and cone photoreceptor distribution in two species of passerine bird: the blue tit (Parus caeruleus L) and the blackbird (Turdus merula L). J Comp Physiol A 186:375-387

Examples

```
## Not run:
# Blue tit visual system based on Hart et al (2000)
bluesens <- sensmodel(c(371,448,502,563), beta = F, lambdacut = c(330, 413, 507, 572),
oiltype = c("T", "C", "Y","R"), om = TRUE)

# Danio aequipinnatus based on Govardovskii et al. (2000)
daniosens <- sensmodel(c(357, 411, 477, 569))

## End(Not run)
```

`sicalis` *Spectral curves from three body regions of Stripe-Tailed Yellow Finch (*Sicalis citrina*) males*

Description

dataset containing reflectance measurements from 3 body parts ("C": crown, "B": breast, "T": throat) from seven male stripe-tailed yellow finches

Author(s)

Rafael Maia <`rm72@zips.uakron.edu`>

`spec2rgb` *Spectrum to rgb color conversion*

Description

Calculates rgb values from spectra based on human color matching functions.

Usage

```
spec2rgb(rspecdata, alpha = 1)
```

Arguments

`rspecdata`	(required) a data frame, possibly an object of class `rspec`, with a column with wavelength data, named 'wl', and the remaining column containing spectra to process.
`alpha`	alpha value to use for colors (defaults to 1, opaque).

Value

A character vector of class `spec2rgb` consisting of hexadecimal color values for passing to further plotting functions.

Author(s)

Chad Eliason <`cme16@zips.uakron.edu`>

References

CIE(1932). Commission Internationale de l'Eclairage Proceedings, 1931. Cambridge: Cambridge University Press.

Color matching functions obtained from Colour and Vision Research Laboratory online data repository at `http://www.cvrl.org/`.

`http://www.cs.rit.edu/~ncs/color/t_spectr.html`.

Examples

```
## Not run:
data(teal)
spec2rgb(teal)

# Plot data using estimated perceived color
plot(teal, col = spec2rgb(teal), type = 'o')
## End(Not run)
```

`subset.rspec` *Subset rspec, vismodel, and colspace objects*

Description

Subsets various object types based on a given vector or grep partial matching of data names.

Usage

```
## S3 method for class 'rspec'
subset(x, subset, ...)

## S3 method for class 'colspace'
subset(x, subset, ...)

## S3 method for class 'vismodel'
subset(x, subset, ...)
```

Arguments

`x`	(required) an object of class `rspec`, `vismodel`, or `colspace`, containing spectra, visual model output or colorspace data to subset.
`subset`	a string used for partial matching of observations.
`...`	additional attributes passed to `grep`. Ignored if `subset` is logical.

Value

a subsetted object of the same class as the input object.

Note

if more than one value is given to `subset`, any spectra that matches *either* condition will be included. It's a union, not an intersect.

Author(s)

Chad Eliason <cme16@zips.uakron.edu>

Examples

```
## Not run:
data(sicalis)
vis.sicalis <- vismodel(sicalis)
tcs.sicalis <- colspace(vis.sicalis, space = 'tcs')

# Subset all 'crown' patches (C in file names)
head(subset(sicalis, "C"))
subset(vis.sicalis, "C")
subset(tcs.sicalis, "C")[, 1:5]
subset(sicalis, c("B","C"))
subset(sicalis, "T", invert=TRUE)

## End(Not run)
```

`summary.colspace` *Colorspace data summary*

Description

Returns the attributes of `colspace` objects.

Usage

```
## S3 method for class 'colspace'
summary(object, by = NULL, ...)
```

Arguments

`object`	(required) a `colspace` object.
`by`	when the input is in `tcs` colorspace, `by` is either a single value specifying the range of color points for which summary tetrahedral-colorspace variables should be calculated (for example, `by = 3` indicates summary will be calculated for groups of 3 consecutive color points (rows) in the quantum catch color data frame) or a vector containing identifications for the rows in the quantum catch color data frame (in which case summaries will be calculated for each group of points sharing the same identification). If `by` is left blank, the summary statistics are calculated across all color points in the data.
`...`	class consistency (ignored).

Value

returns all attributes of the data as mapped to the selected colourspace, including options specified when calculating the visual model. Also return the default `data.frame` summary, except when the object is the result of `tcs`, in which case the following variables are output instead:

`centroid.u, .s, .m, .l` the centroids of `usml` coordinates of points.

