Package 'ggOceanMaps'

January 14, 2021

Type Package

Title Plot Data on Oceanographic Maps using 'ggplot2'

Version 1.0.9

Date 2021-01-08

URL https://mikkovihtakari.github.io/ggOceanMaps/

BugReports https://github.com/MikkoVihtakari/ggOceanMaps/issues

Description Allows plotting data on bathymetric maps using 'ggplot2'. Plotting oceanographic spatial data is made as simple as feasible, but also flexible for custom modifications. Data that contain geographic information from anywhere around the globe can be plotted on maps generated by the basemap() function using 'ggplot2' layers separated by the '+' operator. The package uses spatial shapefiles stored in the 'ggOceanMapsData' package, geospatial packages for R to manipulate, and the 'ggspatial' package to help to plot these shapefiles.

Depends R (>= 3.5.0), ggplot2, ggspatial

Imports sp, raster, rgdal, rgeos, methods, utils, sf, stars, smoothr, units, dplyr, parallel

Suggests ggOceanMapsData, cowplot, knitr, rmarkdown, scales

Additional_repositories https://mikkovihtakari.github.io/drat

License GPL-3

Encoding UTF-8

LazyData true

RoxygenNote 7.1.1

VignetteBuilder knitr

NeedsCompilation no

Author Mikko Vihtakari [aut, cre] (Institute of Marine Research, <https://orcid.org/0000-0003-0371-4319>),
Hadley Wickham [ctb],
Simon O'Hanlon [ctb],
Roger Bivand [ctb]

Maintainer Mikko Vihtakari <mikko.vihtakari@hi.no>

Repository CRAN

Date/Publication 2021-01-14 09:40:05 UTC

R topics documented:

basemap 2
dist2land 7
qmap 10
raster_bathymetry 12
reorder_layers 14
shapefile_list 14
theme_map 15
transform_coord 16
vector_bathymetry 18

Index **19**

---

basemap *Create a ggplot2 basemap for plotting variables*

Description

Creates a ggplot2 basemap for further plotting of variables.

Usage

```
basemap(
  limits = NULL,
  data = NULL,
  shapefiles = NULL,
  bathymetry = FALSE,
  glaciers = FALSE,
  resolution = "low",
  rotate = FALSE,
  legends = TRUE,
  legend.position = "right",
  lon.interval = NULL,
  lat.interval = NULL,
  bathy.style = "poly_blues",
  bathy.border.col = NA,
  bathy.size = 0.1,
  land.col = "grey60",
  land.border.col = "black",
  land.size = 0.1,
  gla.col = "grey95",
```

```
  gla.border.col = "black",
  gla.size = 0.1,
  grid.col = "grey70",
  grid.size = 0.1,
  base_size = 11,
  projection.grid = FALSE,
  verbose = TRUE
)
```

Arguments

`limits`	Map limits. One of the following: • **numeric vector** of length 4: The first element defines the start longitude, the second element the end longitude (counter-clockwise), the third element the minimum latitude and the fourth element the maximum latitude of the bounding box. The coordinates can be given as decimal degrees or coordinate units for shapefiles used by a projected map. Produces a rectangular map. Latitude limits not given in min-max order are automatically ordered to respect this requirement. • **single integer** between 30 and 88 or -88 and -30 produces a polar map for the Arctic or Antarctic, respectively. Can be omitted if `data` or `shapefiles` are defined.
`data`	Data frame or SpatialPolygons shape containing longitude and latitude coordinates in decimal degrees. The limits are extracted from these coordinates and produces a rectangular map. Suited for situations where a certain dataset is plotted on a map. The function attempts to guess the correct columns and it is advised to use intuitive column names for longitude (such as "lon", "long", or "longitude") and latitude ("lat", "latitude") columns. Can be omitted if `limits` or `shapefiles` are defined.
`shapefiles`	Either a list containing shapefile information or a character argument referring to a name of pre-made shapefiles in `shapefile_list`. This name is partially matched. Can be omitted if `limits` or `data` are defined as decimal degrees.
`bathymetry`	Logical indicating whether bathymetry should be added to the map.
`glaciers`	Logical indicating whether glaciers and ice-sheets should be added to the map.
`resolution`	Not implemented yet.
`rotate`	Logical indicating whether the projected maps should be rotated to point towards the pole relative to mid-longitude limit. Experimental.
`legends`	Logical indicating whether the legend for bathymetry should be shown.
`legend.position`	The position for ggplot2 legend. See the argument with the same name in theme.
`lon.interval, lat.interval`	Numeric value specifying the interval of longitude and latitude grids. `NULL` finds reasonable defaults depending on `limits`.
`bathy.style`	Character defining the style for bathymetry contours. Alternatives: • `"poly_blues"` plots polygons filled with different shades of blue.

