Package ‘visualFields’

February 26, 2022

Type Package

Title Statistical Methods for Visual Fields

Version 1.0.1

Date 2022-02-15

Maintainer Ivan Marin-Franch <imarinfr@optocom.es>

Description A collection of tools for analyzing the field of vision. It provides a framework for development and use of innovative methods for visualization, statistical analysis, and clinical interpretation of visual-field loss and its change over time. It is intended to be a tool for collaborative research. The package is described in Marin-Franch and Swanson (2013) <doi:10.1167/13.4.10> and is part of the Open Perimetry Initiative (OPI) [Turpin, Artes, and McKendrick (2012) <doi:10.1167/12.11.22>].

License GPL-3

URL https://www.optocom.es

Depends R(>= 3.5)

Imports Hmisc, dplyr, polyclip, deldir, plotrix, gtools, combinat, XML, oro.dicom, rlang, shiny, shinyjs, DT, htmlTable, boot, pracma

LazyData true

LazyDataCompression xz

Encoding UTF-8

RoxygenNote 7.1.2

NeedsCompilation no

Author Ivan Marin-Franch [cre, aut, rev, cph],
William H Swanson [fnd, dtc],
Michael Wall [fnd, dtc],
Andrew Turpin [ctb],
Paul H Artes [ctb, dtc],
Cord Huchzermeyer [ctb],
Giovanni Montesano [ctb],
Mitchell W Dul [dtc]

Repository CRAN

Date/Publication 2022-02-26 17:50:02 UTC

R topics documented:

`visualFields-package` *visualFields: statistical methods for visual fields*

Description

visualFields is a collection of tools for analyzing the field of vision. It provides a framework for development and use of innovative methods for visualization, statistical analysis, and clinical interpretation of visual-field loss and its change over time. It is intended to be a tool for collaborative research.

Details

The development version of visualFields 1.x, can be found in https://github.com/imarinfr/vf1. For developers who want to collaborate extending, updating, and patching visualFields, all necessary imports are to be added to the source file `visualFields.R`. visualField developers can use the source codes here as examples on how to craft new source code and keep documentation that is consistent with the rest of the package, roxygen2, and CRAN.

The previous version of visualFields, 0.6, is still available for use in https://github.com/imarinfr/vf0, but is no longer maintained.

This work was supported by the NIH grant number **R01EY007716** and the Veterans Administration grant number **I01 RX-001821-01A1**.

Author(s)

Maintainer: Ivan Marin-Franch <imarinfr@optocom.es> [reviewer, copyright holder]

Other contributors:

- William H Swanson [funder, data contributor]
- Michael Wall [funder, data contributor]
- Andrew Turpin [contributor]
- Paul H Artes [contributor, data contributor]
- Cord Huchzermeyer [contributor]
- Giovanni Montesano [contributor]
- Mitchell W Dul [data contributor]

References

Marín-Franch I & Swanson WH. *The visualFields package: A tool for analysis and visualization of visual fields*. Journal of Vision, 2013, 13(4):10, 1-12

Turpin A, Artes PH, & McKendrick AM. *The Open Perimetry Interface: An enabling tool for clinical visual psychophysics*. Journal of Vision, 2012, 12(11):22, 21–25

See Also

`OPI`: the Open Perimetry Initiative https://people.eng.unimelb.edu.au/aturpin/opi/index.html `cart2jpolar` *The Jansonius map for average path of nerve fiber bundles*

Description

Generates a function that renders the average path of a nerve fiber bundle that exits through the optic nerve head (ONH) with a particular angle

Usage

```
cart2jpolar(coord)

jpolar2cart(rpsi)

bundlePath(psi0, r0 = 4)

loc2psi(coord, r0 = 4)

psi2oct(psi0, diam = 12)

vf2gc(coord, angle = 0)
```

Arguments

`coord`	coordinates of locations in the visual field
`rpsi`	visual field locations in polar coordinates of the distorted space of the Jansonius map
`psi0`	angle of incidence at the ONH
`r0`	radius of the ONH. Its default value is `4`. Changing it changes the calculated average bundle paths.
`diam`	diamater in degrees of visual angle of the OCT circular scan centered at the center of the ONH
`angle`	fovea-disc angle in degrees

Details

- `cart2jpolar` converts the cartesian coordinates to the polar coordinates in the distorted space used in the Jansonius map
- `jpolar2cart` converts back from the Jansonius polar coordinates to cartesian coordinates
- `bundlePath` returns a function describing the expected fiber path given an angle of incidence on the ONH
- `loc2psi` returns the angle of incidence of the average bundle path that passes through specific locations of the visual field
- `psi2oct` returns the angle of OCT circular scans corresponding to average bundle paths with specific angle of incidence at the ONH
- `vf2gc` calculates ganglion-cell soma locations

Value

`cart2jpolar`: returns the Jansonius modified polar coordinates

`jpolar2cart`: returns Cartesian coordinates

`bundlePath`: returns a function describing a retinal ganglion cell bundle path

`loc2psi`: returns the angle of incidence on the ONH

`psi2oct`: returns the corresponding angle in the OCT circular scan

`vf2gc`: returns the ganglion cell soma corresponding to the photoreceptors of a visual field location

References

N. M. Jansonius, J. Nevalainen, B. Selig, L. M. Zangwill, P. A. Sample, W. M. Budde, J. B. Jonas, W. A. Lagreze, P. J. Airaksinen, R. Vonthein, L. A. Levin, J. Paetzold, and U. Schiefer. *A mathematical description of nerve fiber bundle trajectories and their variability in the human retina*. Vision Research, 49(17):2157-2163, 2009

N. M. Jansonius, J. Nevalainen, B. Selig, L. M. Zangwill, P. A. Sample, W. M. Budde, J. B. Jonas, W. A. Lagreze, P. J. Airaksinen, R. Vonthein, L. A. Levin, J. Paetzold, and U. Schiefer. *Erratum to "A mathematical description of nerve fiber bundle trajectories and their variability in the human retina"*. Vision Research, 50:1501, 2010

N. M. Jansonius, J. Schiefer, J. Nevalainen, J. Paetzold, and U. Schiefer. *A mathematical model for describing the retinal nerve fiber bundle trajectories in the human eye: Average course, variability, and influence of refraction, optic disc size and optic disc position*. Experimental Eye Research, 105:70-78, 2012

N. Drasdo, C. L. Millican, C. R. Katholi, and C. A. Curcio. *The length of Henle fibers in the human retina and a model of ganglion receptive field density in the visual field*. Vision Research, 47:2901–2911, 2007

D. C. Hood, A. S. Raza, D. M. C. G. V., J. G. Odel, V. C. Greenstein, J. M. Liebmann, and R. Ritch. *Initial arcuate defects within the central 10 degrees in glaucoma*. Investigative Ophthalmology and Visual Science, 52(2):940-946, 2011

A. S. Raza, J. Cho, D. M. C. G. V., H. Wang, X. Zhang, R. H. Kardon, J. M. Liebmann, R. Ritch, and D. C. Hood. *Retinal ganglion cell layer thickness and local visual field sensitivity in glaucoma*. Archives of Ophthalmology, 129(12):1529-1536, 2011

G. Montesano, G. Ometto, R. E. Hogg, L. M. Rossetti, D. F. Garway-Heath, and D. P. Crabb. *Revisiting the Drasdo Model: Implications for Structure-Function Analysis of the Macular Region*. Translational Vision Science and Technology, 9(10):15, 2020

