Package ‘prevR’

May 6, 2019

Type Package
Title Estimating Regional Trends of a Prevalence from a DHS
Version 3.4.0
Maintainer Joseph Larmarange <joseph.larmarange@ird.fr>

Acknowledgements

funding from ANRS and IRD, and technical support from LYSIS (info@lysis-consultants.fr)

Description

Spatial estimation of a prevalence surface or a relative risks surface, using data from a Demographic and Health Survey (DHS) or an analog survey, see Larmarange et al. (2011) <doi:10.4000/cybergeo.24606>.

License CeCILL
URL https://github.com/larmarange/prevR/

BugReports https://github.com/larmarange/prevR/issues

LazyLoad yes
LazyData yes

Depends R (>= 2.10)
Imports sp, rgdal (>= 0.7), ggplot2, directlabels, GenKern, fields, gstat, foreign, maptools, methods
Suggests tcltk, geoR, knitr, rmarkdown

Encoding UTF-8

Collate 'prevR-package.r' 'setClass.prevR.r' 'is.prevR.r' 'as.prevR.r' 'onAttach.r' 'isInputOk.prevR.r' 'changeproj.prevR.r' 'colors.prevR.r' 'as.data.frame.prevR.r' 'plot.prevR.r' 'show.prevR.r' 'print.prevR.r' 'summary.prevR.r' 'rings.prevR.r' 'as.SpatialGrid.prevR.r' 'create.boundary.prevR.r' 'export.prevR.r' 'import.dhs.prevR.r' 'point.in.SpatialPolygons.prevR.r' 'NA[outside.SpatialPolygons.prevR.r'] 'xyz2dataframe.prevR.r' 'init.exp.model.variogram.r' 'eyefit.prevR.r' 'krige.prevR.r' 'kde.prevR.r' 'Noptim.r' 'theme_prevR.r' 'direct.label_prevR.r' 'quick.prevR.r'
**VignetteBuilder**  knitr

**RoxygenNote**  6.1.1

**NeedsCompilation**  no

**Author**  Joseph Larmarange [aut, cre] (<https://orcid.org/0000-0001-7097-700X>)

**Repository**  CRAN

**Date/Publication**  2019-05-06 20:51:23 UTC

### R topics documented:

<table>
<thead>
<tr>
<th>Topic</th>
<th>Page</th>
</tr>
</thead>
<tbody>
<tr>
<td>prevR-package</td>
<td>3</td>
</tr>
<tr>
<td>as.data.frame.prevR</td>
<td>6</td>
</tr>
<tr>
<td>as.prevR</td>
<td>7</td>
</tr>
<tr>
<td>as.SpatialGrid,prevR-method</td>
<td>8</td>
</tr>
<tr>
<td>changeproj.prevR-method</td>
<td>9</td>
</tr>
<tr>
<td>create.boundary</td>
<td>10</td>
</tr>
<tr>
<td>direct.label_prevR</td>
<td>11</td>
</tr>
<tr>
<td>export.prevR-method</td>
<td>12</td>
</tr>
<tr>
<td>fdhs</td>
<td>14</td>
</tr>
<tr>
<td>import.dhs</td>
<td>14</td>
</tr>
<tr>
<td>is.prevR</td>
<td>15</td>
</tr>
<tr>
<td>kde.prevR-method</td>
<td>16</td>
</tr>
<tr>
<td>krige,ANY,prevR-method</td>
<td>19</td>
</tr>
<tr>
<td>NA.outside.SpatialPolygons</td>
<td>21</td>
</tr>
<tr>
<td>Noptim</td>
<td>21</td>
</tr>
<tr>
<td>plot.prevR,missing-method</td>
<td>22</td>
</tr>
<tr>
<td>point.in.SpatialPolygons</td>
<td>23</td>
</tr>
<tr>
<td>prevR-class</td>
<td>24</td>
</tr>
<tr>
<td>prevR.colors</td>
<td>26</td>
</tr>
<tr>
<td>print.prevR-method</td>
<td>28</td>
</tr>
<tr>
<td>quick.prevR</td>
<td>28</td>
</tr>
<tr>
<td>rings.prevR-method</td>
<td>30</td>
</tr>
<tr>
<td>show.prevR-method</td>
<td>32</td>
</tr>
<tr>
<td>summary.prevR-method</td>
<td>32</td>
</tr>
<tr>
<td>theme_prevR</td>
<td>33</td>
</tr>
<tr>
<td>TMWorldBorders</td>
<td>34</td>
</tr>
<tr>
<td>xyz2dataframe</td>
<td>35</td>
</tr>
</tbody>
</table>

**Index**  36
**Description**

prevR package

Estimating regional trends of a prevalence from a DHS.

**Details**

Package: prevR
Type: Package
Licence: CeCILL-C - [http://www.cecill.info/licences/Licence_CeCILL-C_V1-en.html](http://www.cecill.info/licences/Licence_CeCILL-C_V1-en.html)
Website: [https://larmarange.github.io/prevR/](https://larmarange.github.io/prevR/)

This package performs a methodological approach for spatial estimation of regional trends of a prevalence using data from surveys using a stratified two-stage sample design (as Demographic and Health Surveys). In these kind of surveys, positive and control cases are spatially positioned at the centre of their corresponding surveyed cluster.

This package provides functions to estimate a prevalence surface using a kernel estimator with adaptive bandwiths of equal number of persons surveyed (a variant of the nearest neighbour technique) or with fixed bandwiths. The prevalence surface could also be calculated using a spatial interpolation (kriging or inverse distance weighting) after a moving average smoothing based on circles of equal number of observed persons or circles of equal radius.

With the kernel estimator approach, it’s also possible to estimate a surface of relative risks.

For a quick demo, enter `quick.prevR(fdhs)`.

For a full demo, enter `demo(prevR)`.

The content of prevR can be broken up as follows:

**Datasets**

*fdhs* is a fictive dataset used for testing the package.

*TMWorldBorders* provides national borders of every countries in the World and could be used to define the limits of the studied area.

**Creating objects**

prevR functions takes as input objects of class prevR.

import.dhs allows to import easily, through a step by step procedure, data from a DHS (Demographic and Health Surveys) downloaded from [http://www.measuredhs.com](http://www.measuredhs.com).

as.prevR is a generic function to create an object of class prevR.

create.boundary could be used to select borders of a country and transfer them to as.prevR in order to define the studied area.