`c.vol` the total volume occupied by the points.

`rel.c.vol` volume occupied by the points relative to the tetrahedron volume.

`colspan.m` the mean hue span.

`colspan.v` the variance in hue span.

`huedisp.m` the mean hue disparity.

`huedisp.v` the variance in hue disparity.

`mean.ra` mean saturation.

`max.ra` maximum saturation achieved by the group of points.

Author(s)

Rafael Maia <`rm72@zips.uakron.edu`>

References

Stoddard, M. C., & Prum, R. O. (2008). Evolution of avian plumage color in a tetrahedral color space: A phylogenetic analysis of new world buntings. The American Naturalist, 171(6), 755-776.

Endler, J. A., & Mielke, P. (2005). Comparing entire colour patterns as birds see them. Biological Journal Of The Linnean Society, 86(4), 405-431.

Examples

```
## Not run:
# Colour hexagon
data(flowers)
vis.flowers <- vismodel(flowers, visual = 'apis', qcatch = 'Ei', relative = FALSE,
                        vonkries = TRUE, bkg = 'green')
flowers.hex <- hexagon(vis.flowers)
summary(flowers.hex)

# Tetrahedral model
data(sicalis)
vis.sicalis <- vismodel(sicalis, visual='avg.uv')
csp.sicalis <- colspace(vis.sicalis)
summary(csp.sicalis, by = rep(c('C', 'T', 'B'), 7))
## End(Not run)
```

`summary.rspec` *Colorimetric variables*

Description

Calculates all 23 colorimetric variables reviewed in Montgomerie (2006).

Usage

```
## S3 method for class 'rspec'
summary(object, subset = FALSE, wlmin = NULL,
  wlmax = NULL, ...)
```

Arguments

`object`	(required) a data frame, possibly an object of class `rspec`, with a column with wavelength data, named 'wl', and the remaining column containing spectra to process.
`subset`	Either `FALSE` (the default), `TRUE`, or a character vector. If `FALSE`, all variables calculated are returned. If `TRUE`, only a subset of the complete output (composed of B2, S8 and H1; the variables described in Andersson and Prager 2006) are returned. Finally, a user-specified string of variable names can be used in order to filter and show only those variables.
`wlmin, wlmax`	minimum and maximum used to define the range of wavelengths used in calculations (default is to use entire range in the `rspec` object).
`...`	class consistency (ignored)

Value

A data frame containing either 23 or 5 (`subset = TRUE`) variables described in Montgomerie (2006) with spectra name as row names. The colorimetric variables calculated by this function are described in Montgomerie (2006) with corrections included in the README CLR file from the May 2008 distribution of the CLR software. Authors should reference both this package,Montgomerie (2006), and the original reference(s). Description and notes on the measures:

B1 (Total brightness): Sum of the relative reflectance over the entire spectral range (area under the curve). Frequently used but should be discouraged because values are difficult to compare across studies (B2 is preferred). REF 1-3, 7, 9-11, 13

B2 (Mean brightness): Mean relative reflectance over the entire spectral range. This is preferred to B1 since values are easier to compare across studies. REF 4, 12

B3 (Intensity): Maximum relative reflectance (Reflectance at wavelength of maximum reflectance). Note that may be sensitive to noise near the peak. REF 1, 5, 6

S1 (Chroma): Relative contribution of a spectral range to the total brightness (B1) S1 is arbitrarily divided in 6 measures of chroma based on the wavelength ranges normally associated with specific hues. The values are calculated using the following ranges: S1U (UV, if applicable): lambda min-400nm; S1V (Violet) lambda min-415nm; S1B (Blue) 400nm-510nm; S1G (Green) 510nm-605nm; S1Y (Yellow) 550nm-625nm; S1R (Red) 605nm-lambda max. REF 2, 7, 8, 11-13

S2 (Spectral saturation): Rmax/Rmin This measure is sensitive to spectral noise. Proper interpretation of this value may be difficult for spectra with multiple peaks in the range of interest. REF 1

S3 (Chroma): Reflectance over the Rmax +- 50nm range divided by B1. Values for peaks within 50nm of either the minimum or maximum range of the data will not be comparable since the area under the curve for the area of interest will not always be based on the same wavelength range. Therefore, S3 should be interpreted with caution for peaks in the UV or Red range. REF 11