Examples

```
# get ganglion-cell soma locations from visual field locations
vf2gc(locmaps$p10d2$coord)
# convert to polar of the distorted space used by Jansonius map and back
coord <- data.frame(x = c(3, 0, -3), y = c(0, 0, 0))
(rpsi <- cart2jpolar(coord))
jpolar2cart(rpsi)

# get an average bundle path from a specific angle of incidence in the ONH
# The object returned is a function that returns polar angles of the
```

```
# distorted space of the Jansonius map for distances from the ONH center
pathFun <- bundlePath(-125)
jpolar2cart(data.frame(10:20, pathFun(10:20)))

# get angle of incidence in the ONH from locations of the visual field
loc2psi(coord)

# get the OCT circular scan angles from the angle of incidence in the ONH
# for the 10-2 map of locations, ...
psi2oct(loc2psi(locmaps$p10d2$coord))
# the previous operation was actually fundamentally wrong! We need to
# obtain first the
psi2oct(loc2psi(vf2gc(locmaps$p10d2$coord)))
```

`drasdolut` *Precomputed X and Y displacement of ganglion cell bodies for any given X and Y location on the retina*

Description

It contains a first list with two LUTs for the X and Y displacement of ganglion cell bodies for arbitrary locations in the retina (in mm assuming 24 mm axial length). The other two elements of the list contain precomputed vectors of degrees and mm on the retina for the same schematic eye, used for conversions. These are used by the function vf2gc().

Usage

```
drasdolut
```

Format

A large list containing

Drasdo_LUT a list of four elements: xlut and ylut are 2d matrices containing X and Y ganglion cell positions for any given location. Xv and Yv are vectors defining the corresponding locations for the matrices along the X and Y axis.

Degs A vector of degrees from the fovea, using a schematic eye. Corresponds to distances on the retina stored in MM

MM A vector of MM distance from the fovea, using a schematic eye. Corresponds to distances in degrees stored in Degs

References

G. Montesano, G. Ometto, R. E. Hogg, L. M. Rossetti, D. F. Garway-Heath, and D. P. Crabb. *Revisiting the Drasdo Model: Implications for Structure-Function Analysis of the Macular Region*. Translational Vision Science and Technology, 9(10):15, 2020

N. Drasdo, C. L. Millican, C. R. Katholi, and C. A. Curcio. *The length of Henle fibers in the human retina and a model of ganglion receptive field density in the visual field*. Vision Research, 47:2901–2911, 2007

getage — *Calculates age*

Description

Computes ages at specific dates

Usage

```
getage(dob, date)
```

Arguments

`dob`	date(s) of birth
`date`	date(s) for which to calculate age

Value

`getage` returns the age from the date of birth and a certain date

Examples

```
getage("1977-01-31", "2014-01-30")
```

glr — *Global and pointwise linear regression analyses*

Description

Functions that compute global and pointwise linear regression analyses:

- `glr` performs global linear regression analysis
- `plr` performs pointwise linear regression (PLR) analysis
- `poplr` performs PoPLR analysis as in O'Leary et al (see reference)

Usage

```
glr(g, type = "md", testSlope = 0)

plr(vf, type = "td", testSlope = 0)

poplr(vf, type = "td", testSlope = 0, nperm = factorial(7), trunc = 1)
```

Arguments

- `g` global indices
- `type` type of analysis. For `glr`, it can be ‘`ms`‘, ‘`ss`‘, ‘`md`‘, ‘`sd`‘, ‘`pmd`‘, ‘`psd`‘, ‘`vfi`‘, or ‘`gh`‘ for mean sensitivity, standard deviation of sensitivities, mean deviation, standard deviation of total deviation values, pattern mean deviation, pattern standard deviation, VFI, and general height, respectively. For `plr` and `poplr`, it can be ‘`s`‘, ‘`td`‘, or ‘`pd`‘ for sensitivities, total deviation values, or pattern deviation values, respectively
- `testSlope` slope, or slopes, to test as null hypothesis. Default is 0. if a single value, then the same null hypothesis is used for all locations. If a vector of values, then (for `plr` and `poplr`) each location of the visual field will have a different null hypothesis. The length of testSlope must be 1 or equal to the number of locations to be used in the PLR or PoPLR analysis
- `vf` visual fields sensitivity data
- `nperm` number of permutations. If the number of visits is 7 or less, then `nperm = factorial(nrow(vf))`. For series greater than 8 visits, default is factorial(7). For series up to 7 visits, it is the factorial of the number of visits (with less than 7 visits, the number of possible permutations is small and results can be unreliable. For instance, for 5 visits, the number of possible permutations is only 120.)
- `trunc` truncation value for the Truncated Product Method (see reference)

Details

- `poplr` there is a small difference between this implementation of PoPLR and that proposed by O’Leary et al. The combined S statistic in the paper used a natural logarithm. Here we not only use a logarithm of base 10 but we also divide by the number of locations. This way the S statistic has a more direct interpretation as the average number of leading zeros in the p-values for pointwise (simple) linear regression. That is, if S = 2, then the p-values have on average 2 leading zeros, if S = 3, then 3 leading zeros, and so on

Value

- `glr` and `plr` return a list with the following
  - `id` patient ID
  - `eye` patient eye
  - `type` type of data analysis. . For `glr`, it can be ‘`ms`‘, ‘`ss`‘, ‘`md`‘, ‘`sd`‘, ‘`pmd`‘, ‘`psd`‘, ‘`vfi`‘, or ‘`gh`‘ for mean sensitivity, standard deviation of sensitivities, mean deviation, standard deviation of total deviation values, pattern mean deviation, pattern standard deviation, VFI, and general height, respectively. For `plr` and `poplr`, it can be ‘`s`‘, ‘`td`‘, or ‘`pd`‘ for sensitivities, total deviation values, or pattern deviation values, respectively
  - `testSlope` slope for `glr` or list of slopes for `plr` to test as null hypotheses
  - `nvisits` number of visits
  - `years` years from baseline. Used for the pointwise linear regression analysis

  - `data` data analyzed. For `glr`, it is the values of the global indes analyzed. For `plr`, each column is a location of the visual field used for the analysis. Each row is a visit (as many as years)
  - `pred` predicted values. Each column is a location of the visual field used for the analysis. Each row is a visit (as many as years)
  - `sl` slopes estimated at each location for pointwise (simple) linear regression
  - `int` intercept estimated at each location for pointwise (simple) linear regression
  - `tval` t-values obtained for the left-tailed-t-tests for the slopes obtained in the pointwise (simple) linear regression at each location
  - `pval` p-values obtained for the left-tailed t-tests for the slopes obtained
- `poplr` returns a list with the following additional fields
  - `csl` the modifed Fisher's S-statistic for the left-tailed permutation test
  - `cslp` the p-value for the left-tailed permutation test
  - `csr` the modifed Fisher's S-statistic for the right-tailed permutation test
  - `csrp` the p-value for the right-tailed permutation test
  - `pstats` a list with the poinwise slopes (‘`sl`‘), intercepts (‘`int`‘), standard errors (‘`se`‘), and p-values (‘`pval`‘) obtained for the series at each location analyzed and for all `nperm` permutations (in ‘`permutations`‘)
  - `cstats` a list with all combined stats:
    * `csl,csr` the combined Fisher S-statistics for the left- and right-tailed permutation tests respectively
    * `cslp,csrp` the corresponding p-values for the permutation tests
    * `cslall,csrall` the combined Fisher S-statistics for all permutations

