Package ‘geotopbricks’

March 21, 2019

Maintainer  Emanuele Cordano <emanuele.cordano@gmail.com>
License GPL (>= 2)
Title  An R Plug-in for the Distributed Hydrological Model GEOtop
Type  Package
Author  Emanuele Cordano, Daniele Andreis, Fabio Zottele
Description  It analyzes raster maps and other information as input/output files from the Hydrological Distributed Model GEOtop. It contains functions and methods to import maps and other keywords from geotop.inpts file. Some examples with simulation cases of GEOtop 2.x/3.x are presented in the package. Any information about the GEOtop Distributed Hydrological Model source code is available on www.geotop.org. Technical details about the model are available in Endrizzi et al, 2014 (<http://www.geosci-model-dev.net/7/2831/2014/gmd-7-2831-2014.html>).
Version 1.5.3
Date  2019-03-15
Repository CRAN
Depends R (>= 2.10),methods,raster,stringr,zoo
Imports rgdal
Suggests soilwater
RoxygenNote  6.1.1
NeedsCompilation no
Date/Publication 2019-03-21 15:00:03 UTC

R topics documented:

  argsParser ................................................................. 2
  bondone ................................................................. 3
  brick ................................................................. 4
  brick.decimal.formatter ............................................. 5
  brickFromOutputSoil3DTensor ....................................... 6
Description

This command parses ...DESCRIPTION TO DO !!!

Usage

argsParser(option, args, sep = " ", novalue_response = NULL)
### Arguments

- **option**: character strings containing options (or flag) whose values
- **args**: String containing all the arguments of an R script
- **sep**: separator character. Default is " ". If it is of length 2, the first is separator among different options, the second is between option name and its value.
- **novalue_response**: value used in case the option is missing. Default is NULL.

### Examples

```r
args <- "--value 6 --fruit apple"
option <- "--fruit"
value <- argsParser(option=option,args=args)

option2 <- "--jobs"
value2 <- argsParser(option=option2,args=args)
value22 <- argsParser(option=option2,args=args,novalue_response="./")

args_b <- "value=6 , fruit=apple"
option3 <- "value"
value <- argsParser(option=option3,args=args_b,sep=c("","="))
```

---

### Bondone Dataset

**Description**

It contains hourly meteorological data observed at MeteoTrentino T0327 station located at Monte Bondone-Viotte (Trentino, Easter Alps, Italy) from August 2004 to December 2012.

The **zoo** object 'meteo' contains:

- **iprec**: Hourly Precipitation Depth expressed in millimeters
- **airt**: Air Temperature expressed in Celsius Degree
- **rh**: Relative Humidity in PerCent
- **windir**: Wind Direction expressed in Degrees North Clockwise
- **winsp**: Wind Direction expressed in meters per second
- **swglob**: Short-Wave Radiation expressed in Watts per square meters

The corresponding time axis vector for each observation can be printed by typing `index(meteo)`.
Usage

data(bondone)

Format

Data frame, 'zoo' object

Details

This data set stores all meteorological information useful for a GEOtop www.geotop.org simulation. The user can easily use the package with his/her own data after replacing the values of such variables.

Source

Original data are provided by Provincia Autonoma di Trento (http://www.meteotrentino.it/). This dataset is intended for research purposes only, being distributed in the hope that it will be useful, but WITHOUT ANY WARRANTY.

Description

Added implementation for 'brick' S4 method

@title brick

Usage

## S4 method for signature 'zoo'
brick(x, layer = 1, timerange = NULL, time = NULL,
   rows = 1:nrow(x), crs = NULL, use.read.raster.from.url = TRUE)

## S4 method for signature 'GeotopRasterBrick'
brick(x)

Arguments

x a 'zoo' object returned by function pointer.to.maps.xyz.time or pointer.to.maps.xy.time or a GeotopRasterBrick-class object

layer layer at which raster maps are imported. If is NULL, maps are no-layer distributed and zoo must be returned by pointer.to.maps.xy.time

timerange two-elements vector containing the time range at which geotop maps are imported

time vector of time instants at which geotop maps are imported
rows    rows of zoo correspondig to the geotop maps that are imported. By default all rows of zoo are considered. It is calculated by time or timerange if they are not set as NULL.
crs    coordinate system see RasterBrick-class
use.read.raster.from.url logical value. Default is TRUE. If TRUE the RasterLayer are read with read.raster.from.url, istead of raster (otherwise). It is recomended in case the files whose paths are contained in x are remote and are 'http' addresses. In this cases the stand-alone method raster(x) does not always work and use.read.raster.from.url is necessary.