- "poly_greys" plots polygons filled with different shades of gray.
- "contour_blues" contour lines with different shades of blue.
- "contour_grey" plots gray contour lines.

land.col, gla.col, grid.col
: Character code specifying the color of land, glaciers and grid lines, respectively. Use NA to remove the grid lines.

land.border.col, gla.border.col, bathy.border.col
: Character code specifying the color of the border line for land, glacier, and bathymetry shapes.

land.size, gla.size, bathy.size, grid.size
: Numeric value specifying the width of the border line land, glacier and bathymetry shapes as well as the grid lines, respectively. Use the LS function for a specific width in pt. See Details.

base_size
: Base size parameter for ggplot. See ggtheme.

projection.grid
: Logical indicating whether the coordinate grid should show projected coordinates instead of decimal degree values. Useful to define limits for large maps in polar regions.

verbose
: Logical indicating whether information about the projection and guessed column names should be returned as message. Set to FALSE to make the function silent.

Details

The function uses ggplot2, ggspatial, GIS packages of R, and shapefiles to plot maps of the world's oceans.

Projections

If the shapefiles are not specified, the function uses either the limits or data arguments to decide which projection to use. Up-to-date conditions are defined in define_shapefiles and shapefile_list functions. At the time of writing, the function uses three different projections (given as EPSG codes)

- **3995** WGS 84 / Arctic Polar Stereographic. Called "ArcticStereographic". For max latitude (limits[4]) >= 60 (if min latitude (limits[3]) >= 30), and single integer latitudes >= 30 and <= 89.
- **3031** WGS 84 / Antarctic Polar Stereographic. Called "AntarcticStereographic". For max latitude (limits[4]) <= -60 (if min latitude (limits[3]) <= -30), and single integer latitudes <= -30 and >= -89.
- **4326** WGS 84 / World Geodetic System 1984, used in GPS. Called "DecimalDegree". For min latitude (limits[3]) < 30 or > -30, max latitude (limits[4]) < 60 or > -60, and single integer latitudes < 30 and > -30.

Limits

If the limits are in decimal degrees, the longitude limits ([1:2]) specify the start and end segments of corresponding angular lines that should reside inside the map area. The longitude limits are defined **counter-clockwise**. The latitude limits [3:4] define the parallels that should reside inside

the limited region given the longitude segments. Note that the actual limited region becomes wider than the polygon defined by the coordinates (shown in Examples). Using `data` to limit the map expands the map all around the data points to make them fit into the map. If the limits are given as projected coordinates or as decimal degrees for maps with -60 < latitude < 60, limits elements represent lines encompassing the map area in cartesian space.

Pre-made shapefiles

If the limits are not defined as decimal degrees (any longitude outside range [-180, 180] or latitude [-90, 90]), the function will ask to specify `shapefiles`. The `shapefiles` can be defined by partially matching the names of the pre-made shapefiles in `shapefile_list` (e.g. "Ar" would be enough for "ArcticStereographic") or by specifying custom shapefiles.

Custom shapefiles

Custom shapefiles have to be a named list containing at least following elements:

- **land** Object name of the `SpatialPolygonsDataFrame` containing land. Required.
- **glacier** Object name of the `SpatialPolygonsDataFrame` containing glaciers. Use `NULL` if glaciers are not needed.
- **bathy** Object name of the `SpatialPolygonsDataFrame` containing bathymetry contours. Use `NULL` if bathymetry is not needed.

See Examples.

Line width and font size

The line size aesthetics in ggplot2 generates approximately 2.13 wider lines measured in pt than the given values. If you want a specific line width in pt, use the internal function `LS` to convert the desired line width to ggplot2 equivalent. A similar function is also available for font sizes (`FS`).

CRS warnings

Open-source GIS systems are rolling over to a new to a new projection definition system. The changes to underlying systems appear to sometimes trigger warnings the user can ignore as long as the resulting map looks OK. Bug reports regarding these warnings are appreciated.