References

N. O'Leary, B. C. Chauhan, and P. H. Artes. *Visual field progression in glaucoma: estimating the overall significance of deterioration with permutation analyses of pointwise linear regression (PoPLR)*. Investigative Ophthalmology and Visual Science, 53, 2012

Examples

```
vf <- vffilter(vfpwgRetest24d2, id == 1) # select one patient
res <- glr(getgl(vf)) # linear regression with global indices
res <- plr(vf)   # pointwise linear regression (PLR) with TD values
res <- poplr(vf) # Permutation of PLR with TD values
```

---

gpars *List of graphical parameters*

Description

It contains a list of normative values, including pointwise and smoothed SUNY-IU normative reference values for 24-2 static automated perimetry (`sunyiu_24d2_pw` and `sunyiu_24d2`) obtained with the dataset `vfctrSunyiu24d2`

Usage

```
gpars
```

Format

See section `Structure of graphical parameters` in `vfplot`

`loadhfaxml` *Loaders from perimeters*

Description

Functions to load from commercial perimeters

Usage

```
loadhfaxml(file, type = "pwg", repeated = mean)

loadhfadicom(file, type = "pwg", repeated = mean)

loadoctopus(file, type = "pwg", repeated = mean, dateFormat = "%d.%m.%Y")

loadhfaxmlbatch(file, repeated = mean)

loadhfadicombatch(file, repeated = mean)
```

Arguments

`file`	name of the csv file exported by the eyesuite software
`type`	type of patient. It can be ‘`ctr`‘ (for control or healthy subject-eye) or ‘`pwg`‘ (for patient with glaucoma) or other
`repeated`	function to apply if there are repeated values in a particular location
`dateFormat`	format to be used for date. Its default value is %d.%m.%Y

Details

The XML loader for the Humphrery Field Analyser (HFA) by Carl Zeiss Meditec is essentially a XML parser that reads in the XML generated with the scientific export license. The DICOMM loader is also a parser to read HFA data generated in a DICOMM file. The loader for the Octopus perimeter by Haag-Streit is a csv reader from files generated with the Eyesuite software. The parser also extracts information on visual field pattern deviation values and normative values. The list that is returned with the `loadoctopus` loader contains data frames which are structured with keys so that redundancy is minimized (similar to a relational database). Detailed examples for `loadoctopus`: https://rpubs.com/huchzi/645357

Value

Visual field data

`locmaps` *Location maps*

Description

List of common and some custom location maps, including the 24-2, 10-2, 30-2, 60-4, etc used the the HFA and Octopus, the 24-2 used by the Matrix (FDT), and others used in Swanson's and Wall's labs

Usage

```
locmaps
```

Format

See section `Structure of location maps` in `setlocmap`

`locread` *Locmap management*

Description

Functions to handle location maps, which are lists with x and y coordinates and other importan information about the visual field test locations. Check section `Structure of location maps` below for details

Usage

```
locread(file, name = "", desc = "", bs = numeric(), ...)

locwrite(locmap, file, ...)
```

Arguments

`file`	the name of the file which the data are to be read from
`name`	to give the location map
`desc`	brief description for the location map
`bs`	locations that should be excluded from statistical analysis because of their proximity to the blind spot
`...`	arguments to be passed to or from methods
`locmap`	location map from which to get coordinates to export as csv file

Details

- `locread` reads a csv file with location map data
- `locwrite` writes a csv file with location map data

Value

`locread` a list with information about a location map

`locwrite` No return value

Structure of location maps

Each element in the list `locmaps` is a location map that contains the following fields

- `name` descriptive name
- `desc` brief description
- `coord` coordinates of the visual field locations
- `bs` if not empty, the locations that ought to be removed for statistical analysis due to their proximity to the blind spot

Examples

```
# write and read location map
tf <- tempfile("locmap")
locwrite(getlocmap(), file = tf) # save current locmap in a temp file
print(locread(tf, name = "name", desc = "desc", bs = c(1, 2))) # read the temp file
```

`normvals` *List of normative values that can be used for statistical analysis and visualization*

Description

It contains a list of normative values, including pointwise and smoothed SUNY-IU normative reference values for 24-2 static automated perimetry (`sunyiu_24d2_pw` and `sunyiu_24d2`) obtained with the dataset `vfctrSunyiu24d2`

Usage

```
normvals
```

Format

See section `Structure of normative values` in `setnv`

References

H. J. Wyatt, M. W. Dul, and W. H. Swanson. *Variability of visual field measurements is correlated with the gradient of visual sensitivity*. Vision Research, 47, 2007.

A. Shafi, W. H. Swanson, and M. W. Dul. *Structure and Function in Patients with Glaucomatous Defects Near Fixation*. Optometry and Vision Science, 88, 2011.

---

nvgenerate — *Normative values generation and management*

Description

Functions to generate and handle normative values. Check section `Structure of normative values` below for details about how to generate functioning normative values

Usage

```
nvgenerate(
  vf,
  method = "pointwise",
  probs = c(0, 0.005, 0.01, 0.02, 0.05, 0.95, 0.98, 0.99, 0.995, 1),
  name = "",
  perimetry = "static automated perimetry",
  strategy = "",
  size = ""
)

agelm(vf)

tddef(agem)

ghdef(perc = 0.85)

pddef(ghfun = ghdef(0.85))

lutdef(vf, probs, type = "quantile", ...)

gdef(agem, sdtd, sdpd)

lutgdef(g, probs, type = "quantile", ...)
```

Arguments

`vf`	visual field data with sensitivity values
`method`	method to generate normative values, pointwise (‘`pointwise`‘) or smoothed with 2-dimensional quadratic functions (‘`smooth`‘)

probs	numeric vector of probabilities with values in [0,1]. The values 0 and 1 must be included
name	name for the normative values, e.g., "SUNY-IU pointwise NVs". Default is blank
perimetry	perimetry used to obtain normative data, e.g., "static automated perimetry" (default)
strategy	psychophysical strategy used to obtain threshold values, e.g., "SITA standard". Default is blank
size	stimulus size, if the same size was used for all visual field locations or empty (default)
agem	age model to construct the function to obtain TD values
perc	the percentile to obtain the ranked TD value as reference for the general height (GH) of the visual field. Default is the 85th percentile, thus `0.85`
ghfun	function used for determination of the GH and PD values
type	type of estimation for the weighted quantile values. See `wtd.quantile` for details. Default is 'quantile'
...	arguments to be passed to or from methods
sdtd	standard deviations obtained for TD values
sdpd	standard deviations obtained for PD values
g	a table with global indices

Value

`nvgenerate` returns a list with normative values

`agelm` returns a list with coefficients and a function defining a linear age model

`tddef` returns a function for the computation of TD values

`ghdef` returns a function for the computation of the general height

`pddef` returns a function for the computation of PD values

`lutdef` returns a look up table and a function for the computation of the probability values for TD and PD

`gdef` returns a function to compute global indices

`lutgdef` returns a look up table and a function for the computation of the probability values for global indices

Structure of normative values

This is one of the most complex structures in visualFields. It is necessary to be able to run statistical analyses of visual fields obtained from perimetry and it requires data from healthy eyes for its generation. The normative values are only as good as the data they are generated from. Two common ways to generate full normative values from a dataset of healthy eyes, are provided in the package, depending on the `method` selected. The first one, `method="pointwise"`, generates normative values directly from pointwise statistics. The second one, `method="smooth"`, uses a 2D quadratic functions to smooth out those pointwise statistics. Variations or improvements can be regenerated by copying the code in those functions and editing it.