S4 (Spectral purity): |bmaxneg| , calculated by approximating the derivative of the spectral curve. As such, it is very sensitive to noise and should only be considered when data is adequately smoothed. NAs are returned for curves which do not, at any range of wavelength, decrease in intensity. Therefore, reflectance curves for brown and red surfaces, for example, should not generate a values. REF 1

S5 (Chroma): Similar in design to segment classification measures (see Montgomerie 2006 for details). REF 10

S6 (Contrast): Rmax - Rmin. Because it uses both Rmin and Rmax, this measure may be sensitive to spectral noise. REF 5, 6

S7 (Spectral saturation): Relative reflectance between the area around the peak with reflectance equal to or larger to half of that of the peak (an approximation to the full-width at half maxima. See Montgomerie (2006) for details). Somewhat sensitive to noise and can be misleading when more than one maxima and/or minima are present. REF 3, 9

S8 (Chroma): (Rmax - Rmin)/B2. Because it uses both Rmin and Rmax, this measure may be sensitive to spectral noise. REF 3, 13

S9 (Carotenoid chroma): (R450 - R700)/R700. Should only be used when the color of the surface is clearly due to carotenoid pigmentation and R450 is lower than R700. Could be sensitive to noise. REF 8

S10 (Peaky chroma): (Rmax - Rmin)/B2 x |bmaxneg|. Should be used with properly smoothed curves. REF 7

H1 (Peak wavelength, hue): Wavelength of maximum reflectance. May be sensitive to noise and may be variable if there is more than one maxima. REF 1, 2, 4, 6, 7, 10-13

H2 (Hue): Wavelength at bmaxneg. Should be calculated using smoothed data. REF 2, 13

H3 (Hue): Wavelength at Rmid. Sensitive to noisy spectra and may be variable if there are more than one maxima and minima. REF 3, 9, 13

H4 (Hue): Similar in design to segment classification measures see Montgomerie (2006) for details. REF 10

H5 (Hue): Wavelength at bmax. Sensitive to noise and may be variable if there is more than one maxima and minima. REF 5

Note

If minimum wavelength is over 400, UV chroma is not computed.

Variables which compute bmax and bmaxneg should be used with caution, for they rely on smoothed curves to remove noise, which would otherwise result in spurious results. Make sure chosen smoothing parameters are adequate.

Smoothing affects only B3, S2, S4, S6, S10, H2, and H5 calculation. All other variables can be reliably extracted using non-smoothed data.

Author(s)

Pierre-Paul Bitton `<bittonp@windsor.ca>`, Rafael Maia `<rm72@zips.uakron.edu>`

References

Montgomerie R. 2006. Analyzing colors. In Hill, G.E, and McGraw, K.J., eds. Bird Coloration. Volume 1 Mechanisms and measurements. Harvard University Press, Cambridge, Massachusetts.

References describing variables:

1- Andersson, S. 1999. Morphology of uv reflectance in a whistling-thrush: Implications for the study of structural colour signalling in birds. Journal of Avian Biology 30:193-204.

2- Andersson, S., J. Ornborg, and M. Andersson. 1998. Ultraviolet sexual dimorphism and assortative mating in blue tits. Proceedings of the Royal Society B 265:445-450.

3- Andersson, S., S. Pryke, J. Ornborg, M. Lawes, and M. Andersson. 2002. Multiple receivers, multiple ornaments, and a trade-off between agonistic and epigamic signaling in a widowbird. American Naturalist 160:683-691.

4- Delhey, K., A. Johnsen, A. Peters, S. Andersson, and B. Kempenaers. 2003. Paternity analysis reveals opposing selection pressures on crown coloration in the blue tit (parus caeruleus). Proceedings of the Royal Society B 270:2057-2063.

5- Keyser, A. and G. Hill. 1999. Condition-dependent variation in the blue-ultraviolet coloration of a structurally based plumage ornament. Proceedings of the Royal Society B 266:771-777.

6- Keyser, A.J. and G. Hill. 2000. Structurally based plumage coloration is an honest signal of quality in male blue grosbeaks. Behavioural Ecology 11:202-209.

7- Ornborg, J., S. Andersson, S. Griffith, and B. Sheldon. 2002. Seasonal changes in a ultraviolet structural colour signal in blue tits, parus caeruleus. Biological Journal of the Linnean Society 76:237-245.