**Data visualisation**

Methods `show`, `print` and `summary` display a summary of a object of class prevR.
The method \texttt{plot} could be used on a object of class \texttt{prevR} for visualising the studied area, spatial position of clusters, number of observations or number of positive cases by cluster.

\textit{Data manipulation}

The method \texttt{changeproj} changes the projection of the spatial coordinates.

The method \texttt{as.data.frame} converts an object of class \texttt{prevR} into a data frame.

The method \texttt{export} export data and/or the studied area in a text file, a dbf file or a shapefile.

\textit{Data analysis}

\texttt{rings} calculates rings of equal number of observations and/or equal radius.

\texttt{kde} calculates a prevalence surface or a relative risks surface using gaussian kernel density estimators (kde) with adaptative bandwidths.

\texttt{krige} executes a spatial interpolation using an ordinary kriging.

\texttt{idw} executes a spatial interpolation using an inverse distance weighting (idw) technique.

\textit{Results visualisation and export}

Outputs of \texttt{kde, krige} and \texttt{idw} are objects of class \texttt{SpatialPixelsDataFrame\{sp\}}.

Results could be plotted using the function \texttt{spplot\{sp\}}.

\texttt{prevR} provides several continuous color palettes (see \texttt{prevR.colors}) compatible with \texttt{spplot}. Calculated surfaces could be export using the function \texttt{writeAsciiGrid\{maptools\}}.

\textbf{Acknowledgement}

\texttt{prevR} has been developed with funding from the French National Agency for Research on AIDS and Viral Hepatitis (ANRS - \url{http://www.anrs.fr}) and the French Research Institute for Sustainable Development (IRD - \url{https://www.ird.fr}), and technical support from LYSIS (info@lysis-consultants.fr).

\textbf{Citation}

To cite \texttt{prevR}:


\textbf{Author(s)}

Joseph Larmarange <joseph.larmarange@ird.fr>
IRD - CEPED (UMR 196 Université Paris Descartes Ined IRD)

\textbf{References}


**Examples**

```r
## Not run:
par(ask = TRUE)
# Creating an object of class prevR
col <- c(id = "cluster",
        x = "x",
        y = "y",
        n = "n",
        pos = "pos",
        c.type = "residence",
        wn = "weighted.n",
        wpos = "weighted.pos"
)
dhs <- as.prevR(fdhs.clusters, col, fdhs.boundary)

str(dhs)
print(dhs)

plot(dhs, main="Clusters position")
plot(dhs, type="c.type", main="Clusters by residence")
plot(dhs, type="count", main="Observations by cluster")
plot(dhs, type="flower", main="Positive cases by cluster")

# Changing coordinates projection
plot(dhs, axes=TRUE)
dhs <- changeproj(dhs,
                  "+proj=utm +zone=30 +ellps=WGS84 +datum=WGS84 +units=m +no_defs")
print(dhs)
plot(dhs, axes=TRUE)

# Calculating rings of equal number of observations for different values of N
dhs <- rings(dhs, N=c(100,200,300,400,500))
print(dhs)
summary(dhs)

# Prevalence surface for N=300
prev.N300 <- kde(dhs, N=300, nb.cells=200)
spplot(prev.N300, 'k.prev.N300.RInf',
       cuts=100, col.regions=prevR.colors.red(101),
       main="Regional trends of prevalence (N=300)"
)

# Smoothing ring radii surface (spatial interpolation by kriging)
```

---

**prevR-package**


**Examples**

```r
## Not run:
par(ask = TRUE)
# Creating an object of class prevR
col <- c(id = "cluster",
        x = "x",
        y = "y",
        n = "n",
        pos = "pos",
        c.type = "residence",
        wn = "weighted.n",
        wpos = "weighted.pos"
)
dhs <- as.prevR(fdhs.clusters, col, fdhs.boundary)

str(dhs)
print(dhs)

plot(dhs, main="Clusters position")
plot(dhs, type="c.type", main="Clusters by residence")
plot(dhs, type="count", main="Observations by cluster")
plot(dhs, type="flower", main="Positive cases by cluster")

# Changing coordinates projection
plot(dhs, axes=TRUE)
dhs <- changeproj(dhs,
                  "+proj=utm +zone=30 +ellps=WGS84 +datum=WGS84 +units=m +no_defs")
print(dhs)
plot(dhs, axes=TRUE)

# Calculating rings of equal number of observations for different values of N
dhs <- rings(dhs, N=c(100,200,300,400,500))
print(dhs)
summary(dhs)

# Prevalence surface for N=300
prev.N300 <- kde(dhs, N=300, nb.cells=200)
spplot(prev.N300, 'k.prev.N300.RInf',
       cuts=100, col.regions=prevR.colors.red(101),
       main="Regional trends of prevalence (N=300)"
)

# Smoothing ring radii surface (spatial interpolation by kriging)
```
as.data.frame.prevR  

Convert an object of class prevR into a data.frame.

Description

This function merges the slots clusters et rings of a object of class prevR.

Usage

## S3 method for class 'prevR'
as.data.frame(x, ..., N = NULL, R = NULL, clusters.only = FALSE)

Arguments

x       object of class prevR.
...   not used, for compatibility with the generic method as.data.frame.
N   integer or list of integers setting elements of rings to extract.
R   integer or list of integers setting elements of rings to extract.
clusters.only return only the slot clusters of x?

Value

If clusters.only = TRUE, the function will return only the slot clusters of x.

Otherwise, slots clusters and rings of x will be merged in a unique data frame. The columns of rings will be renamed adding a suffix like .N300.RInf.

N and R define the elements of rings to extract. If not specified (NULL), all the elements of rings will be included.

See Also

as.data.frame(base), prevR-class.
### Examples

```r
str(fdhs)
str(as.data.frame(fdhs))
## Not run:
  r.fdhs <- rings(fdhs, N=c(100,200,300))
  str(r.fdhs)
  str(as.data.frame(r.fdhs, clusters.only=TRUE))
  str(as.data.frame(r.fdhs))
  str(as.data.frame(r.fdhs, N=300))

## End(Not run)
```

---

**as.prevR**  
*Create an object of class prevR.*

---

**Description**

This function creates an object of class `prevR` from a data frame.