- `info` information regarding normative values. Info is not necessary to carry out statistics, but is useful for the generation of reports. The fields need not be the same as the ones listed here, although these are used in the reports in `vfsfa` for single field analysis and `vfspa` for series progression analysis.
  - `name` name of the normative values
  - `perimetry` perimetry device for which normative values are intended
  - `strategy` psychophysical strategy
  - `size` stimulus size, e.g. Goldmann size III, size V
- `agem` The normative values' age model. The default methods' generate age linear models with coefficients for each location in `locmap` in `coeff` and the function definining the model in `model`
- `sd` standard deviations of the sensitivities, `s`, total deviation (TD) values, `td`, and pattern deviation (PD) values, `pd`
- `luts` Lookup tables to obtain probability levels for TD and PD values.
  - `probs` probability levels
  - `td`, `pd` lookup tables for TD and PD values at each location in `locmaps`
  - `global` lookup table for the following global visual field indices
    - `ms` mean sensitivity (MS) calculated as the unweithed average over locations' values
    - `ss` standard deviation of sensitivity calculated as the unweithed standard deviation over locations' values
    - `md` mean deviation (MD) calculated as the weithed average over locations' values. Weights are the inverse of the standard deviation in `sd` for TD at each location.
    - `sd` standard deviation of total deviation calculated as the weithed standard deviation over locations' values. Weights are the inverse of the standard deviation in `sd` for TD at each location.
    - `pmd` pattern mean deviation calculated as the weithed average over locations' values. Weights are the inverse of the standard deviation in `sd` for PD at each location.
    - `psd` pattern standard deviation calculated as the weithed standard deviation over locations' values. Weights are the inverse of the standard deviation in `sd` for PD at each location.
    - `gh` general height. This is defined traditionally for the 24-2 and the 30-2 as the approximatelly the 85th percentile of TD values
    - `vfi` the oddly defined visual field index
- `tdfun` a function defining how to obtain the TD values. Typically, it is a function of age and sensitivity values and it is defined as sensitivity values minus the age-corrected mean normal obtained as defined in `agem`. Thus, TD values are negative is visual field sensitivity values are below mean normal and positive if they are above mean normal
- `ghfun` a function defining how to obtain the general height
- `pdfun` a function defining how to obtain the PD values. Tipically, they are obtaines as the TD values minus the general height
- `glfun` a function defining how to obtain different global indices
- `tdpfun`, `pdpfun`, `glpfun` mapping functions to get the probability levels corresponding to TD, PD and global indices values and based on the lookup tables defined in `luts`

Examples

```
# generate normative values from SUNY-IU dataset of healthy eyes
# pointwise
sunyiu_24d2_pw <- nvgenerate(vfctrSunyiu24d2, method = "pointwise",
                             name      = "SUNY-IU pointwise NVs",
                             perimetry = "static automated perimetry",
                             strategy  = "SITA standard",
                             size      = "Size III")
# smooth
sunyiu_24d2 <- nvgenerate(vfctrSunyiu24d2, method = "smooth",
                          name      = "SUNY-IU smoothed NVs",
                          perimetry = "static automated perimetry",
                          strategy  = "SITA standard",
                          size      = "Size III")
```

`setdefaults` *Settings in the visualField environment*

Description

Functions to set and get settings in the visualField environment

Usage

```
setdefaults()

getnv()

setnv(nv)

getgpar()

setgpar(gpar)

getlocmap()

setlocmap(locmap)

getlocini()

setlocini(locini = 11)

getvfcols()
```

Arguments

`nv`	normative values to to set in the visualFields environment

gpar	structure with all graphical parameters
locmap	location map to to set in the visualFields environment
locini	column from where to start reading the visual field data

Details

- `setdefaults` sets the default location map, normative value and graphical parameters visualFields environment
- `setnv` sets normative values in the visualFields environment
- `getnv` gets current normative values from the visualFields environment
- `setlocmap` sets a location map in the visualFields environment
- `getlocmap` gets the current location map from the visualFields environment
- `setgpar` sets graphical parameters in the visualFields environment
- `getgpar` gets current graphical parameters from the visualFields environment
- `setlocini` sets the column where visual field data start in the visualFields environment
- `getlocini` gets the column where visual field data starts from the visualFields environment
- `getlocini` gets the column where visual field data starts from the visualFields environment
- `getvfcols` gets all the columns with visual field data

Value

`setdefaults`: No return value

`getnv`: Returns the normative value currently in used by visualFields

`setnv`: No return value

`getgpar`: Returns the graphical parameters currently in used by visualFields

`setgpar`: No return value

`getlocmap`: Returns the location map currently in used by visualFields

`setlocmap`: No return value

`getlocini`: Returns the column where visual field data starts

`setlocini`: No return value

`getvfcols`: Returns the columns with visual field data

Examples

```
# get and set normative values
getnv()$info$name                # print name of set normative values
setnv(normvals$iowa_PC26_pw_cps) # set pointwise normative values
getnv()$info$name                # print name of set normative values
setdefaults()                    # return back to defaults
# get and set a location map
getlocmap()$name         # name of set normative values
setlocmap(locmaps$p30d2) # set the 30-2 location map
getlocmap()$name         # name of set normative values
```

```
setdefaults()            # return back to defaults
# get and set a graphical parameters
getgpar()$tess$xlim  # limits of x axis
setgpar(gpars$pPeri) # set graphical parameters for the Peripheral test
getgpar()$tess$xlim  # limits of x axis
setdefaults()        # return back to defaults
# get and set initial column for visual field data
getlocini()
getvfcols() # get columns with visual fields data
setlocini(15)
getvfcols() # get columns with visual fields data
setdefaults() # return back to defaults
```

---

vfaggregate *Statistical analyses for visual fields data*

Description

- `vfaggregate` computes summary statistics of visual field data
- `vfmean` computes the mean statistics of visual field data. It is a wrapper for vfaggregate but only to compute means
- `vfretestdist` computes the conditional distribution from test-retest data

Usage

```
vfaggregate(vf, by = "date", fun = mean, ...)

vfmean(vf, by = "date", ...)

vfretestdist(vf, nbase = 1, nfollow = 1, alpha = 0.1, ...)
```

Arguments

`vf`	a table with visual fields data. Data is rounded, which leaves sensitivity data unchanged, but it is necessary for the nature of the algorithm if the data passed are TD or PD values or summary stats such as averages. Beware of the locations in the blind spot, which very likely need to be removed
`by`	aggregate by `date`, that is by id, eye, and date (default) or by `eye`, that is by id and eye
`fun`	a function to compute the summary statistics which can be applied to all data subsets. The default is ‘`mean`‘
`...`	arguments to be passed to or from methods. A useful one to try is type of quantile calculation ‘`type`‘ use in `quantile`
`nbase`	number of visual fields to be used as baseline
`nfollow`	number of visual fields to be used as follow up
`alpha`	significance level to derive the conditional retest intervals. Default value is `0.1`

Details

- vfaggregate this is a restricted version of aggregate that only allows to use part of the key hierarchically, and operates on all data frames of the VisualField object. The restriction is that only aggregates that are allowed are 'newkey = c("id","eye")' and 'newkey = c("id","eye","date")'. It returns the aggregated value for all numeric columns grouped and ordered by the new key (id and eye, or id, eye, and date). If the aggregate grouping is by eye and the function, then the date returned is the average.