8- Peters, A., A. Denk, K. Delhey, and B. Kempenaers. 2004. Carotenoid-based bill colour as an indicator of immunocompetence and sperm performance in male mallards. Journal of Evolutionary Biology 17:1111-1120.

9- Pryke, S., M. Lawes, and S. Andersson. 2001. Agonistic carotenoid signalling in male red-collared widowbirds: Aggression related to the colour signal of both the territory owner and model intruder. Animal Behaviour 62:695-704.

10- Saks, L., K. Mcgraw, and P. Horak. 2003. How feather colour reflects its carotenoid content. Functional Ecology 17:555-561.

11- Shawkey, M., A. Estes, L. Siefferman, and G. Hill. 2003. Nanostructure predicts intraspecific variation in ultraviolet-blue plumage colour. Proceedings of the Royal Society B 270:1455-1460.

12- Siefferman, L. and G. Hill. 2005. Uv-blue structural coloration and competition for nestboxes in male eastern bluebirds. Animal Behaviour 69:67-72.

13- Smiseth, P., J. Ornborg, S. Andersson, and T. Amundsen. 2001. Is male plumage reflectance correlated with paternal care in bluethroats? Behavioural Ecology 12:164-170.

Examples

```
## Not run:
data(sicalis)
summary(sicalis)
summary(sicalis, subset = TRUE)
summary(sicalis, subset = c('B1', 'H4'))

## End(Not run)
```

`summary.vismodel` *Visual model summary*

Description

Returns the attributes used when calculating a visual model using `vismodel`

Usage

```
## S3 method for class 'vismodel'
summary(object, ...)
```

Arguments

`object`	(required) Results of `vismodel`
`...`	class consistency (ignored)

Value

Returns all attributes chosen when calculating the visual model, as well as the default `data.frame` summary

Author(s)

Rafael Maia <rm72@zips.uakron.edu>

References

Vorobyev, M., Osorio, D., Bennett, A., Marshall, N., & Cuthill, I. (1998). Tetrachromacy, oil droplets and bird plumage colours. Journal Of Comparative Physiology A-Neuroethology Sensory Neural And Behavioral Physiology, 183(5), 621-633.

Hart, N. S. (2001). The visual ecology of avian photoreceptors. Progress In Retinal And Eye Research, 20(5), 675-703.

Stoddard, M. C., & Prum, R. O. (2008). Evolution of avian plumage color in a tetrahedral color space: A phylogenetic analysis of new world buntings. The American Naturalist, 171(6), 755-776.

Endler, J. A., & Mielke, P. (2005). Comparing entire colour patterns as birds see them. Biological Journal Of The Linnean Society, 86(4), 405-431.

Examples

```
## Not run:
data(sicalis)
vis.sicalis <- vismodel(sicalis, visual='avg.uv')
summary(vis.sicalis)

## End(Not run)
```

`tcsplot` *Interactive plot of a tetrahedral colorspace*

Description

Produces an interactive 3D plot of a tetrahedral colorspace using OpenGL capabilities.

Plots points in a tetrahedral color space

Produces a 3D convex hull in tetrahedral color space

Usage

```
tcsplot(tcsdata, size = 0.02, alpha = 1, col = "black",
  vertexsize = 0.02, achro = TRUE, achrosize = 0.01, achrocol = "grey",
  lwd = 1, lcol = "lightgrey", new = FALSE, hspin = FALSE,
  vspin = FALSE, floor = TRUE, grid = TRUE, fill = TRUE)

tcspoints(tcsdata, size = 0.02, col = "black", alpha = 1)

tcsvol(tcsdata, col = "black", alpha = 0.2, grid.alpha = 1, grid = T,
  fill = T, lwd = 1)
```

Arguments

`tcsdata`	(required) a data frame, possibly a result from the `colspace` function, containing values for the 'x', 'y' and 'z' coordinates as columns (labeled as such)
`size`	size of the points in the plot (defaults to 0.02)
`alpha`	transparency of points (or volume fill in `tcsvol`)
`col`	color of the points in the plot (defaults to black)
`vertexsize`	size of the points at the vertices
`achro`	plot a point at the origin? (defaults to `TRUE`)
`achrosize`	size of the point in the achromatic center