**Usage**

```r
as.prevR(data, col, boundary = NULL, 
  proj = "+proj=longlat +ellps=WGS84")
```

**Arguments**

- `data`  
  data frame, each line corresponding to an observed cluster.

- `col`  
  vector identifying the columns of data to use.

  clusters columns names are fixed:

  - "id" (optional) cluster's identifier.
  - "x" cluster's longitude.
  - "y" cluster's latitude.
  - "n" number of valid observations in the cluster.
  - "pos" number of positive cases in the cluster.
  - "wn" (optional) sum of observations weight.
  - "wpos" (optional) sum of positive cases weight.
  - "c.type" (optional) type of cluster (used only by `plot`).

  See examples.

- `boundary`  
  object of class `SpatialPolygons` defining the studied area.

- `proj`  
  projection of clusters coordinates used in `data` (longitude and latitude in decimal degrees by default).
Details

Only "x", "y", "n" and "pos" are required in col. If "id" is not specified, a numerical identifier will be automatically created.

proj defines projection used by data. It could be a character string corresponding to a PROJ.4 projection (see http://trac.osgeo.org/proj/ for more details) or an object of class CRS(sp).

If the projection of boundary is defined in a slot called proj4string, boundary will be projected according to proj. If the slot proj4string is missing, boundary will be considered to be already in the same projection as proj.

If boundary is not defined (NULL), a considered corresponding to minimal and maximal coordinates of data will be used.

boundary could be the result of the function create.boundary.

It’s not possible to change projection of data with as.prevR. Use changeproj instead.

Value

Object of class prevR (see prevR-class for more details)

See Also

prevR-class, create.boundary, changeproj, import.dhs.

Examples

```r
col <- c(id = "cluster",
    x = "x",
    y = "y",
    n = "n",
    pos = "pos",
    c.type = "residence",
    wn = "weighted.n",
    wpos = "weighted.pos"
)
dhs <- as.prevR(fdhs.clusters, col, fdhs.boundary)
str(dhs)
print(dhs)
```
Usage

```r
## S4 method for signature 'prevR'
as.SpatialGrid(object, nb.cells = 100,
cell.size = NULL)
```

Arguments

- `object`: object of class `prevR`.
- `nb.cells`: number of cells on the longest side of the studied area (unused if `cell.size` is defined).
- `cell.size`: size of each cell (in the unit of the projection).

Details

This function generates a spatial rectangular grid, each cell being a square of side `cell.size`. If `cell.size` is not defined, side of cells will be calculated as the longest side of the slot boundary of `object` divided by `nb.cells`.

Value

Object of class `SpatialGrid[sp]`.

See Also

`GridTopology[sp], SpatialGrid-class[sp]`.

Examples

```r
str(as.SpatialGrid(fdhs))
str(as.SpatialGrid(fdhs, nb.cells=200))
```

Description

This function converts map projection (and/or datum) used by an object of class `prevR` into another one.

Usage

```r
## S4 method for signature 'prevR'
changeproj(object, proj)
```
Arguments

object object of class prevR.
proj new map projection.

Details

proj could be a character string corresponding to a PROJ.4 projection (see http://trac.osgeo.org/proj/ for more details) or an object of class CRS\{sp\}.

changeproj transform the columns "x" and "y" of the slot clusters of object and convert boundary using the new map projection defined by proj.
If applicable, the slot rings will be recalculated.

Value

Return object expressed in the projection proj.

See Also

spTransform\{rgdal\}, prevR-class.

Examples

print(fdhs)
plot(fdhs, axes=TRUE, main="Projection: longitude/latitude")

fdhs2 <- changeproj(fdhs,
  "+proj=utm +zone=30 +ellps=WGS84 +datum=WGS84 +units=m +no_defs")
print(fdhs2)
plot(fdhs2, axes=TRUE, main="Projection: UTM Zone 30")
Arguments

countries
a vector of character string corresponding to the name of the countries you want to extract from the dataset. If NULL, a dialog box will be appear in order to select the desired country.
multiple
should the dialog box allow multiple selection (unused if countries is specified)?
proj
map projection to use for the result (longitude and latitude in decimal degrees by default).

Details

proj could be a character string corresponding to a PROJ.4 projection (see http://trac.osgeo.org/proj/ for more details) or an object of class CRS[sp].

Value

Object of class SpatialPolygons{sp}.

Note

The result will be automatically plotted.

See Also

TMWorldBorders.

Examples

## Not run:
boundary <- create.boundary()

## End(Not run)

boundary <- create.boundary("Burkina Faso")
boundary <- create.boundary("Burkina Faso",
  proj="+proj=utm +zone=30 +ellps=WGS84 +datum=WGS84 +units=m +no_defs")
boundary <- create.boundary(countries = c("Burkina Faso", "Ghana", "Benin"))
Usage

direct.label.prevR(p, method = NULL, debug = FALSE)

Arguments

p The ggplot object.
method Method for direct labeling as described in ?label.positions.
debug Show debug output?

Value

The ggplot object with direct labels added.

Note

This function is based on and similar to direct.label.ggplot except that legend is not hidden.

See Also

direct.label.ggplot[directlabels]

Description

This method could be used to export an object of class prevR in different formats (text, shapefile, dbase...)

Usage

## S4 method for signature 'prevR'
export(object, element, format, file, N = NULL,
   R = NULL, clusters.only = FALSE, ext = NULL, sep = NULL,
   dec = NULL, ...)

Arguments

object object of class prevR.

element element to export: "clusters" or "boundary".

format format: "dbf", "txt", "csv", "csv2" or "shp" (unused if element="boundary").

file file name without extension.

N integer or list of integers setting elements of rings to export (unused if element="boundary").

R integer or list of integers setting elements of rings to export (unused if element="boundary").

clusters.only export only the slot clusters of object (unused if element="boundary")?
ext coerce the extension of the export file (unused if `element="boundary"` or if `format="shp"`).

sep coerce the field separator string (unused if `element="boundary"` or if `format="shp"` or if `format="dbf"`).

dec coerce the string to use for decimal point (unused if `element="boundary"` or if `format="shp"` or if `format="dbf"`).