Value

vfaggregate and vfmean return a vf data frame with aggregate values

vfretestdist returns a list with the following elements:

- x with all the test values (x-axis)
- y the distribution of retest dB values conditional to each test value in x. It is a list with as many entries as x
- n number of retest values conditional to each value in x. It is a list with as many entries as x
- ymed median for each value in x. It is a list with as many entries as x
- ylow quantile value for significance 1 -alpha / 2 for each value in x. It is a list with as many entries as x
- yup quantile value for significance alpha / 2 for each value in x. It is a list with as many entries as x

Together ylow and yup represent the lower and upper limit of the (1 -alpha)% confidence intervals at each value x.

Examples

```
# aggregate by date
vfaggregate(vfpwgRetest24d2, by = "date")           # compute the mean
vfaggregate(vfpwgRetest24d2, by = "date", fun = sd) # compute standard deviation
# aggregate by eye
vfaggregate(vfpwgRetest24d2, by = "eye")           # compute the mean
vfaggregate(vfpwgRetest24d2, by = "eye", fun = sd) # compute standard deviation
# mean by date
vfmean(vfpwgRetest24d2, by = "date")
# mean by eye
vfmean(vfpwgRetest24d2, by = "eye")
# get the retest sensitivity data after removing the blind spot
retest <- vfretestdist(vfpwgRetest24d2, nbase = 1, nfollow = 1)

plot(0, 0, typ = "n", xlim = c(0, 40), ylim = c(0,40),
     xlab = "test in dB", ylab = "retest in dB", asp = 1)
for(i in 1:length(retest$x)) {
  points(rep(retest$x[i], length(retest$y[[i]])), retest$y[[i]],
         pch = 20, col = "lightgray", cex = 0.75)
}
lines(c(0,40), c(0,40), col = "black")
lines(retest$x, retest$ymed, col = "red")
```

```
lines(retest$x, retest$ylow, col = "red", lty = 2)
lines(retest$x, retest$yup, col = "red", lty = 2)
```

---

vfctrIowaPC26 *Central visual field*

Description

Locations of the visual field tested have eccentricities up to 26 degrees and were obtained with a custom static automated perimetry. Data are from 98 eyes of 98 ocular healthy subjects. Each subject underwent two visual field tests, one of the central visual field (64 locations within 26 degrees of fixation) and one of the peripheral visual field (64 locations with eccentricity from 26 to up to 81 degrees)

Usage

```
vfctrIowaPC26
```

Format

See section `Structure of visual fields data` in `vfdesc`

Details

Data are for locations within the central 26 degrees. The data for locations with eccentricity from 26 to up to 81 degrees are in `vfctrIowaPeri`. This dataset of healthy eyes was used to generate the normative values `iowa_PC26_pw`, and `iowa_PC26_pw_cps` included in `normvals`.

References

I. Marin-Franch, P. H. Artes, L. X. Chong, A. Turpin, and M. Wall. *Data obtained with an open-source static automated perimetry test of the full visual field in healthy adults*. Data in Brief, 21:75–82, 2018.

See Also

`vfpwgSunyiu24d2`, `vfctrIowaPeri`, `vfctrSunyiu10d2`, `vfctrSunyiu24d2`, `vfpwgRetest24d2`

vfctrIowaPeri — *Peripheral visual field*

Description

Locations of the visual field tested have eccentricities from 26 to up to 81 degrees and were obtained with a custom static automated perimetry. Data are from 98 eyes of 98 ocular healthy subjects. Each subject underwent two visual field tests, one of the central visual field (64 locations within 26 degrees of fixation) and one of the peripheral visual field (64 locations with eccentricity from 26 to up to 81 degrees)

Usage

```
vfctrIowaPeri
```

Format

See section `Structure of visual fields data` in `vfdesc`

Details

Data are for locations with eccentricity from 26 to up to 81 degrees. The dataset for locations within the central 26 degrees are in `vfctrIowaPC26`. This dataset of healthy eyes was used to generate the normative values `iowa_Peri_pw`, and `iowa_Peri_pw_cps` included in `normvals`.

References

I. Marin-Franch, P. H. Artes, L. X. Chong, A. Turpin, and M. Wall. *Data obtained with an open-source static automated perimetry test of the full visual field in healthy adults*. Data in Brief, 21:75–82, 2018.

See Also

`vfpwgSunyiu24d2`, `vfctrIowaPC26`, `vfctrSunyiu10d2`, `vfctrSunyiu24d2`, `vfpwgRetest24d2`

vfctrSunyiu10d2 — *SUNY-IU dataset of healthy eyes for 10-2 static automated perimetry*

Description

SUNY-IU dataset of healthy eyes for 10-2 static automated perimetry. Courtesy of William H Swanson.

Usage

```
vfctrSunyiu10d2
```

Format

See section `Structure of visual fields data` in `vfdesc`

References

H. J. Wyatt, M. W. Dul, and W. H. Swanson. *Variability of visual field measurements is correlated with the gradient of visual sensitivity*. Vision Research, 47, 2007.

A. Shafi, W. H. Swanson, and M. W. Dul. *Structure and Function in Patients with Glaucomatous Defects Near Fixation*. Optometry and Vision Science, 88, 2011.

See Also

`vfpwgSunyiu24d2`, `vfctrIowaPC26`, `vfctrIowaPeri`, `vfctrSunyiu24d2`, `vfpwgRetest24d2`

`vfctrSunyiu24d2` *SUNY-IU dataset of healthy eyes for 24-2 static automated perimetry*

Description

This dataset of healthy eyes was used to generate the normative values `sunyiu_24d2`, `sunyiu_24d2_pw`, `sunyiu_24d2`, and `sunyiu_24d2_pw_cps` included in `normvals`. Courtesy of William H Swanson and Mitch W Dul

Usage

```
vfctrSunyiu24d2
```

Format

See section `Structure of visual fields data` in `vfdesc`

References

H. J. Wyatt, M. W. Dul, and W. H. Swanson. *Variability of visual field measurements is correlated with the gradient of visual sensitivity*. Vision Research, 47, 2007.

A. Shafi, W. H. Swanson, and M. W. Dul. *Structure and Function in Patients with Glaucomatous Defects Near Fixation*. Optometry and Vision Science, 88, 2011.

See Also

`vfpwgSunyiu24d2`, `vfctrIowaPC26`, `vfctrIowaPeri`, `vfctrSunyiu10d2`, `vfpwgRetest24d2`

vfdesc *Visual field dataset*

Description

The main object of the visualFields package is a table with a specific format and fields that are mandatory for their management and processing (mainly statistical analysis). Each record (row) in the table contains data for a single visual field test. The mandatory fields specify subject (by its ID code), eye, and test date and time. There are required fields statistical and reliability analyses (e.g., age for the determination of total-deviation and pattern-deviation values, and for global indices and fpr, fnr, fl for the proportion of false positives, false negative, and fixation losses). The rest of mandatory fields are sensitivity or deviation data for each visual field test location. (The number of fields for tested locations varies with the location map, 54 for the 24-2, 76 for the 30-2, 68 for the 10-2, etc.). Check section `Structure of visual fields data` below for details about the required structure of the table contatining the visual fields datasets.