Value

Returns a ggplot map, which can be assigned to an object and modified as any ggplot object.

Author(s)

Mikko Vihtakari

References

Note that if you use this function to generate maps for a publication, it is advised to cite the underlying data. The spatial data used by this function have been acquired from following sources:

- **Land polygons.** Natural Earth Data 1:10m Physical Vectors with the Land and Minor Island datasets combined. Distributed under the CC Public Domain license (terms of use).
- **Glacier polygons.** Natural Earth Data 1:10m Physical Vectors with the Glaciated Areas and Antarctic Ice Shelves datasets combined. Distributed under the CC Public Domain license (terms of use)

- **Bathymetry.** Amante, C. and B.W. Eakins, 2009. ETOPO1 1 Arc-Minute Global Relief Model: Procedures, Data Sources and Analysis. NOAA Technical Memorandum NESDIS NGDC-24. National Geophysical Data Center, NOAA. Distributed under the U.S. Government Work license.

See Also

ggplot

Other basemap functions: qmap(), shapefile_list(), transform_coord()

Examples

```
# The easiest way to produce a map is to use the limits
# argument and decimal degrees:

if(requireNamespace("ggOceanMapsData")) {
basemap(limits = 60)
}

# Bathymetry and glaciers can be added using the respective arguments:

basemap(limits = -60, bathymetry = TRUE, glaciers = TRUE)

# The easiest way to add data on the maps is to use the ggspatial functions:

dt <- data.frame(lon = c(-150, 150), lat = c(60, 90))

basemap(data = dt, bathymetry = TRUE) +
geom_spatial_point(data = dt, aes(x = lon, y = lat), color = "red")

## Not run:
# Note that writing out data = dt is required because there are multiple
# underlying ggplot layers plotted already:
basemap(data = dt) +
geom_spatial_point(dt, aes(x = lon, y = lat), color = "red")
#> Error: `mapping` must be created by `aes()`

## End(Not run)

# If you want to use native ggplot commands, you need to transform your data
# to the projection used by the map:

dt <- transform_coord(dt, bind = TRUE)

if(requireNamespace("ggOceanMapsData")) {
basemap(data = dt) + geom_point(data = dt, aes(x = lon.proj, y = lat.proj))
}

# The limits argument of length 4 plots a map anywhere in the world:

basemap(limits = c(100, 160, -20, 30), bathymetry = TRUE)
```

```
# The argument leads to expanded maps towards poles:

dt <- data.frame(lon = c(-160, 160, 160, -160), lat = c(80, 80, 60, 60))

basemap(limits = c(160, -160, 60, 80)) +
geom_spatial_polygon(data = dt, aes(x = lon, y = lat),
fill = NA, color = "red")

# The limits are further expanded when using the data argument:

basemap(data = dt) +
geom_spatial_polygon(data = dt, aes(x = lon, y = lat),
fill = NA, color = "red")

# Rotate:

basemap(data = dt, rotate = TRUE) +
geom_spatial_polygon(data = dt, aes(x = lon, y = lat),
                     fill = NA, color = "red")

## To find UTM coordinates to limit a polar map:
basemap(limits = 60, projection.grid = TRUE)
basemap(limits = c(2.5e4, -2.5e6, 2e6, -2.5e5), shapefiles = "Arctic")

# Using custom shapefiles
data(bs_shapes, package = "ggOceanMapsData")
basemap(shapefiles = list(land = bs_land, glacier = NULL, bathy = bs_bathy),
bathymetry = TRUE)

# grid.col = NA removes grid lines, rotate = TRUE rotates northwards

basemap(limits = c(-180, -140, 50, 70), grid.col = NA, rotate = TRUE)

# Rename axis labels

basemap(limits = c(-140, -105, 20, 40), bathymetry = TRUE) + xlab("Lat")

# Remove axis labels

basemap(limits = c(0, 60, 68, 80)) + labs(x = NULL, y = NULL)

basemap(limits = c(0, 60, 68, 80), rotate = TRUE) +
theme(axis.title = element_blank(),
      axis.text = element_blank(),
      axis.ticks.x = element_blank(),
      axis.ticks.y = element_blank()
      )
```

`dist2land` *Calculate distance to the closest land for coordinates in a data frame*