`achrocol`	color of the point in the achromatic center
`lwd, lcol`	graphical parameters for the edges of the tetrahedron.
`new`	should a new 3D plot be called (defaults to `FALSE`)?
`hspin`	if `TRUE`, the graphic will spin horizontally (around the 'z' axis)(defaults to `FALSE`)
`vspin`	if `TRUE`, the graphic will spin vertically (around the 'x' axis)(defaults to `FALSE`)
`floor`	if `TRUE`, a reference xy plane is plotted under the tetrahedron (defaults to `TRUE`)
`grid`	if `TRUE`, connects the polygon outlining the volume occupied by points (defaults to `TRUE`)
`fill`	if `TRUE`, fills the volume occupied by points (WARNING: transparency is not saved properly if exported using `rgl.postscript`)(defaults to `TRUE`).
`grid.alpha`	transparecny of the volume polygon grid lines

Value

`tcsplot` creates a 3D plot using functions of the package `rgl`, based on openGL capabilities. Plot is interactive and can be manipulated with the mouse (left button: rotate along 'z' axis; right button: rotate along 'x' axis; third button: zoom). `tcsvol` creates polygon based on points, determining the volume occupied by them in the colorspace. `tcspoints` adds points to the plot. Points are currently plotted only as spheres to maintain export capabilities.

`tcspoints` creates 3D points in a tetrahedral color space plot produced by `tcsplot` using functions of the package `rgl`, based on openGL capabilities.

`tcsvol` creates a 3D convex hull within a `tcsplot` object.

Author(s)

Rafael Maia <`rm72@zips.uakron.edu`>

References

Stoddard, M. C., & Prum, R. O. (2008). Evolution of avian plumage color in a tetrahedral color space: A phylogenetic analysis of new world buntings. The American Naturalist, 171(6), 755-776.

Endler, J. A., & Mielke, P. (2005). Comparing entire colour patterns as birds see them. Biological Journal Of The Linnean Society, 86(4), 405-431.

See Also

`spheres3d`,`rgl.postscript`, `rgl.snapshot`,`rgl.material`

Examples

```
## Not run:
# For plotting
data(sicalis)
vis.sicalis <- vismodel(sicalis, visual = 'avg.uv')
tcs.sicalis <- colspace(vis.sicalis, space = 'tcs')
plot(tcs.sicalis, size = 0.005)
rgl.postscript('testplot.pdf',fmt='pdf')
rgl.snapshot('testplot.png')

# For adding points
patch <- rep(c('C', 'T', 'B'), 7)
tcs.crown <- subset(tcs.sicalis, 'C')
tcs.breast <- subset(tcs.sicalis, 'B')
plot(tcs.crown, col ='blue')
points(tcs.breast, col ='red')

# For plotting convex hull
plot(tcs.sicalis, col = 'blue', size = 0.005)
vol(tcs.sicalis)

## End(Not run)
```

`teal` *Angle-resolved reflectance data for the iridescent wing patch of a male green-winged teal (Anas carolinensis)*

Description

dataset containing reflectance measurements from the wing patch of a single male at different incident angles (15-75 degrees in 5-degree increments).

Author(s)

Chad Eliason <`cme16@zips.uakron.edu`>

`transmissiondata` *Default ocular transmission data*

Description

Default ocular transmission data

Author(s)

Rafael Maia <`rm72@zips.uakron.edu`>

References

Hart, N. S., Partridge, J. C., Cuthill, I. C., Bennett, A. T. D. (2000). Visual pigments, oil droplets, ocular media and cone photoreceptor distribution in two species of passerine

ttvertex *vertex for the tetrahedral color space*

Description

internal data for plotting devices.

Author(s)

Rafael Maia <rm72@zips.uakron.edu>

References

Stoddard, M. C., & Prum, R. O. (2008). Evolution of avian plumage color in a tetrahedral color space: A phylogenetic analysis of new world buntings. The American Naturalist, 171(6), 755-776.

vismodel *Visual models*

Description

Calculates quantum catches at each photoreceptor. Both raw and relative values can be returned, for use in a suite of colorspace and non-colorspace models.