The following functions carry out analysis on visual fields data:

- `vfdesc` descriptive summary of a visual field dataset
- `vfsort` sort visual field data
- `vfisvalid` check if a table with visual field data is properly formatted and valid for analysis
- `vfread` read a csv file with visual field data
- `vfwrite` write a csv file with visual field data
- `vfjoin` joins two visual field datasets
- `vffilter` filters elements from a visual field dataset with matching conditions. This function is just a wrapper for `dplyr`'s function `filter`
- `vfselect` select visual field data by index or the first or last `n` visits per subject and eye
- `gettd` computes total-deviation (TD) values and probability values
- `gettdp` computes total-deviation (TD) probability values
- `getpd` computes pattern-deviation (PD) values
- `getpdp` computes pattern-deviation (PD) probability values
- `getgh` computes the general height (GH) from the TD tables
- `getgl` computes visual fields global indices
- `getglp` computes computes visual fields global indices probability values

Usage

```
vfdesc(vf)

vfsort(vf, decreasing = FALSE)

vfisvalid(vf)
```

```
vfread(file, dateformat = "%Y-%m-%d", eyecodes = c("OD", "OS", "OU"), ...)

vfwrite(
  vf,
  file,
  dateformat = "%Y-%m-%d",
  eyecodes = c("OD", "OS", "OU"),
  ...
)

vfjoin(vf1, vf2)

vffilter(vf, ...)

vfselect(vf, sel = "last", n = 1)

gettd(vf)

gettdp(td)

getpd(td)

getpdp(pd)

getgh(td)

getgl(vf)

getglp(g)
```

Arguments

vf	visual field data
decreasing	sort decreasing or increasing? Default is increasing, that is decreasing = FALSE
file	the name of the csv file where to write the data
dateformat	format to be used for date. Its default value is %Y-%m-%d
eyecodes	codification for right and left eye, respectively. By default in visualField uses 'OD' and 'OS' for right and left eye respectively, but it is common to receive csv files with the codes 'R' and 'L'. The code 'OU' for both eyes is also allowed eyecodes should be equal to 'c("OD","OS")' or 'c("R","L")'. By default it is 'eyecodes = c("OD","OS","OU")'
...	arguments to be passed to or from methods
vf1, vf2	the two visual field data objects to join or merge
sel	it can be two things, an array of indices to select from visual field data or a string with the values 'first' or 'last' indicating that only the first few n visits per subject 'id' and 'eye' are to be selected. Default is 'last'.

n	number of visits to select. Default value is 1, but it is ignored if sel is an index array
td	total-deviation (TD) values
pd	pattern-deviation (PD) values
g	global indices

Details

- vfselect when selecting the last or first few visual fields per subject and eye, if that subject and eye has fewer than n visits, then all visits are returned

Value

vfdesc returns descriptive statistics of a visual field dataset

vfsort returns a sorted visual field dataset

vfisvalid returns TRUE or FALSE

vfread returns a visual field dataset

vfwrite No return value

vfjoin returns a visual field dataset

vffilter returns a visual field dataset

vfselect returns a visual field dataset

gettd returns a visual field dataset with total deviation values

gettdp returns a visual field dataset with total deviation probability values

getpd returns a visual field dataset with pattern deviation values

getpdp returns a visual field dataset with pattern deviation probability values

getgh returns the general height of visual fields tests

getgl returns visual fields global indices

getglp returns probability values of visual fields global indices

Structure of visual fields data

Visual fields data is the central object used in visualFields. It is a table of visual field data collected with the same perimeter, background and stimulus paradigm (e.g., static automated perimetry or frequency-doubling perimetry), stimulus size (e.g., Goldmann size III), grid of visual field test locations (e.g., 24-2), and psychophysical testing strategy (e.g., SITA standard). Normative values can be obtained from appropriate datasets with data for healthy eyes and these normative values can then be used to generate statistical analyses and visualizations of data for patients with retinal or visual anomalies.

Each record correspond to a specific test for an eye of a subject taken on a specific date at a specific time. Visual field data must have the following columns

- id an id uniquely identifying a subject. This field is mandatory
- eye should be "OD" for right eye or "OS" for left eye. This field is mandatory

- `date` test date. This field is mandatory
- `time` test time. This field is mandatory
- `age` age of the patient on the test date. This field is required to obtain total-deviation, pattern-deviation values, and other age-dependent local and global indices
- `type` type of subject, Could be a healthy subject (ctr for control) or a patient with glaucoma (pwg) or a patient with idiopatic intraocular hypertension (iih) or other. This field is no required for management or statistical analysis.
- `fpr` false positive rate. This field is no required for management or statistical analysis.
- `fnr` false negative rate. This field is no required for management or statistical analysis.
- `fl` fixation losses. This field is no required for management or statistical analysis.
- `l1..ln` sensitivity, total-deviation, or pattern-deviation values for each location. For analysis with visualFields there should be as many columns as coordinates in the location map set in the visualFields environment. These fields are mandatory.

Examples

```
# get dataset description from visual field table
vfdesc(vfctrSunyiu24d2)
# sort dataset
vfsort(vfctrSunyiu24d2[c(5, 4, 10, 50, 30),])
# check if a visualField is valid
vf <- vfctrSunyiu24d2
vfisvalid(vf) # valid visual field data
vf$id[5] <- NA
vfisvalid(vf) # invalid visual field data
# write and read visual field data
vf <- vfctrSunyiu24d2
tf <- tempfile("vf")
vfwrite(vf, file = tf) # save current locmap in a temp file
head(vfread(tf)) # read the temp file
# join visual fields datasets
vfjoin(vfctrSunyiu24d2, vfpwgRetest24d2)
# visual field subselection
vffilter(vf, id == 1) # fields corresponding to a single subject
vffilter(vf, id == 1 & eye == "OD") # fields for a single subject's right eye
unique(vffilter(vf, eye == "OS")$eye) # only left eyes
vffilter(vfjoin(vfctrSunyiu24d2, vfpwgRetest24d2), type == "ctr") # get only controls
vffilter(vfjoin(vfctrSunyiu24d2, vfpwgRetest24d2), type == "pwg") # get only patients
# select visual fields by index
vfselect(vfctrSunyiu24d2, sel = c(1:4, 150))
# select last few visual fields per subject and eye
vfselect(vfpwgRetest24d2, sel = "last")
# select first few visual fields per subject and eye
vfselect(vfpwgRetest24d2, sel = "first")
vfselect(vfpwgRetest24d2, sel = "first", n = 5) # get the last 5 visits
# compute visual field statistics
vf  <- vfpwgSunyiu24d2
td  <- gettd(vf)  # get TD values
tdp <- gettdp(td) # get TD probability values
```

```
pd  <- getpd(td)  # get PD values
pdp <- getpdp(pd) # get PD probability values
gh  <- getgh(td)  # get the general height
g   <- getgl(vf)  # get global indices
gp  <- getglp(g)  # get global indices probability values
```

---

vfgpar *Plots for visual fields data*

Description

Graphical tools for visualization and statistical analysis of visual fields.