Description

Calculates the closest distance to land for coordinates in a data frame

Usage

```
dist2land(
  data,
  lon = NULL,
  lat = NULL,
  shapefile = NULL,
  proj.in = "+init=epsg:4326",
  bind = TRUE,
  dist.col = "ldist",
  binary = FALSE,
  cores = getCores(),
  verbose = TRUE
)
```

Arguments

data	Data.frame containing geographic coordinates
lon, lat	Either the names of the longitude and latitude columns in data or NULL to guess the longitude and/or latitude columns in data.
shapefile	Land shape to which distances should be calculated. Either a character argument referring to a name of pre-made shapefiles in shapefile_list, a single SpatialPolygons object or NULL to enable automatic definition of the land shapes based on data.
proj.in	proj4string projection argument for the coordinates in data.
bind	Logical indicating whether x should be returned with the distances (TRUE, default) or should the distances be returned as vector (FALSE).
dist.col	The name of the distance column, if bind = TRUE. Defaults to "dist".
binary	Logical indicating whether binary (TRUE = the position is in the ocean, FALSE = the position is on land) should be returned instead of distances. Speeds up the function considerably.
cores	Integer value defining how many cores should be used in the distance calculations. Parallelization speeds up the function (see parallel::mclapply), but naturally eats up computer resources during the calculation. Set to 1 to remove parallelization.
verbose	Logical indicating whether information about the process should be returned as messages. Set to FALSE to make the function silent.

Details

The function calculates distances using projected coordinates and the rgeos::gDistance function. These distances do not consider the curvature of the Earth unless the projection of the used land shape does so (check out geosphere::dist2Line and this SO solution if you want exact distances).

The function is fairly slow for large datasets. If you only want to use the function to remove (wrong) observations reported on land, set the `binary` argument to `TRUE`. This speeds up the calculations considerably.

The `dist2land` function offers parallel processing, which speeds up the calculations for large datasets. Parallel processing has not been tested under Windows yet and may not work.

Value

Returns a vector if `bind = FALSE`, otherwise a data frame. The distances are given in a new column defined by the `dist.col` argument. The distances are **kilometers** if `binary = FALSE`, otherwise logical (TRUE = the position is in the ocean, FALSE = the position is on land).

Author(s)

Mikko Vihtakari

Examples

```
# Simple example:
dt <- data.frame(lon = seq(-20, 80, length.out = 41), lat = 50:90)
dt <- dist2land(dt, cores = 1)
qmap(dt, color = ldist) + scale_color_viridis_c()

# No premade shapefiles for datasets covering the entire globe
data.frame(lon = -20:20, lat = seq(-90, 90, length.out = 41))
dist2land(dt, cores = 1) # wrong!

## Not run:
dt <- data.frame(lon = seq(-179, 179, length.out = 1000), lat = rep(60, 1000))
# The distance calculation is slow for large datasets
system.time(dist2land(dt))
#> user  system elapsed
#> 0.073   0.041   5.627

# The parallel processing speeds it up
system.time(dist2land(dt, cores = 1))
#> user  system elapsed
#> 19.719   1.237  20.894

# binary = TRUE further speeds the function up
system.time(dist2land(dt, binary = TRUE))
#> user  system elapsed
#> 1.624   0.041   1.680

## End(Not run)
```

qmap *Quick map*

Description

qmap is a shortcut similar to ggplot2's qplot designed to quickly plot data with a limited range of options.

Usage

```
qmap(
  data,
  x = NULL,
  y = NULL,
  geom = "point",
  limits = NULL,
  bathymetry = FALSE,
  glaciers = FALSE,
  resolution = "low",
  rotate = FALSE,
  legends = TRUE,
  legend.position = "right",
  lon.interval = NULL,
  lat.interval = NULL,
  bathy.style = "poly_blues",
  bathy.border.col = NA,
  bathy.size = 0.1,
  land.col = "grey60",
  land.border.col = "black",
  land.size = 0.1,
  gla.col = "grey95",
  gla.border.col = "black",
  gla.size = 0.1,
  grid.col = "grey70",
  grid.size = 0.1,
  base_size = 11,
  projection.grid = FALSE,
  verbose = FALSE,
  ...
)
```

Arguments

data	Data frame to use.
x, y, ...	Aesthetics passed into each layer. Longitude and latitude columns are automatically recognized using the guess_coordinate_columns function.