... additional arguments transmitted to `writeOGR`, `write.dbf` or `write.table`.

Details

If `element="boundary"`, the slot `boundary` of object will be exported as a `shapefile`.

Otherwise, the slot `clusters`, merged with the slot `rings`, will be exported.

See `as.data.frame.prevR` for details on the use of the parameters of `N, Ret clusters.only`.

`format` specifies the export format of the data frame returned by `as.data.frame.prevR`:

- "shp" Shape File (require the package `rgdal`)
- "dbf" DBASE format (extension: .dbf, require the package `foreign`)
- "txt" tabulated text (extension: .txt)
- "csv" 'comma separated values' (extension: .csv)
- "csv2" CSV variant using a semicolon as field separator (extension: .csv)

`ext` could be used to coerce the extension of the output file, except for `shapefile` export, which will write four different files (.shp, .shx, .dbf and .prj).

The "txt" format uses by default a tabulation as field separator and a point "." for decimal point. The "csv" format uses a comma "," as field separator and a point "." as decimal point.

The "csv2" format is a variant using a semicolon ";" as field separator and a colon ":" for decimal point, the Excel convention for CSV files in some Western European locales.

`sep` and `dec` could be used to coerce the field separator and the decimal point (together with the "txt" format).

See Also

`writePolyShape [maptools], writePointsShape [maptools], write.dbf [foreign], write.table [utils]`.

Examples

```r
## Not run:
export(fdhs, element="boundary", file="area")
export(fdhs, element="clusters", format="shp", file="points")

dhs <- rings(fdhs,N=c(100,300,500))
export(dhs, element="clusters", format="csv", N=300, file="points")

## End(Not run)
```
import.dhs

Fictitious data generated by a DHS simulation.

Description

Data set generated by a Demographic and Health Survey (DHS) simulation on a fictitious country with a national prevalence of 10%, 8000 having been surveyed, distributed in 401 clusters. This dataset is composed of 3 objects:

- **fdhs.clusters**: data frame (one line per cluster).
- **fdhs.boundary**: object of class `SpatialPolygons` corresponding to the borders of the fictitious country.
- **fdhs**: object of class `prevR` returned by `as.prevR` using the two previous objects.

Examples

```r
# Not run:
str(fdhs)
str(fdhs.clusters)
str(fdhs.boundary)
demo(prevR)

# End(Not run)
```

import.dhs

Import DHS data.

Description

This step by step function guides users to import data from a Demographic and Health Survey (DHS) and create an object of class `prevR`.

Usage

```r
import.dhs(file.sav, file.dbf)
```

Arguments

- **file.sav**: DHS data (one individual per line) in SPSS format (.sav), downloaded from [http://www.dhsprogram.com/](http://www.dhsprogram.com/). Could also be directly a data.frame.
is.prevR

Note

If you don’t provide the precise path of files, R will check the working directory (see `setwd`). To specify the file path, see `file.path`.

This function was developed specifically for importing DHS. For a generic function for creating an object of class `prevR`, see `as.prevR`.

See Also

`as.prevR`, `prevR-class`.

Examples

```r
## Not run:
imported_data <- import.dhs("data.sav", "gps.dbf")

## End(Not run)
```

---

### is.prevR

Test if an object is of class prevR. This function test if the class of an object is `prevR`. It could be used to test the slot `rings` or the slot `boundary`.

#### Description

Test if an object is of class prevR. This function test if the class of an object is `prevR`. It could be used to test the slot `rings` or the slot `boundary`.

#### Usage

```r
is.prevR(object, slot = NULL)
```

#### Arguments

- `object`: object to test.
- `slot`: "clusters", "rings", "boundary" or "proj".

#### Details

Slots `rings` and `boundary` are always present in an object of class `prevR`, but `rings` could be `NULL` and `boundary` a `SpatialPolygons` with an attribute named `valid` with the value `FALSE` (when boundaries of the studied area have not been specified explicitly).

- If `rings` is `NULL`, `is.prevR(object,"rings")` will return `FALSE`.
- If `boundary` has an attribute `valid` equal to `FALSE`, `is.prevR(object,"boundary")` will return `FALSE`. 
Value

TRUE or FALSE.

See Also

prevR-class.

Examples

col <- c(id = "cluster",
         x = "x",
         y = "y",
         n = "n",
         pos = "pos",
         c.type = "residence",
         wn = "weighted.n",
         wpos = "weighted.pos"
)
dhs <- as.prevR(fdhs.clusters, col, fdhs.boundary)

is.prevR(dhs)
is.prevR(dhs,"rings")
is.prevR(dhs,"boundary")
dhs <- rings(dhs,N=300)
is.prevR(dhs,"rings")

---

kde.prevR-method

*Kernel density estimation for prevR object.*

Description

This function allows to calculate a prevalence surface (ratio of two intensity surfaces) and/or a relative risks surface (ratio of two density surfaces) using gaussian kernel estimators with adaptative bandwidths of equal number of observations or equal radius.

Usage

```r
## S4 method for signature 'prevR'
kde(object, N = NULL, R = NULL, weighted = TRUE,
     risk.ratio = FALSE, keep.details = FALSE, nb.cells = 100,
     cell.size = NULL, progression = TRUE, short.names = FALSE)
```
**kde.prevR-method**

**Arguments**

- **object**: object of class `prevR`.
- **N**: integer or list of integers corresponding to the rings to use.
- **R**: integer or list of integers corresponding to the rings to use.
- **weighted**: use weighted data (TRUE, FALSE or "2")?
- **risk.ratio**: calculate a relative risks surface instead of a prevalence surface (TRUE, FALSE or "2")?
- **keep.details**: return surface of positive cases and surface of observed cases?
- **nb.cells**: number of cells on the longest side of the studied area (unused if `cell.size` is defined).
- **cell.size**: size of each cell (in the unit of the projection).
- **progression**: show a progress bar?
- **short.names**: should names of the output be short?

**Details**

This function calculates a prevalence surface as the ratio of the intensity surface (expressed in cases per surface unit) of positive cases on the intensity surface of observed cases and could also calculate a relative risks surface corresponding to the ratio of the density surface (whose integral has been normalized to one) of positive cases on density surface of observed cases.