Value

a RasterBrick-class containing the geotop maps indicated by x, which is already in a GeotopRasterBrick-class object or a 'zoo' object returned by function pointer.to.maps.xyz.time or pointer.to.maps.xy.time.

See Also

getvalues.brick.at.depth,vertical.aggregate.brick.within.depth

Examples

# TON TOSS
# See the examples in the functions listed in the 'SeeAlso' section

brick.decimal.formatter

Imports a brick of raster ascii maps into a 'brick' object

Description

Imports a brick of raster ascii maps into a 'brick' object

Usage

brick.decimal.formatter(file = NULL, file_prefix, formatter = "%04d",
file_extension = "_.asc", nLayers = 10,
use.read.raster.from.url = FALSE, crs = NULL,
start.from.zero = FALSE)

Arguments

file    filename of the 'brick' files containing the decimal formatter. It is NULL by default, otherwise it replaces file_suffix, formatter and file_extension.
file_prefix character string suffix name of the 'brick' files.
formatter string value. Default is "%04d". 
**brickFromOutputSoil3DTensor**

```r
file_extension strinf value. Default is ".asc"
nlayers number of layers
use.read.raster.from.url
  logical value. Default is FALSE. (this is recommended in this function). If TRUE
  the RasterLayer are read with read.raster.from.url, instead of raster (other-
  wise). It is recomended in case the files whose paths are contained in x are re-
  mote and are "http" addresses. In this cases the stand-alone method raster(x)
  does not always work and use.read.raster.from.url is necessary.
crs coordinate system see RasterBrick-class.brick, Default is NULL.
start.from.zero
  logical value. Default is FALSE. If TRUE the formatter starts from 0000, otherwise
  it starts from 0001.
```

**Value**

the output is returned as a RasterBrick-class object

**Examples**

```r
library(geotopbricks)
library(raster)
file <- system.file("doc/examples/snowthickness",package="geotopbricks")
file <- paste(file,"SnowThickness0000L%04d.asc",sep="/")
# n_layers=15
nlayers <- 6 ## Only 6 layers are read to minimize the elapsed time of the example!!
b <- brick.decimal.formatter(file=file,nlayers=nlayers)
nlayers(b)
names(b)
```

**Description**

Extracts a brick or a raster layer from a output 3D Tensor or 2D map respectively

**Usage**

```r
brickFromOutputSoil3DTensor(x, when, layers = "SoilLayerThicknesses",
  one.layer = FALSE, suffix = "L%04dN%04d.asc",
  time_formatter = "N%04d", suffix_one.layer = "N%04d.asc",
  wpath = NULL, tz = "A", start_date_key = "InitDateDDMMYYYYhhmm",
  end_date_key = "EndDateDDMMYYYYhhmm", timestep = "OutputSoilMaps",
  use.read.raster.from.url = FALSE, crs = NULL,
  projfile = "geotop.proj", start.from.zero = FALSE,
  secondary.suffix = NULL, only.map.filename = FALSE, ...)
rasterFromOutput2DMap(x, when, ...)
```
Arguments

- **x**: string. GEOtop keyword related to the 3D or 2D variable to be imported in R. When `POSIXct-class` for date and time on which the variable x is requested.
- **layers**: number of soil layer or geotop keyword for soil layer (e.g. `SoilLayerThicknesses` or `SoilFile`). Default is `SoilLayerThicknesses`.
- **one.layer**: logical value. If TRUE a `RasterLayer-class` object is imported, otherwise a `RasterBrick-class` object is returned. Default for `brickFromOutputSoil3DTensor` is `FALSE`.
- **suffix**: character string containing the decimal formatter used by GEOtop in the output file names. Default is "L%04dN%04.asc". A simple user is recommended not to modify the value of this argument and use the default value.
- **time_formatter**, **suffix_one.layer**: character string (suffix_one.layer is used for 2Dxy map) containing the decimal formatter used by GEOtop in the output file names to indicate time instant. Default is "N%04.asc". A simple user is recommended not to modify the value of this argument and use the default value.
- **wpath**, **tz**, **use.read.raster.from.url**
  - see `get.geotop.inpts.keyword.value`
- **start_date_key**, **end_date_key**: initial and final dates and times of the GEOtop simulation or alternatively the respective keywords of *.inpts file (Default)
- **timestep**: time step expressed in seconds every which the raster file has been created. It can be a string corresponding to the geotop keyword in the inpts file. Default value is "OutputSoilMaps".
- **crs**, **start.from.zero**: see `brick.decimal.formatter`. If crs is not `NULL` (Default), projfile is ignored.
- **projfile**: name of the *.proj file containing CRS information. See `get.geotop.inpts.keyword.value`. Default is "geotop.proj". If is `NULL` or `NA` or this file does not exist, it is not searched and read. In case `use.read.raster.from.url` is `TRUE` and no `NULL` or `NA` values are assigned, the *.proj file is searched.
- **secondary.suffix**: String secondary suffix which can be added at the end of the Map file name (optional). Default is `NULL` and no secondary suffix is added.
- **only.map.filename**: logical value. If it is `TRUE`, only map file names are returned and maps are not imported. Default is `FALSE`.
- **...**: additional arguments for `get.geotop.inpts.keyword.value` or `brickFromOutputSoil3DTensor`