`geom` Character argument specifying geom(s) to draw. Defaults to "point". Other geoms have not been implemented yet.

`limits` Map limits. See the `limits` argument in `basemap`. If `NULL` the limits are automatically taken from `data`

`bathymetry` Logical indicating whether bathymetry should be added to the map.

`glaciers` Logical indicating whether glaciers and ice-sheets should be added to the map.

`resolution` Not implemented yet.

`rotate` Logical indicating whether the projected maps should be rotated to point towards the pole relative to mid-longitude limit. Experimental.

`legends` Logical indicating whether the legend for bathymetry should be shown.

`legend.position`
The position for ggplot2 legend. See the argument with the same name in theme.

`lon.interval, lat.interval`
Numeric value specifying the interval of longitude and latitude grids. `NULL` finds reasonable defaults depending on `limits`.

`bathy.style` Character defining the style for bathymetry contours. Alternatives:

- `"poly_blues"` plots polygons filled with different shades of blue.
- `"poly_greys"` plots polygons filled with different shades of gray.
- `"contour_blues"` contour lines with different shades of blue.
- `"contour_grey"` plots gray contour lines.

`land.col, gla.col, grid.col`
Character code specifying the color of land, glaciers and grid lines, respectively. Use `NA` to remove the grid lines.

`land.border.col, gla.border.col, bathy.border.col`
Character code specifying the color of the border line for land, glacier, and bathymetry shapes.

`land.size, gla.size, bathy.size, grid.size`
Numeric value specifying the width of the border line land, glacier and bathymetry shapes as well as the grid lines, respectively. Use the `LS` function for a specific width in pt. See Details.

`base_size` Base size parameter for ggplot. See ggtheme.

`projection.grid`
Logical indicating whether the coordinate grid should show projected coordinates instead of decimal degree values. Useful to define limits for large maps in polar regions.

`verbose` Logical indicating whether information about the projection and guessed column names should be returned as message. Set to `FALSE` to make the function silent.

Value

Returns a ggplot map, which can be assigned to an object and modified as any ggplot object.

Author(s)

Mikko Vihtakari

See Also

Other basemap functions: basemap(), shapefile_list(), transform_coord()

Examples

```
dt <- data.frame(lon = c(-100, -80, -60), lat = c(10, 25, 40), var = c("a", "a", "b"))

# Set color

if(requireNamespace("ggOceanMapsData")) {
qmap(dt, color = I("red"))
}
# Map color

qmap(dt, color = var)


dt <- data.frame(lon = c(-80, -80, -50, -50), lat = c(65, 80, 80, 65))

if(requireNamespace("ggOceanMapsData")) {
qmap(dt, rotate = TRUE)
}
```

---

raster_bathymetry *Simplify a bathymetry raster ready to be vectorized*

Description

Simplifies bathymetry raster ready for the vector_bathymetry function. Warning: processing may take a long time if the bathymetry raster is large.

Usage

```
raster_bathymetry(
  bathy,
  depths,
  proj.out,
  boundary = NULL,
  file.name = NULL,
  aggregation.factor = NA
)
```

Arguments

bathy	A raster object or a string giving the path to a bathymetry NetCDF or grd file
depths	Numeric vector giving the cut points for depth contours (see cut.
proj.out	A character string specifying the PROJ.4 projection arguments for the output. See CRS and proj.org.
boundary	A SpatialPolygons(DataFrame) object, text string defining the file path to a spatial polygon, or a numeric vector of length 4 giving the boundaries for which bathy should be cut to. Should be given as **decimal degrees**. If numeric vector, the first element defines the minimum longitude, the second element the maximum longitude, the third element the minimum latitude and the fourth element the maximum latitude of the bounding box. Use NULL not to cut bathy.
file.name	A character string specifying the file path **without extension** where the output should be saved. If NULL a temporary file will be used. See writeRaster.
aggregation.factor	An integer defining the fact argument from the aggregate function. Set to NA to ignore aggregation.

Details

You can use GEBCO or ETOPO1 bathymetry grids downloaded from respective sources as the bathy argument. The bathymetry grids read from files must be in NetCDF/grd format and defined using decimal degrees. Alternatively use the marmap::getNOAA.bathy function to download ETOPO1 bathymetry and convert it to a raster object using the marmap::as.raster function.