Usage

```
vismodel(rspecdata, visual = c("avg.uv", "avg.v", "bluetit", "star", "pfowl",
  "apis", "canis", "cie2", "cie10", "musca", "segment"),
  achromatic = c("none", "bt.dc", "ch.dc", "st.dc", "ml", "l", "md.r1",
  "all"), illum = c("ideal", "bluesky", "D65", "forestshade"),
  trans = c("ideal", "bluetit", "blackbird"), qcatch = c("Qi", "fi", "Ei"),
  bkg = c("ideal", "green"), vonkries = FALSE, scale = 1,
  relative = TRUE)
```

Arguments

`rspecdata`	(required) a data frame, possibly an object of class `rspec` that has wavelength range in the first column, named 'wl', and spectral measurements in the remaining columns.
`visual`	the visual system to be used. Options are:

- a data frame such as one produced containing by `sensmodel`, containing user-defined sensitivity data for the receptors involved in colour vision. The data frame must contain a `'wl'` column with the range of wavelengths included, and the sensitivity for each other cone as a column.
- `'apis'`: Honeybee *Apis mellifera* visual system.

- `'avg.uv'`: average avian UV system.
- `'avg.v'`: average avian V system.
- `'bluetit'`: Blue tit *Cyanistes caeruleus* visual system.
- `'canis'`: Canid *Canis familiaris* visual system.
- `'cie2'`: 2-degree colour matching functions for CIE models of human colour vision. Functions are linear transformations of the 2-degree cone fundamentals of Stockman & Sharpe (2000), as ratified by the CIE (2006).
- `'cie10'`: 10-degree colour matching functions for CIE models of human colour vision. Functions are linear transformations of the 10-degree cone fundamentals of Stockman & Sharpe (2000), as ratified by the CIE (2006).
- `'musca'`: Housefly *Musca domestica* visual system.
- `'pfowl'`: Peafowl *Pavo cristatus* visual system.
- `'segment'`: Generic tetrachromat 'viewer' for use in the segment analysis of Endler (1990).
- `'star'`: Starling *Sturnus vulgaris* visual system.

`achromatic` the sensitivity data to be used to calculate luminance (achromatic) receptor stimulation. Either a vector containing the sensitivity for a single receptor, or one of the options:

- `'none'`: no achromatic stimulation calculated
- `'bt.dc'`: Blue tit *Cyanistes caeruleus* double cone
- `'ch.dc'`: Chicken *Gallus gallus* double cone
- `'st.dc'`: Starling *Sturnus vulgaris* double cone
- `'md.r1'`: Housefly *Musca domestica* R1-6 photoreceptor
- `'ml'`: the summed response of the two longest-wavelength photoreceptors
- `'l'`: the longest-wavelength photoreceptor
- `'all'`: the summed response of all photoreceptors

`illum` either a vector containing the illuminant, or one of the options:

- `'ideal'`: homogeneous illuminance of 1 across wavelengths (default)
- `'bluesky'` open blue sky.
- `'D65'`: standard daylight.
- `'forestshade'` forest shade.

`trans` either a vector containing the ocular or environmental transmission spectra, or one of the options:

- `'ideal'`: homogeneous transmission of 1 across all wavelengths (default)
- `'bluetit'`: blue tit *Cyanistes caeruleus* ocular transmission (from Hart et al. 2000).
- `'blackbird'`: blackbird *Turdus merula* ocular transmission (from Hart et al. 2000).

`qcatch` Which quantal catch metric to return. Options are:

- `'Qi'`: Quantum catch for each photoreceptor
- `'fi'`: Quantum catch according to Fechner law (the signal of the receptor channel is proportional to the logarithm of the quantum catch)

	• `'Ei'`: Hyperbolic-transformed quantum catch, where Ei = Qi / (Qi + 1).
`bkg`	background spectrum. Note that this will have no effect when `vonkries = FALSE`. Either a vector containing the spectral data, or one of the options: • `'ideal'`: homogeneous illuminance of 1 across all wavelengths (default). • `'green'`: green foliage.
`vonkries`	logical. Should the von Kries color correction transformation be applied? (defaults to `FALSE`).
`scale`	a value by which the illuminant will be multiplied. Useful for when the illuminant is a relative value (i.e. transformed to a maximum of 1 or to a percentage), and does not correspond to quantum flux units ($umol*s^-1*m^-2$). Useful values are, for example, 500 (for dim light) and 10000 (for bright illumination). Note that if `vonkries = TRUE` this transformation has no effect.
`relative`	should relative quantum catches be returned (i.e. is it a color space model? Defaults to `TRUE`).