This method is a variant of the nearest neighbour technique. Surfaces are estimated using gaussian kernel estimators with adaptative bandwidths, bandwidth size being determined by a minimum number of observations in the neighbourhood (see `rings` for more details). Fixed bandwidths could also be used. More precisely, the bandwidth used is half the radius of rings of equal number of observations or equal radius (parameters `N` and `R`) calculated by the function `rings`.

See references for a detailed explanation of the implemented methodology.

`N` and `R` determine the rings to use for the estimation. If they are not defined, surfaces will be estimated for each available couples (`N,R`). Several estimations could be simultaneously calculated if several values of `N` and `R` are defined.

A suggested value of `N` could be computed with `Noptim`.

**Value**

Object of class `SpatialPixelsDataFrame`. Surfaces are named according to the name of the corresponding variable, `N` and `R` (for example: `k.prev.N300.RInf`). If `short.names` is TRUE and if there is only one combination of couples (`N, R`), variable names will not be suffixed by the value of `N` and `R`.

Estimated variables are (depending on the function parameters):

- "k.pos" unweighted intensity surface of positive cases.
- "k.obs" unweighted intensity surface of observed cases.
- "k.prev" unweighted surface of prevalence (k.pos/k.obs).
- "k.case" unweighted density surface of positive cases.
• "k.control" unweighted density surface of observed cases.
• "k.rr" unweighted surface of relative risks (k.case/k.control).
• "k.wpos" weighted intensity surface of positive cases.
• "k.wobs" weighted intensity surface of observed cases.
• "k.wprev" weighted surface of prevalence (k.wpos/k.wobs).
• "k.wcase" weighted density surface of positive cases.
• "k.wcontrol" weighted density surface of observed cases.
• "k.wrr" weighted surface of relative risks (k.wcase/k.wcontrol).

NA value is applied to cells of the grid located outside of the studied area (see `NA.outside.SpatialPolygons`).

Note

Results could be plotted with `spplot(sp)`. `prevR` provides several continuous color palettes (see `prevR.colors`) compatible with `spplot`. Calculated surfaces could be export using the function `writeAsciiGrid(maptools)`. See the package `sparr` for another methodology to estimate relative risks surfaces, adapted for other kind of data than Demographic and Health Surveys (DHS).

References


See Also

`KernSur(GenKern), rings.prevR-method, Noptim`.

Examples

```r
# Not run:
dhs <- rings(fdhs, N=c(100,200,300,400,500))
prev.N300 <- kde(dhs, N=300, nb.cells=200)

library(sp)
spplot(prev.N300, 'k.wprev.N300.RInf',
       cuts=100, col.regions=prevR.colors.red(101),
       main="Regional trends of prevalence (N=300)"
)

prev.krige <- kde(dhs, N=c(100,300,500), R=Inf,
                   nb.cells=200, risk.ratio=2, keep.details=FALSE
)
str(prev.krige)
spplot(prev.krige,
```
krige,ANY,prevR-method

Spatial interpolation (kriging and inverse distance weighting) for objects of class prevR.

Description

These functions execute a spatial interpolation of a variable of the slot rings of an object of class prevR. The method krige implements the ordinary kriging technique. The method idw executes an inverse distance weighting interpolation.

Usage

```r
# S4 method for signature 'ANY,prevR'
krige(formula, locations, N = NULL, R = Inf,
      model = NULL, nb.cells = 100, cell.size = NULL, fit = "auto",
      keep.variance = FALSE, show.variogram = FALSE, ...)

# S4 method for signature 'ANY,prevR'
idw(formula, locations, N = NULL, R = Inf,
     nb.cells = 100, cell.size = NULL, idp = 2, ...)
```

Arguments

- **formula** variable(s) to interpolate (see details).
- **locations** object of class prevR.
- **N** integer or list of integers corresponding to the rings to use.
- **R** integer or list of integers corresponding to the rings to use.
- **model** a variogram model returned by the function `vgm[gstat]`.
- **nb.cells** number of cells on the longest side of the studied area (unused if cell.size is defined).
- **cell.size** size of each cell (in the unit of the projection).
- **fit** "auto" for using a variogram automatically fitted from the data, "manual" for using a variogram fitted through a graphic interface (unused if model is defined).
- **keep.variance** return variance of estimates?
- **show.variogram** plot the variogram?
- **...** additional arguments transmitted to `krige[gstat]` or `idw[gstat]`.
- **idp** inverse distance weighting power (see `idw[gstat]`).
Details

formula specifies the variable(s) to interpolate. Only variables available in the slot rings of locations could be used. Possible values are "r.pos", "r.n", "r.prev", "r.radius", "r.clusters", "r.wpos", "r.wn" or "r.wprev". Variables could be specified with a character string or a formula (example: list(r.pos~1, r.prev~1). Only formula like variable.name~1 are accepted. For more complex interpolations, use directly functions krig and idw from gstat.

N and R determine the rings to use for the interpolation. If they are not defined, surfaces will be estimated for each available couples (N,R). Several interpolations could be simultaneously calculated if several variables and/or several values of N and R are defined.

A suggested value of N could be computed with Noptim.

In the case of an ordinary kriging, the method krig from prevR will try to fit automatically an exponential variogram to the sample variogram (fit = "auto"). If you choose fit = "manual", the sample variogram will be plotted and a graphical dialog box (adapted from eyefit{geoR}) will appear for a manual and visual fitting. You can also specify directly the variogram to use with the parameter model. Packages geoR and tcltk are required for manual fit.

Interpolations are calculated on spatial gridd obtained with as.SpatialGrid.

Value

Object of class SpatialPixelsDataFrame. The name of estimated surfaces depends on the name of the interpolated variable, N and R (for example: r.radius.N300.RInf). If you ask the function to return variance (keep.variance=TRUE), corresponding surfaces names will have the suffix .var.

NA value is applied to points located outside of the studied area (voir NA.outside.SpatialPolygons).

Note

Results could be plotted with spplot(sp).

prevR provides several continuous color palettes (see prevR.colors) compatible with spplot. Calculated surfaces could be export using the function writeAsciGrid{maptools}.

References


See Also

krige{gstat}, idw{gstat}, rings, prevR-method, Noptim.