Usage

```
vfgpar(
  coord,
  tess = vftess(coord),
  probs = c(0, 0.005, 0.01, 0.02, 0.05, 0.95, 0.98, 0.99, 0.995, 1),
  cols = c("#000000", colorRampPalette(c("#FF0000", "#FFFF00"))(4), "#F7F0EB",
    colorRampPalette(c("#00FF00", "#008000"))(4)),
  floor = 0,
  ltprobs = c(0, 0.005, 0.01, 0.02, 0.05, 0.95, 1),
  ltcols = c("#000000", colorRampPalette(c("#FF0000", "#FFFF00"))(4), "#F7F0EB",
    "#008000"),
  gtprobs = c(0, 0.05, 0.95, 0.98, 0.99, 0.995, 1),
  gtcols = c("#000000", "#FF0000", "#F7F0EB", colorRampPalette(c("#00FF00",
    "#008000"))(4)),
  neprobs = c(0, 0.0025, 0.005, 0.01, 0.25, 0.975, 0.99, 0.995, 0.9975, 1),
  necols = c("#000000", colorRampPalette(c("#FF0000", "#FFFF00"))(4), "#F7F0EB",
    colorRampPalette(c("#FFFF00", "#FF0000"))(4)),
  bprobs = c(0, 0.005, 0.01, 0.02, 0.05, 0.95, 0.98, 0.99, 0.995, 1),
  bcols = c("#000000", colorRampPalette(c("#FF0000", "#FFFF00"))(4), "#F7F0EB",
    colorRampPalette(c("#00FF00", "#008000"))(4))
)

vftess(coord, floor = 0, delta = 3)

vfcolscheme(
  probs = c(0, 0.005, 0.01, 0.02, 0.05, 0.95, 0.98, 0.99, 0.995, 1),
  cols = c("#000000", colorRampPalette(c("#FF0000", "#FFFF00"))(4), "#F7F0EB",
    colorRampPalette(c("#00FF00", "#008000"))(4)),
  floor = 0
)

vfprogcolscheme(
  probs = c(0, 0.005, 0.01, 0.02, 0.05, 0.95, 1),
```

```
  cols = c("#000000", colorRampPalette(c("#FF0000", "#FFFF00"))(4), "#F7F0EB",
    "#008000")
)

vfplot(vf, type = "td", ...)

vfplotsens(gpar, vf, maxval, digits = 0, ...)

vfplotdev(gpar, vf, dev, devp, digits = 0, ...)

vfplotsdev(gpar, vf, maxval, dev, devp, digits = 0, ...)

vfplotplr(
  vf,
  type = "td",
  alternative = "LT",
  xoffs = 0,
  yoffs = 0,
  addSpark = FALSE,
  thr = 2,
  width = 4,
  height = 2,
  ...
)

vflegoplot(vf, type = "td", grp = 3, ...)

vflegoplotsens(gpar, vfb, vfl, maxb, maxl, crad = 2, digits = 1, ...)

vflegoplotdev(
  gpar,
  vfb,
  devb,
  devpb,
  vfl,
  devl,
  devpl,
  crad = 2,
  digits = 1,
  ...
)

vfsparklines(vf, type = "td", thr = 2, width = 4, height = 2, add = FALSE, ...)
```

Arguments

`coord`	print x and y coordinates. Check section `Structure of graphical parameters` for details

`tess`	tesselation for the visual field maps. Check section `Tesselation in visualFields` for details
`probs`	probability scale to use for TD and PD values. It is a numeric vector of probabilities with values in [0,1]. The values 0 and 1 must be included. Although not technically necessary, it would be best if it is the same as for the normative values used
`cols`	corresponding colors for each of the probability levels
`floor`	Flooring value, typically in dB. Default is 0
`ltprobs, ltcols`	color map for progression with the alternative hypothesis lower than (LT)
`gtprobs, gtcols`	color map for progression with the alternative hypothesis lower than (GT)
`neprobs, necols`	color map for progression with the alternative hypothesis not equal (NE)
`bprobs, bcols`	color map for progression with blth alternative hypotheses LT and GT (B for both)
`delta`	Distance over which the boundary should be shifted. See for `polyclip`
`vf`	the visual fields data to plot
`type`	the type of data to plot: sensitivities (‘`s`‘), total deviation values (‘`td`‘), pattern deviation values (‘`pd`‘), a hybrid plot that shows sensitivity grayscale with TD values and corresponding probability levels (‘`tds`‘), or PD values and corresponding probability levels (‘`pds`‘). Default is ‘`td`‘.
`...`	other graphical arguments. See `plot`
`gpar`	graphical parameters
`maxval`	maximum value, typically in dB, for the generation of a grayscale
`digits`	digits to round values to plot. Default is 0
`dev`	deviation (TD or PD) values
`devp`	deviation (TD or PD) probability values
`alternative`	alternative hypothesis used in progression analyses. Allowed values are ‘`LT`‘ (as in "lower than", default), ‘`GT`‘ (as in "greater than"), ‘`NE`‘ (as in "not equal"), and ‘`both`‘ (both ‘`LT`‘ and ‘`GT`‘)
`xoffs, yoffs`	offset x and y where to print the slope values. That is, the distance from the center of each Voronoy polygons in degrees of visual angle
`addSpark`	whether to overlay a sparkline graph in each visual field location. The parameters `thr`, `width`, and `height` are used only if `addSpark` is `TRUE`. Default value is `FALSE`.
`thr`	threshold used for the median absolute deviation of residuals from simple linear regression. If greater than the threshold, the sparkline for that location is plotted in red and with a thicker line. Default is ‘`2`‘ (dB)
`width`	the width of each pointwise sparkline plot. Default is ‘`4`‘ (degrees of visual angle)
`height`	the height of each pointwise sparkline plot. Default is ‘`2`‘ (degrees of visual angle)

`grp`	number of baseline (first) and last visual fields to group. Default is '`3`'
`vfb`	baseline visual field data
`vfl`	last visual field data
`maxb`	maximum value for the grayscale at baseline visual field data
`maxl`	maximum value for the grayscale for last visual field data
`crad`	radius of the circle in the legoplot
`devb`	baseline visual field (TD or PD) deviation values
`devpb`	baseline visual field (TD or PD) deviation probability values
`devl`	last visual field (TD or PD) deviation values
`devpl`	last visual field (TD or PD) deviation probability values
`add`	whether to generate a new plot ('`FALSE`', as default) or to add to an existing figure ('`TRUE`')

Details

The following functions generate plots using visual fields data

- `vfgpar` generates simple graphical parameters
- `vftess` generates a structure to handle the visual field tessellation. Check section `Tesselation in visualFields` below for further details
- `vfcolscheme` generates the structures to handle the color scheme Check section `Color schemes in visualFields` below for further details
- `vfprogcolscheme` generates the structures to handle the color scheme for progression analysis. Check section `Color schemes in visualFields` below for further details
- `vfplot` plots a single test for visual field data
- `vfplotsens` plots a single test for visual field sensitivity data with a grayscale where darker means greater sensitivity loss
- `vfplotdev` plots a single test for visual field total or pattern deviation data with probability scales represented in color
- `vfplotplr` plots the results of pointwise linear regression for a series of visual fields for an eye from a subject
- `vflegoplot` the legoplot shows the differences between the average values of visual field tests taken as baseline and those at the end of follow up
- `vflegoplotsens` the legoplot for visual field sensitivity data with a grayscale where darker means greater sensitivity loss
- `vflegoplotdev` the legoplot for visual field total or pattern deviation data with probability scales represented in color
- `vfsparklines` the sparklines graph shows spark lines for the series of visual field sensitivities, or total or pattern deviation data for each location

Value

`vfgpar` returns a list with graphical parameters to be used for vfplots

`vftess` returns a list with the `xlim`, `ylim`, tessellation tiles and an outer hull to be used for vfplots

`vfcolscheme` returns a list with a lookup table and a function that define the color scheme to be used for vfplots