Value

An object of class `vismodel` containing the photon catches for each of the photoreceptors considered. Information on the parameters used in the calculation are also stored and can be called using the `summary.vismodel` function.

Author(s)

Rafael Maia <`rm72@zips.uakron.edu`>

Thomas White <`thomas.white026@gmail.com`>

References

Vorobyev, M., Osorio, D., Bennett, A., Marshall, N., & Cuthill, I. (1998). Tetrachromacy, oil droplets and bird plumage colours. Journal Of Comparative Physiology A-Neuroethology Sensory Neural And Behavioral Physiology, 183(5), 621-633.

Hart, N. S., Partridge, J. C., Cuthill, I. C., Bennett, A. T. D. (2000). Visual pigments, oil droplets, ocular media and cone photoreceptor distribution in two species of passerine bird: the blue tit (Parus caeruleus L.) and the blackbird (Turdus merula L.). Journal of Comparative Physiology A, 186, 375-387.

Hart, N. S. (2001). The visual ecology of avian photoreceptors. Progress In Retinal And Eye Research, 20(5), 675-703.

Stoddard, M. C., & Prum, R. O. (2008). Evolution of avian plumage color in a tetrahedral color space: A phylogenetic analysis of new world buntings. The American Naturalist, 171(6), 755-776.

Endler, J. A., & Mielke, P. (2005). Comparing entire colour patterns as birds see them. Biological Journal Of The Linnean Society, 86(4), 405-431.

Chittka L. (1992). The colour hexagon: a chromaticity diagram based on photoreceptor excitations as a generalized representation of colour opponency. Journal of Comparative Physiology A, 170(5), 533-543.

Stockman, A., & Sharpe, L. T. (2000). Spectral sensitivities of the middle- and long-wavelength sensitive cones derived from measurements in observers of known genotype. Vision Research, 40, 1711-1737.

CIE (2006). Fundamental chromaticity diagram with physiological axes. Parts 1 and 2. Technical Report 170-1. Vienna: Central Bureau of the Commission Internationale de l' Eclairage.

Examples

```
## Not run:
# Dichromat (dingo)
data(flowers)
vis.flowers <- vismodel(flowers, visual = 'canis')
di.flowers <- colspace(vis.flowers, space = 'di')

# Trichromat (honeybee)
data(flowers)
vis.flowers <- vismodel(flowers, visual = 'apis')
tri.flowers <- colspace(vis.flowers, space = 'tri')

# Tetrachromat (blue tit)
data(sicalis)
vis.sicalis <- vismodel(sicalis, visual = 'bluetit')
tcs.sicalis <- colspace(vis.sicalis, space = 'tcs')

## End(Not run)
```

`vissyst` *Animal visual systems data*

Description

Internal data for visual model calculations.

Author(s)

Rafael Maia <rm72@zips.uakron.edu>

References

Endler, J. A., & Mielke, P. (2005). Comparing entire colour patterns as birds see them. Biological Journal Of The Linnean Society, 86(4), 405-431.

vol *Plot a tetrahedral color space*

Description

Produces a 3D convex hull in tetrahedral color space when plotting a non-interactive tetrahedral plot.

Usage

```
vol(tcsdata, alpha = 0.2, grid = TRUE, fill = TRUE, new = FALSE, view,
  scale.y, axis, ...)
```

Arguments

`tcsdata`	(required) object of class `colspace`.
`alpha`	transparency of volume (if `fill = TRUE`).
`grid`	logical. if `TRUE` (default), draws the polygon outline defined by the points.
`fill`	logical. if `TRUE` (default), fills the volume defined by the points.
`new`	logical. Should a new plot be started or draw over an open plot? (defaults to FALSE)
`view, scale.y, axis`	deprecated arguments.
`...`	additional graphical options. See `link{polygon}` and `tetraplot`.

Value

`vol` creates a 3D convex hull within a static tetrahedral plot.

Author(s)

Rafael Maia <`rm72@zips.uakron.edu`>

voloverlap *Color volume overlap*

Description

Calculates the overlap between the volumes defined by two sets of points in cartesian space.