Examples

```r
## Not run:
dhs <- rings(fdhs, N = c(100,200,300,400,500))
radius.N300 <- krig("r.radius", dhs, N = 300, nb.cells = 200)
prev.kriges <- krig(r.wprev ~ 1, dhs, N = c(100, 300, 500))
```
library(sp)
splot(prev.krige, c('r.wprev.N100.RInf', 'r.wprev.N300.RInf', 'r.wprev.N500.RInf'))

## End(Not run)

---

NA.outside.SpatialPolygons

Affect NA value to points located outside a polygon.

Description

This function forces points of an object of class `SpatialPixelsDataFrame` located outside the limits defined by an object of class `SpatialPolygons` to NA.

Usage

NA.outside.SpatialPolygons(sp.data, sp.poly)

Arguments

- **sp.data**: object of class `SpatialPixelsDataFrame`.
- **sp.poly**: object of class `SpatialPolygons`.

Value

Return `sp.data` modified.

See Also

- `point.in.SpatialPolygons`.

---

noptim

Suggested optimal value for N

Description

Based on previous simulation work, the function suggests an optimal value for the N parameter based on national prevalence, the total number of observations and the number of clusters. See Larmarange et al. 2011 for more details.

Usage

noptim(object)
 Arguments

 object object of class `prevR`.

 Value

 an integer.

 References


 Examples

 `Noptim(fdhs)`

---

plot.prevR,missing-method

*Plot object of class prevR.*

Description

Method `plot` for object of class `prevR`. Plot clusters, number of observations per cluster or number of positive cases per cluster.

Usage

```r
## S4 method for signature 'prevR,missing'
plot(x, type = "position", add.legend = TRUE,
     legend.location = "bottomright", factor.size = 0.2,
     new.window = FALSE, axes = F, ...)
```

Arguments

- `x` object of class `prevR`.
- `type` graph to plot:
  - "position" clusters position.
  - "c.type" clusters per `c.type`.
  - "count" number of observations per cluster.
  - "flower" number of positive cases per cluster.
- `add.legend` add a legend?
- `legend.location` legend location.
Description

This function tests if one or more points are located inside an object of class `SpatialPolygons`.

Usage

```r
point.in.SpatialPolygons(point.x, point.y, SpP)
```

Arguments

- `point.x` numerical array of x-coordinates of points.
- `point.y` numerical array of y-coordinates of points.
- `SpP` object of class `SpatialPolygons`

Value

Boolean array.
prevR-class

Objects of class prevR.

Description

Class used by the package prevR

Slots

clusters data.frame with observed data (one line per cluster). Columns names are:
- "id" cluster ID.
- "x" longitude.
- "y" latitude.
- "n" number of valid observations per cluster.
- "pos" number of positive cases per cluster.
- "prev" observed prevalence (in %) in the cluster (pos/n).
- "wn" (optional) sum of weights of observations per cluster.
- "wpos" (optional) sum of weights of positive cases per cluster.
- "wprev" (optional) weighted observed prevalence (in %) in the cluster (wpos/wn).
- "c.type" (optional) cluster type.

boundary object of class SpatialPolygons, borders of the studied area.

proj object of class CRS, map projection used.

rings list of results returned by rings. Each entry is composed of 3 elements: N, minimum number of observations per ring; R, maximum radius of rings and estimates, a data frame with the following variables:
- "id" cluster ID.
- "r.pos" number of positive cases inside the ring.
- "r.n" number of valid observations inside the ring.
- "r.prev" observed prevalence (in %) inside the ring (r.pos/r.n).
- "r.radius" ring radius (in kilometers if coordinates in decimal degrees, in the unit of the projection otherwise).
- "r.clusters" number of clusters located inside the ring.
- "r.wpos" (optional) sum of weights of positive cases inside the ring.
- "r.wn" (optional) sum of weights of valid observations inside the ring.
- "r.wprev" (optional) weighted observed prevalence (in %) inside the ring (r.wpos/r.wn).

Note: the list rings is named, the name of each element is NN_value.RR_value, for example N300.RInf.
prevR-class

Objects from the Class

Objects of this class could be created by the function `as.prevR`.

Methods

- `as.data.frame` signature(x = "prevR") converts an object of class prevR into a data frame.
- `as.SpatialGrid` signature(object = "prevR") generates a spatial grid.
- `export` signature(object = "prevR") exports a prevR object as a shapefile, a dbase file or a text file.
- `idw` signature(formula = "ANY", locations = "prevR") calculates a spatial interpolation using an inverse distance weighting.
- `kde` signature(object = "prevR") estimates a prevalence surface using kernel density estimators.
- `krige` signature(formula = "ANY", locations = "prevR") calculates a spatial interpolation by kriging.
- `plot` signature(x = "prevR", y = "ANY") plots data of a prevR object.
- `print` signature(x = "prevR") shows a summary of a prevR object.
- `rings` signature(object = "prevR") calculates rings of equal number of observations and/or equal radius.
- `show` signature(object = "prevR") shows a summary of a prevR object.
- `summary` signature(object = "prevR") shows a summary of the variables of a prevR object.
- `changeproj` signature(object = "prevR") changes the map projection used.

See Also


Examples

```r
showClass("prevR")

col <- c(id = "cluster",
     x = "x",
     y="y",
     n="n",
     pos = "pos",
     c.type = "residence",
     wn="weighted.n",
     wpos="weighted.pos"
)
dhs <- as.prevR(fdhs.clusters,col, fdhs.boundary)
str(dhs)
print(dhs)
```
Continuous color palettes.

Description

Functions generating color palettes useable with R graphical functions, in particular with `spplot`. These palettes are continuous, contrast being accentuated by darkening and lightening extrem values. `prevR.demo.pal` plot the available palettes. `prevR.colors.qgis.pal` export a palette in a text file readable by Quantum GIS, an open-source mapping software.

Usage

- `prevR.colors.blue(n)`
- `prevR.colors.blue.inverse(n)`
- `prevR.colors.gray(n)`
- `prevR.colors.gray.inverse(n)`
- `prevR.colors.green(n)`
- `prevR.colors.green.inverse(n)`
- `prevR.colors.red(n)`
- `prevR.colors.red.inverse(n)`
- `prevR.demo.pal(n, border = if (n < 32) "light gray" else NA, main = NULL)`
- `prevR.colors.qgis.pal(file, at, pal = "red", inverse = FALSE)`

Arguments

- `n` number of different colors in the palette.
- `border` border color.
- `main` title.
- `file` file name with extension.
at
list of values of the palette.

pal
color palette to use ("red", "green", "blue" or "gray").

inverse
use the inverse palette?