Note that the size of the output is heavily influenced by the number of depth contours (depths) as well as the resolution of bathy and choise of aggregation.factor. To make the vector_bathymetry function and consequent plotting faster, limiting the details of the bathymetry raster may be desireable.

Value

A list with raster object containing projected bathymetry defined by the proj.out argument and a data frame of depth intervals.

Author(s)

Mikko Vihtakari

References

GEBCO Compilation Group (2019) GEBCO 2019 15-arcsecond grid (doi:10.5285/836f016a-33be-6ddc-e053-6c86abc0788e). URL: https://www.gebco.net/data_and_products/gridded_bathymetry_data/gebco_2019/gebco_2019_info.html. ETOPO1 1 Arc-Minute Global Relief Model. URL: https://www.ngdc.noaa.gov/mgg/global/relief/ETOPO1/docs/ETOPO1.pdf.

See Also

Other create shapefiles: clip_shapefile(), vector_bathymetry()

reorder_layers *Move* basemap *land, glacier and grid layers on top of other ggplot layers*

Description

Moves existing land, glacier and grid layers on top of other layers. Useful for hiding region polygons under land.

Usage

```
reorder_layers(p)
```

Arguments

p	ggplot object from the basemap function.

Details

This function has not been tested properly yet and is likely to contain bugs.

Value

Returns a ggplot object with land, glacier and grid layers on top.

Author(s)

Mikko Vihtakari

See Also

Other customize shapefiles: auto_limits(), theme_map()

shapefile_list *A list of pre-made shapefiles for* basemap

Description

Lists available pre-made shapefiles for plotting in the basemap function. Gives also instructions how to make custom ones.

Usage

```
shapefile_list(name, get.data = FALSE)
```

Arguments

name	A character argument giving the name of a pre-made shapefile. Will be partially matched. Use "all" to list all available ones.
get.data	Logical indicating whether spatial data should be returned instead of names of spatial data objects.

Details

Custom shapefiles for basemap should be defined as lists with (at least) following names (everything should be provided as characters):

- **land** Object name of the SpatialPolygonsDataFrame containing land. Required.
- **glacier** Object name of the SpatialPolygonsDataFrame containing glaciers. Use NULL if glaciers are not needed.
- **bathy** Object name of the SpatialPolygonsDataFrame containing bathymetry contours. Use NULL if bathymetry is not needed.

All linked spatial data objects must be in same projection. Pre-made shapefiles contain additional elements that are used in the basemap function, but not required for custom shapefile datasets.

Value

Returns a data frame of provided pre-made shapefiles, if name = "all". Returns a shapefile list containing the information for a particular map otherwise.

Author(s)

Mikko Vihtakari

See Also

Other basemap functions: basemap(), qmap(), transform_coord()

Examples

```
shapefile_list("all")
shapefile_list("Arctic") # partial matching
```

---

theme_map *A ggplot2 theme for maps*

Description

A ggplot2 theme for maps.

Usage

```
theme_map(..., grid.col, grid.size)
```

Arguments

...	additional arguments passed to ggtheme.
grid.col	Character code specifying the color of grid lines. Use NA to remove the grid lines.
grid.size	Numeric value specifying the width of grid lines.

Value

A ggplot2 theme layer.

See Also

Other customize shapefiles: auto_limits(), reorder_layers()

---

transform_coord *Transform spatial coordinates to another projection*

Description

Transforms spatial coordinates from original projection (decimal degrees assumed) to another projection.

Usage

```
transform_coord(
  x = NULL,
  lon = NULL,
  lat = NULL,
  new.names = "auto",
  proj.in = "+init=epsg:4326",
  proj.out = NULL,
  verbose = FALSE,
  bind = FALSE,
  na = "ignore"
)
```

Arguments

x	Data frame to be transformed. Can be omitted if numeric vectors are assigned to lon and lat.
lon, lat	Either a name of the longitude and latitude columns in x or a numeric vector containing longitude and latitude coordinates. Use NULL to guess the longitude and/or latitude columns in x.
new.names	Character vector of length 2 specifying the names of transformed longitude and latitude columns, respectively. Alternatively NULL, which returns column names from x or "auto", which uses NULL if bind = FALSE and c("lon.proj","lat.proj") if bind = TRUE.

proj.in	Original proj4string projection. If NULL, the projection is taken from x. x must be a Spatial object in that case.
proj.out	Character. Either NULL, proj4string projection the coordinates should be transformed to or a name of shapefiles in shapefile_list. If NULL, the output projection will be automatically determined from data. This option requires decimal degrees as input option.
verbose	Logical indicating whether information about the projection should be returned as message. Set to FALSE to make the function silent.
bind	logical. Should only transformed coordinates be returned (FALSE, default) or should x be returned with transformed coordinates (TRUE)?
na	character specifying the NA action for missing coordinates. The "ignore" option ignores the coordinates and returns NAs to transformed coordinates. The "remove" option removes missing values from x returning a message while doing it. Any other character argument will trigger na.fail stopping the function in case of missing coordinates.