Usage

```
voloverlap(tcsres1, tcsres2, plot = FALSE, interactive = FALSE,
  col = c("blue", "red", "darkgrey"), new = TRUE, montecarlo = FALSE,
  nsamp = 1000, psize = 0.001, lwd = 1, ...)
```

Arguments

`tcsres1, tcsres2`	(required) data frame, possibly a result from the `colspace` function, containing values for the 'x', 'y' and 'z' coordinates as columns (labeled as such)
`plot`	logical. Should the volumes and points be plotted? (defaults to `FALSE`)
`interactive`	logical. If `TRUE`, uses the rgl engine for interactive plotting; if `FALSE` then a static plot is generated.
`col`	a vector of length 3 with the colors for (in order) the first volume, the second volume, and the overlap.
`new`	logical. Should a new plot window be called? If `FALSE`, volumes and their overlap are plotted over the current plot (defaults to `TRUE`).
`montecarlo`	logical. If `TRUE`, Monte Carlo simulation is used instead of exact solution (not recommended; defaults to `FALSE`)
`nsamp`	if `montecarlo = TRUE`, determines the number of points to be sampled.
`psize`	if `montecarlo = TRUE` and `plot = TRUE`, sets the size to plot the points used in the Monte Carlo simulation.
`lwd`	if `plot = TRUE`, sets the line width for volume grids.
`...`	additional arguments passed to the plot. See `vol`

Value

Calculates the overlap between the volumes defined by two set of points in colorspace. The volume from the overlap is then given relative to:

- `vsmallest` the volume of the overlap divided by the smallest of that defined by the the two input sets of color points. Thus, if one of the volumes is entirely contained within the other, this overlap will be `vsmallest = 1`.
- `vboth` the volume of the overlap divided by the combined volume of both input sets of color points.

The Monte Carlo solution is available mostly for legacy and benchmarking, and is not recommended (see notes). If used, the output will be different:

- `s_in1, s_in2` the number of sampled points that fall within each of the volumes individually.
- `s_inboth` the number of sampled points that fall within both volumes.
- `s_ineither` the number of points that fall within either of the volumes.
- `psmallest` the proportion of points that fall within both volumes divided by the number of points that fall within the smallest volume.
- `pboth` the proportion of points that fall within both volumes divided by the total number of points that fall within both volumes.

If the Monte Carlo solution is used, a number of points much greater than the default should be considered (Stoddard & Stevens(2011) use around 750,000 points, but more or fewer might be required depending on the degree of overlap).

Note

Stoddard & Stevens (2011) originally obtained the volume overlap through Monte Carlo simulations of points within the range of the volumes, and obtaining the frequency of simulated values that fall inside the volumes defined by both sets of color points.

Here we present an exact solution based on finding common vertices to both volumes and calculating its volume. However, we also the Monte Carlo solution is available through the `montecarlo=TRUE` option.

Stoddard & Stevens (2011) also return the value of the overlap relative to one of the volumes (in that case, the host species). However, for other applications this value may not be what one expects to obtain if (1) the two volumes differ considerably in size, or (2) one of the volumes is entirely contained within the other. For this reason, we also report the volume relative to the union of the two input volumes, which may be more adequate in most cases.

Author(s)

Rafael Maia `<rm72@zips.uakron.edu>`, with code from Sebastien Villeger

References

Stoddard, M. C., & Prum, R. O. (2008). Evolution of avian plumage color in a tetrahedral color space: A phylogenetic analysis of new world buntings. The American Naturalist, 171(6), 755-776.

Stoddard, M. C., & Stevens, M. (2011). Avian vision and the evolution of egg color mimicry in the common cuckoo. Evolution, 65(7), 2004-2013.

Villeger, S., Novack-Gottshall, P. M., & Mouillot, D. (2011). The multidimensionality of the niche reveals functional diversity changes in benthic marine biotas across geological time. Ecology Letters, 14(6), 561-568.

Examples

```
## Not run:
data(sicalis)
tcs.sicalis.C <- subset(colspace(vismodel(sicalis)), 'C')
tcs.sicalis.T <- subset(colspace(vismodel(sicalis)), 'T')
tcs.sicalis.B <- subset(colspace(vismodel(sicalis)), 'B')
voloverlap(tcs.sicalis.T, tcs.sicalis.B)
voloverlap(tcs.sicalis.T, tcs.sicalis.C, plot = T)
voloverlap(tcs.sicalis.T, tcs.sicalis.C, plot = T, col = 1:3)
## End(Not run)
```

Index