Details

prevR.colors.red produces a color gradation from white/yellow to red/dark red.
prevR.colors.blue produces a color gradation from light blue to dark blue.
prevR.colors.green produces a color gradation from light green to dark green.
prevR.colors.gray produces a color gradation from white/light gray to dark gray/black.

Functions with a suffix .inverse produce the same color gradation, but from dark colors to light ones.

Value

prevR.demo.pal plot the color palettes.
prevR.colors.qgis.pal export a color palette in a texte file readable by Quantum GIS.

The other functions return a list of colors coded in hexadecimal.

Note

To obtain the liste of colors in RGB (Red/Green/Blue), use the function col2rgb\{grDevices\}. The code of prevR.demo.pal was adapted from the function demo.pal presented in the examples of rainbow.

See Also

Other color palettes are available in R. See for example rainbow\{grDevices\} or the package RColorBrewer.

Examples

prevR.demo.pal(25)
prevR.colors.red(5)
col2rgb(prevR.colors.red(5))

## Not run:
prevR.colors.qgis.pal('palette.txt', seq(0,25,length.out=100), 'red')

## End(Not run)
print.prevR-method  Summary of a prevR object.

Description
Method print for objects of class prevR: shows a summary of the object’s characteristics.

Usage

```r
## S4 method for signature 'prevR'
print(x)
```

Arguments

- `x` object of class prevR.

Note
Exactly the same as show.prevR-method.

See Also

summary.prevR-method.

Examples

```r
print(fdhs)
## Not run:
dhs <- rings(fdhs, N=c(100,300,500))
print(dhs)
## End(Not run)
```

quick.prevR  Quick prevR analysis and plot

Description
This function performs several analysis in one go: (i) apply rings.prevR-method; (ii) compute prevalence surface with kde.prevR-method; (iii) compute the surface of rings radii with krig,ANY.prevR-method; (iv) plot prevalence surface using prevR.colors.red and add rings radii as a contour plot.
Usage

quick.prevR(object, N = Noptim(object), nb.cells = 100,
            cell.size = NULL, weighted = NULL, plot.results = TRUE,
            return.results = FALSE, return.plot = FALSE, legend.title = "%",
            cex = 0.7, progression = TRUE)

Arguments

object object of class prevR.
N integer or list of integers corresponding to the rings to use.
nb.cells number of cells on the longest side of the studied area (unused if cell.size is defined).
cell.size size of each cell (in the unit of the projection).
weighted use weighted data (TRUE, FALSE or "2")?
plot.results plot the results?
return.results return the results?
return.plot return the plot within the results?
legend.title title of the legend
cex to control the text size on the graph
progression show a progress bar?

Details

N determine the rings to use for the estimation. By default, a suggested value of N will be computed with Noptim.

Value

A list of one or several elements, depending on the arguments: (i) prev is a SpatialPixelsDataFrame containing the prevalence surface; (ii) radius a SpatialPixelsDataFrame containing the kriged surface of the rings radii; (iii) plot a ggplot graph.

See Also

Noptim, rings, prevR-method, kde, prevR-method, krige, ANY, prevR-method.

Examples

## Not run:
quick.prevR(fdhs)

## End(Not run)
rings.prevR-method  

Calculation of rings of equal number of observation and/or equal radius.

Description

For each cluster, this function determines a ring of equal number of observations and/or equal radius and calculates several indicators from observations located inside that ring.

Usage

```r
## S4 method for signature 'prevR'
rings(object, N = seq(100, 500, 50), R = Inf,
       progression = TRUE)
```

Arguments

- `object` object of class `prevR`.
- `N` minimum number of observations.
- `R` maximum rings radius (in kilometers if coordinates in decimal degrees, in the unit of the projection otherwise).
- `progression` show a progress bar?

Details

For each ligne of the data frame `clusters` of `object`, `rings` determines a ring, centred on the cluster. It could be:

- rings of equal number of observations if N is finite and R=Inf;
- rings of equal radius if N=Inf and R is finite;
- a combination of both (see below) if N and R are finite.

For **rings of equal number of observations**, `rings` selects the smallest ring containing at least N valid observations.
For **rings of equal radius**, `rings` selects all clusters located at a lower distance than R from the central cluster.
For **combination of both**, `rings` calculates firts the ring with the minimum number of observations and test if its radius is lower than R or not. If so, the ring is kept, otherwise the ring of maximum radius is calculated.

Different series of rings could be simultaneously calculated by providing different values for N and R. `rings` will calculate rings corresponding to each couple (N,R).
Value

Return object with the slot `rings` completed for each couple (N,R).

Each entry is composed of 3 elements: N, minimum number of observations per ring; R, maximum radius of rings and estimates, a data frame with the following variables:

- "id" cluster ID.
- "r.pos" number of positive cases inside the ring.
- "r.n" number of valid observations inside the ring.
- "r.prev" observed prevalence (in %) inside the ring (r.pos/r.n).
- "r.radius" ring radius (in kilometers if coordinates in decimal degrees, in the unit of the projection otherwise).
- "r.clusters" number of clusters located inside the ring.
- "r.wpos" (optional) sum of weights of positive cases inside the ring.
- "r.wn" (optional) sum of weights of valid observations inside the ring.
- "r.wprev" (optional) weighted observed prevalence (in %) inside the ring (r.wpos/r.wn).

Note: the list `rings` is named, the name of each element is N_value.R_value, for example N300.RInf.

Note 2: `r.wpos`, `r.wn` and `r.wprev` are calculated only if the slot `clusters` of object contains weighted data.

References


See Also

`prevR-class`.