Details

These functions `brickFromOutputSoil3DTensor` and `rasterFromOutput2DMap` return 3D or 2D `Raster-class` objects respectively. `rasterFromOutput2DMap` is a wrapper function of `brickFromOutputSoil3DTensor` with the option `one.layer==TRUE`. The functions work with the following output keywords:

- "SoilTempTensorFile"
"SoilAveragedTempTensorFile",
"SoilLiqContentTensorFile",
"SoilAveragedLiqContentTensorFile",
"SoilIceContentTensorFile",
"SoilAveragedIceContentTensorFile",
"SoilLiqWaterPressTensorFile",
"SoilTotWaterPressTensorFile" for \texttt{brickFromOutputSoil3DTensor};
"FirstSoilLayerTempMapFile",
"FirstSoilLayerAveragedTempMapFile",
"FirstSoilLayerLiqContentMapFile",
"FirstSoilLayerIceContentMapFile",
"LandSurfaceWaterDepthMapFile",
"ChannelSurfaceWaterDepthMapFile",
"NetRadiationMapFile",
"InLongwaveRadiationMapFile",
"NetLongwaveRadiationMapFile",
"NetShortwaveRadiationMapFile",
"InShortwaveRadiationMapFile",
"DirectInShortwaveRadiationMapFile",
"ShadowFractionTimeMapFile",
"SurfaceHeatFluxMapFile",
"SurfaceSensibleHeatFluxMapFile",
"SurfaceLatentHeatFluxMapFile",
"SurfaceTempMapFile",
"PrecipitationMapFile",
"CanopyInterceptedWaterMapFile",
"SnowDepthMapFile",
"GlacierDepthMapFile",
"SnowMeltedMapFile",
"SnowSublMapFile",
"GlacierMeltedMapFile",
"GlacierSublimatedMapFile",
"AirTempMapFile",
"WindSpeedMapFile",
"WindDirMapFile",
"RelHumMapFile",
"SWEMapFile",
"GlacierWaterEqMapFile"
"SnowDurationMapFile",
"ThawedSoilDepthMapFile",
"ThawedSoilDepthFromAboveMapFile",
"WaterTableDepthMapFile",
"WaterTableDepthFromAboveMapFile",
"NetPrecipitationMapFile",
"EvapotranspirationFromSoilMapFile" for \texttt{rasterFromOutput2DMap}.

\textbf{Author(s)}

Emanuele Cordano

\textbf{See Also}

\texttt{get.geotop.inpts.keyword.value.brick.decimal.formatter}

\textbf{Examples}

library(geotopbricks)

# The data containing in the link are only for educational use
wpath <- "https://www.rendena100.eu/public/geotopbricks/simulations/idsroclim_test1"
x <- "SoilliqContentTensorFile"
tz <- "Etc/GMT-1"
when <- as.POSIXct("2002-03-22", tz=tz)

# Not Run because it elapses too long time!!!
# Please Uncomment the following lines to run by yourself!!!
b <- brickFromOutputSoil3DTensor(x, when=when, wpath=wpath, tz=tz, use.read.raster.from.url=TRUE)

# a 2D map:
x_e <- "SnowDepthMapFile"
# Not Run: uncomment the following line

m <- rasterFromOutput2DMap(x_e, when=when, wpath=wpath, timestep="OutputSnowMaps",
   tz=tz, use.read.raster.from.url=TRUE)

## NOTE: set use.read.raster.from.url=FALSE (default)
# if the "wpath" directory is in the local file system.
# Not Run: uncomment the following line
plot(m)

## End(Not run)
**color.bar**

*Graphic Representation of a Color bar, function written by John Colby*

**Description**

Graphic Representation of a Color bar, function written by John Colby

**Usage**

```
color.bar(lut, min, max = -min, nticks = 11, ticks = seq(min, max, len = nticks), title = "", width = 1.75, height = 5, ncolmax = 100, digits = 4, pdf = NULL)
```

**Arguments**

- `width`, `height` width and height of the device
- `ncolmax` maximum number of colors. Default is 100.
- `digits` specified number of significant digits
- `pdf` character value for pdf output file. Default is NULL and no pdf file is created.