`vfprogcolscheme` returns the default `vfcolscheme` to be used for vfplots

`vfplot` No return value

`vfplotsens` No return value

`vfplotdev` No return value

`vfplotsdev` No return value

`vfplotplr` No return value

`vflegoplot` No return value

`vflegoplotsens` No return value

`vflegoplotdev` No return value

`vfsparklines` No return value

Structure of graphical parameters

Graphical parameters for visualFields must be a list containing

- `coord` print x and y coordinates. They could be different from the the real visual field location testing coordinates in complex visual field grids to help readability and improve visualization of statistical results
- `tess` tesselation for the visual field maps. Check section `Tesselation in visualFields`
- `colmap` color map representing the probability scale. Check section `Color schemes in visualFields`

A default graphical parameters can be generated with `generategpar`

Tesselation in visualFields

A tesselation in visualFields must be defined with a list containing

- `xlim`,
- `ylim` 2-dimensional vectors containing the minimum and maximum x and y values
- `floor` the value to be assinged to any sensitivity value lower than `floor`
- `tiles` a list of as many tiles defining the tesselation as visual field test locations. Each element of the list is a table with x and y coordinates defining a polygon containing the corresponding test location. Each polygon is thus the tile for each visual field test location
- `hull` a table with x and y coordinates defining the outer hull of the tessellation

A default tessellation can be generated with `vftess`

Color schemes in visualFields

A color scheme in visualFields must be defined with a list containing

- `map` a table mapping probabilities levels with colors defined in hexadecimal base
- `fun` a function that takes sensitivity values and deviation probability levels and returns the corresponding color code.

A default color scheme can be generated with `vfcolscheme`

Examples

```
# generate a structure with default graphical parameters for the 30-2 map
vfgpar(locmaps$p30d2$coord)
# generate a structure with default tesselation for the 30-2 map
vftess(locmaps$p30d2$coord)
# default color scheme
vfcolscheme()
# default color scheme for progression
vfprogcolscheme()
# plot visual field values for the last field in the series for the first
# subject in the dataset vfpwgSunyiu24d2
# grayscale with sensitivity values
vfplot(vfselect(vffilter(vfpwgRetest24d2, id == 1), n = 1), type = "s")
# TD values
vfplot(vfselect(vffilter(vfpwgRetest24d2, id == 1), n = 1), type = "td")
# PD values
vfplot(vfselect(vffilter(vfpwgRetest24d2, id == 1), n = 1), type = "pd")
# hybrid sensitivities and TD values
vfplot(vfselect(vffilter(vfpwgRetest24d2, id == 1), n = 1), type = "tds")
# hybrid sensitivities and PD values
vfplot(vfselect(vffilter(vfpwgRetest24d2, id == 1), n = 1), type = "pds")
# plot results from pointwise linear regression for the series of
# visual fields for the right eye in the dataset vfpwgSunyiu24d2
# with sensitivity values
vfplotplr(vffilter(vfpwgSunyiu24d2, eye == "OD"), type = "s")
# TD values
vfplotplr(vffilter(vfpwgSunyiu24d2, eye == "OD"), type = "td")
# PD values
vfplotplr(vffilter(vfpwgSunyiu24d2, eye == "OD"), type = "pd")
# legoplot for the series of visual fields for the right eye
# of the subject in the dataset vfpwgSunyiu24d2
# with sensitivity values
vflegoplot(vffilter(vfpwgSunyiu24d2, eye == "OD"), type = "s")
# TD values
vflegoplot(vffilter(vfpwgSunyiu24d2, eye == "OD"), type = "td")
# PD values
vflegoplot(vffilter(vfpwgSunyiu24d2, eye == "OD"), type = "pd")
# sparklines for the series of visual fields for the right eye of
# the subject in the dataset vfpwgSunyiu24d2
# with sensitivity values
vfsparklines(vffilter(vfpwgSunyiu24d2, eye == "OD"), type = "s")
# TD values
```

```
vfsparklines(vffilter(vfpwgSunyiu24d2, eye == "OD"), type = "td")
# PD values
vfsparklines(vffilter(vfpwgSunyiu24d2, eye == "OD"), type = "pd")
```

`vfpwgRetest24d2` *Short-term retest static automated perimetry data*

Description

Thirty patients recruited from the glaucoma clinics at the Queen Elizabeth Health Sciences Centre in Halifax, Nova Scotia. Each patient underwent 12 visual fields in 12 consecutive weekly sessions.

Usage

```
vfpwgRetest24d2
```

Format

See section `Structure of visual fields data` in `vfdesc`

References

P. H. Artes, N. O'Leary, M. T. Nicolela, B. C. Chauhan, and D. P. Crabb. *Visual field progression in glaucoma: What is the specificity of the guided progression analysis?* American Academy of Ophthalmology, 121(10):2023-2027, 2014.

See Also

`vfpwgSunyiu24d2`, `vfctrIowaPC26`, `vfctrIowaPeri`, `vfctrSunyiu10d2`, `vfctrSunyiu24d2`

`vfpwgSunyiu24d2` *Series of 24-2 static automated perimetry data for a patient with glaucoma*

Description

This is real data for the right and left eyes, but the age has been changed to protect anonymity of the subject. Courtesy of William H Swanson and Mitch W Dul

Usage

```
vfpwgSunyiu24d2
```

Format

See section `Structure of visual fields data` in `vfdesc`

See Also

vfctrIowaPC26, vfctrIowaPeri, vfctrSunyiu10d2, vfctrSunyiu24d2, vfpwgRetest24d2

---

vfsfa *Single Field Reporting*

Description

Generates of one-page reports of single field analyses

Usage

```
vfsfa(vf, file, ...)

vfsfashiny(vf, ...)
```

Arguments

vf	visual field data
file	The pdf file name where to save the one-page reports of single field analysis
...	other graphical arguments

Details

- `vfsfa` saves a pdf with one-page reports of single field analyses
- `vfsfashiny` generates interactive one-page reports of single field analyses based on Shiny

Value

No return value

---

vfspa *Series Progession Analysis*

Description

Generation of one-page reports of series progression analyses

- `vfspa` saves a pdf with one-page reports of series progression analyses
- `vfspashiny` generates interactive one-page reports of series progression analyses based on Shiny

Usage

```
vfspa(
  vf,
  file,
  type = "td",
  nperm = factorial(7),
  trunc = 1,
  testSlope = 0,
  ...
)

vfspashiny(
  vf,
  type = "td",
  nperm = factorial(7),
  trunc = 1,
  testSlope = 0,
  ...
)
```

Arguments

`vf`	visual field data
`file`	The pdf file name where to save the one-page reports of single field analysis
`type`	Type of data to use. It can be '`s`', '`td`', or '`pd`'.
`nperm`	Number of permutations. Default is 7!
`trunc`	value for the Truncated Product Method (see reference). Default is 1
`testSlope`	slope, or slopes, to test as null hypothesis. Default is 0. if a single value, then the same null hypothesis is used for all locations. If a vector of values, then (for `plr` and `poplr`) each location of the visual field will have a different null hypothesis. The length of testSlope must be 1 or equal to the number of locations to be used in the PLR or PoPLR analysis
`...`	other graphical arguments

Value

No return value

References

N. O'Leary, B. C. Chauhan, and P. H. Artes. *Visual field progression in glaucoma: estimating the overall significance of deterioration with permutation analyses of pointwise linear regression (PoPLR)*. Investigative Ophthalmology and Visual Science, 53, 2012

N. O'Leary, B. C. Chauhan, and P. H. Artes. *Visual field progression in glaucoma: estimating the overall significance of deterioration with permutation analyses of pointwise linear regression (PoPLR)*. Investigative Ophthalmology and Visual Science, 53, 2012

Index