Examples

```r
## Not run:
print(fdhs)
dhs <- rings(fdhs,N=c(100,200,300,400,500))
print(dhs)
## End(Not run)
```
**show.prevR-method**  
*Summary of a prevR object.*

**Description**  
Method `show` for objects of class `prevR`: shows a summary of the object’s characteristics.

**Usage**  
```r
## S4 method for signature 'prevR'
show(object)
```

**Arguments**  
- `object`  
  object of class `prevR`.

**Note**  
Exactly the same as `print.prevR-method`.

**See Also**  
`summary.prevR-method`.

**Examples**

```r
defhs  
## Not run:  
dhs <- rings(fdhs,N=c(100,300,500))  
dhs

## End(Not run)
```

---

**summary.prevR-method**  
*Detailed summary of the variables of a prevR object*

**Description**  
Method `summary` for objects of class `prevR`: shows a summary of the variables of the object.

**Usage**  
```r
## S4 method for signature 'prevR'
summary(object, probs = c(0, 0.1, 0.25, 0.5, 0.75, 0.8, 0.9, 0.95, 0.99, 1))
```
Arguments

object: object of class prevR.
probs: vector of probabilities with values in [0,1] for computing quantiles of the rings radii (see examples).

See Also

print.prevR-method.

Examples

summary(fdhs)
## Not run:
dhs <- rings(fdhs, N=c(100,300,500))
summary(dhs)
summary(dhs, c(0, 0.25, 0.5, 0.75, 1))
## End(Not run)

theme_prevR prevR themes for ggplot2

Description

Two custom themes for ggplot2 graphs, hiding axis.

Usage

theme_prevR(base_size = 12)

theme_prevR_light(base_size = 12)

Arguments

base_size: base font size

See Also

ggtheme(ggplot2)
Description

This dataset provides boundaries of all countries in the world, in decimal degrees. Available variables are:

- "FIPS" FIPS 10-4 Country Code.
- "ISO3" ISO 3166-1 Alpha-3 Country Code.
- "UN" ISO 3166-1 Numeric-3 Country Code.
- "NAME" Name of country/area.
- "REGION" Macro geographical (continental region), UN Statistics.
- "SUBREGION" Geographical sub-region, UN Statistics.
- "LON" Longitude.
- "LAT" Latitude.

Format

Object of class **SpatialPolygonsDataFrame**.

Note

The boundaries, names designations used do not imply official endorsement or acceptance by the authors. Use this dataset with care, as several of the borders are disputed.

Source

Provided by Bjorn Sandvik on [http://thematicmapping.org/downloads/worldBorders.php](http://thematicmapping.org/downloads/worldBorders.php). The dataset was derived by Schuyler Erle from public domain sources. Sean Gilles did some clean up and made some enhancements. The dataset is available under a Creative Commons Attribution-Share Alike License ([http://creativecommons.org/licenses/by-sa/3.0/](http://creativecommons.org/licenses/by-sa/3.0/)).

Examples

```r
plot(TMWorldBorders)
```
xyz2dataframe

Convert a surface in xyz to a data frame.

Description

Several functions (for example KernSur[GenKern]) return a surface as a list "xyz" composed of three elements: vector of ordinates in the x dimension, vector of ordinates in the y dimension and a matrix with the values of the surface in x and y. This function transforms a list "xyz" into a data frame.

Usage

xyz2dataframe(xyz, xcol = 1, ycol = 2, zcol = 3)

Arguments

xyz                a list with 3 elements: a vector with x-coordinates, a vector with y-coordinates and a matrix with value for each point of coordinates x[i],y[j].

xcol               x index.

ycol               y index.

zcol               z index.

Value

A data.frame.

Note

xyz could be a list like x,y,z1,z2,z3. If so, zcol should be equal to c("z1","z2","z3") or c(3,4,5).

Examples

```r
x <- c(2,4,6,8,10)
y <- x
op <- GenKern::KernSur(x,y, xgridsize=50, ygridsize=50,
                       correlation=0,
                       xbandwidth=1, ybandwidth=1,
                       range.x=c(0,13), range.y=c(0,13))
str(op)

op.df <- xyz2dataframe(op)
str(op.df)
```
Index

*Topic **classes**
  prevR-class, 24
*Topic **class**
  is.prevR, 15
*Topic **color**
  prevR.colors, 26
*Topic **datasets**
  fdhs, 14
  TMWorldBorders, 34
*Topic **hplot**
  plot,prevR,missing-method, 22
*Topic **manip**
  as.data.frame.prevR, 6
  as.prevR, 7
  as.SpatialGrid,prevR-method, 8
  changeproj,prevR-method, 9
  create.boundary, 10
  export,prevR-method, 12
  import.dhs, 14
  NA.outside.SpatialPolygons, 21
  point.in.SpatialPolygons, 23
  xyz2dataframe, 35
*Topic **math**
  rings,prevR-method, 30
*Topic **package**
  prevR-package, 3
*Topic **plot**
  quick.prevR, 28
*Topic **smooth**
  kde,prevR-method, 16
  krige,ANY,prevR-method, 19
  quick.prevR, 28
*Topic **spatial**
  as.SpatialGrid,prevR-method, 8
  changeproj,prevR-method, 9
  create.boundary, 10
  export,prevR-method, 12
  kde,prevR-method, 16
  krige,ANY,prevR-method, 19
  NA.outside.SpatialPolygons, 21
  point.in.SpatialPolygons, 23
  quick.prevR, 28
  rings,prevR-method, 30
  TMWorldBorders, 34
  xyz2dataframe, 35
*Topic **stat**
  Noptim, 21
  as.data.frame, 4, 6
  as.data.frame(as.data.frame.prevR), 6
  as.data.frame.prevR, 6, 13
  as.prevR, 3, 7, 14, 15, 25
  as.SpatialGrid, 20
  as.SpatialGrid
    (as.SpatialGrid,prevR-method), 8
  as.SpatialGrid,prevR-method, 8
  as.SpatialGrid-methods
    (as.SpatialGrid,prevR-method), 8
  changeproj, 4, 8
  changeproj(changeproj,prevR-method), 9
  changeproj,prevR-method, 9
  changeproj-methods
    (changeproj,prevR-method), 9
  col2rgb, 27
  create.boundary, 3, 8, 10
  CRS, 8, 10, 11, 24
  direct.label.ggplot, 12
  direct.label_prevR, 11
  export, 4
  export(export,prevR-method), 12
  export,prevR-method, 12
  export-methods(export,prevR-method), 12
  eyefit, 20
  fdhs, 3, 14

36