Details

If x is specified, the function guesses longitude and latitude columns from x by default.

Value

Returns a data frame with transformed spatial coordinates.

Author(s)

Mikko Vihtakari

See Also

Other basemap functions: basemap(), qmap(), shapefile_list()

Examples

```
# Coordinates are automatically transformed to the pre-made shapefile
# projections:
dt <- data.frame(lon = c(-150, 150), lat = c(60, 90))
transform_coord(dt)
transform_coord(dt, bind = TRUE)

dt <- data.frame(lon = c(-150, 150), lat = c(20, 50))
transform_coord(dt, bind = TRUE) # no transformation required.
```

`vector_bathymetry` *Create a SpatialPolygonsDataFrame bathymetry from a raster bathymetry file*

Description

Vectorizes bathymetry rasters. Designed to be used for the output of `raster_bathymetry` function. Warning: processing may take a long time if the bathymetry raster is large.

Usage

```
vector_bathymetry(
  bathy,
  drop.crumbs = NULL,
  remove.holes = NULL,
  smooth = FALSE
)
```

Arguments

`bathy`	bathyRaster object from the `raster_bathymetry` function.
`drop.crumbs`	Single numeric value specifying a threshold (area in km2) for disconnected polygons which should be removed. Set to `NULL` to bypass the removal. Uses the drop_crumbs function.
`remove.holes`	Single numeric value specifying a threshold (area in km2) for holes which should be removed. Set to `NULL` to bypass the removal. Uses the fill_holes function.
`smooth`	Logical indicating whether the pixelated contours should be smoothed. Uses the smooth_ksmooth function.

Details

The `drop.crumbs` and `remove.holes` arguments can be used to make the resulting object smaller in file size. The `smooth` argument can be used to remove the pixelated contours, but often increases file size. Note also that using this option will bias the contours with respect to real world.

Value

SpatialPolygonsDataFrame containing the depth polygons. Uses same projection than `bathy` (see `CRS`).

Author(s)

Mikko Vihtakari

See Also

Other create shapefiles: `clip_shapefile()`, `raster_bathymetry()`

Index

∗ **basemap functions**
basemap, 2
qmap, 10
shapefile_list, 14
transform_coord, 16
∗ **create shapefiles**
raster_bathymetry, 12
vector_bathymetry, 18
∗ **customize shapefiles**
reorder_layers, 14
theme_map, 15
∗ **shapefiles**
shapefile_list, 14

aggregate, *13*
auto_limits, *14*, *16*

basemap, 2, *11, 12*, *14, 15*, *17*

clip_shapefile, *13*, *18*
CRS, *13*, *18*
cut, *13*

define_shapefiles, *4*
dist2land, 7
drop_crumbs, *18*

fill_holes, *18*
FS, *5*

ggplot, *5, 6*, *11*
ggplot2, *4, 5*
ggspatial, *4*
ggtheme, *4*, *11*, *16*
guess the correct columns, *3*
guess the longitude and/or latitude columns, *8*, *16*
guess_coordinate_columns, *10*

list containing shapefile information, *3*

LS, *4, 5*, *11*

proj4string, *8*, *17*

qmap, *6*, 10, *15*, *17*
qplot, *10*

raster, *13*
raster_bathymetry, 12, *18*
reorder_layers, 14, *16*

shapefile_list, *3–6*, *8*, *12*, 14, *17*
smooth_ksmooth, *18*
Spatial, *17*
SpatialPolygons, *3*, *8*, *13*
SpatialPolygonsDataFrame, *5*, *15*, *18*

theme, *3*, *11*
theme_map, *14*, 15
transform_coord, *6*, *12*, *15*, 16

vector_bathymetry, *12, 13*, 18

writeRaster, *13*