**Note**

Please visit the URL for major details and give your feedback if possible.

**Author(s)**


**References**


**Examples**

```
color.bar(colorRampPalette(c("light green", "yellow", "orange", "red"))(100), -1)
```
color.bar.raster

---

**color.bar.raster**

Graphic Representation of a Color legend of a Raster or GeotopbrickRaster object as a Color bar, inspired by the function written by John Colby

---

**Description**

Graphic Representation of a Color legend of a Raster or GeotopbrickRaster object as a Color bar, inspired by the function written by John Colby

**Usage**

color.bar.raster(x, col, min = NA, max = NA, ...)

**Arguments**

- **x**: a Raster or GeotopRasterBrick object
- **col**: the color palette used
- **max**, **min**: maximum and minimum value (used if you need to crop the scale legend within a cartein interval)
- **...**: arguments to be passed to `color.bar`

**See Also**

color.bar.setMinMax

color.bar.setMinMax

**Examples**

```r
library(geotopbricks)

## Simulation working path

file <- system.file("rendena100/SnowDepthMapFile-2014-MA-mean-winter-2013-2014.asc", package="geotopbricks")
snow <- raster(file)

min <- 0 # snow depth expressed in millimeters
max <- 2500 # snow depth expressed in millimeters

colors <- terrain.colors(1000)

color.bar.raster(x=snow,col=colors,digits=2)
color.bar.raster(x=snow,col=colors,min=min,max=max,digits=2)
```
create.geotop.inpts.keyword

Creates an 'geotop.inpts' files the keyword and their values of a date.frame like the one returned by declared.geotop.inpts.keywords

Description

Creates an 'geotop.inpts' files the keyword and their values of a date.frame like the one returned by declared.geotop.inpts.keywords

Usage

create.geotop.inpts.keyword(df, file = "geotop.inpts.copy",
file = NULL, comment.lines = "default", header = "default", ...)

Arguments

df  
data frame returnend by declared.geotop.inpts.keywords
file  
connection or file name where to write 'df'
wpapth  
complere path to file (optional). Default is NULL.
comment.lines  
string or vector of strings to add as comments for each keyword. If it is NULL the comment lines are omitted.
header  
string or vector of strings to add as a header. If it is NULL the header is omitted.
...  
further arguments for writeLines

Details

In case comment.lines and header are set equal to "default", they are suitably modified within the function code. See the example output.

See Also

writeLines.declared.geotop.inpts.keywords

Examples

library(geotopbricks)
# Not run:
#Simulation working path
wpapth <- 'https://www.rendena100.eu/public/geotopbricks/simulations/panola13_run2xC_test3'
df <- declared.geotop.inpts.keywords(wpath=wpapth)
create.geotop.inpts.keyword(df=df)

# End(Not run)
create.geotop.meteo.files

Creates geotop meteo files from (a list of) ‘zoo’ objects

Description

Creates geotop meteo files from (a list of) ‘zoo’ objects

Usage

```r
create.geotop.meteo.files(x, format = "%d/%m/%Y %H:%M",
file_prefix = "meteo", file_extension = ".txt",
formatter = "%04d", na = "-9999", col.names = TRUE,
row.names = FALSE, date_field = "Date", sep = ",", level = NULL,
quote = FALSE, ...
)
```

Arguments

- `x`: ‘zoo’ object or a list of ‘zoo’ object representing the meteorological station
- `format`: string format representing the date, see `as.POSIXlt`. Default is "%d/%m/%Y %H:%M" (which is the same format used in geotop.inpts keyword InitDateDDMMYYYYhhmm)
- `file_prefix`: string containing file prefix (full path). It correspos to the value of in geotop.inpts meteofile keyword MeteoFile
- `file_extension`: string containing the extensions of final files. Default is c(".txt")
- `formatter`: string value. It is the decimal formatter contained in the file name and used in case the tabular data are referred at several points. Default is "%04d". See `sprintf`
- `na`: NA value indicator. Default is "-9999". See `write.table`
- `col.names`: logical parameter. Default is TRUE. See `write.table`
- `row.names`: logical parameter. Default is FALSE. See `write.table`
- `date_field`: string value. Default is "Date", otherwise defined by the value of HeaderDateDDMMYYYYhhmmMeteo geotop keyword.
- `sep`: string value. Default is ",,". See `write.table`
- `level`: integer argument. See `get.geotop.inpts.keyword.value` for major details. Default is NULL and is ignored.
- `quote`: logical parameter. Default is TRUE. See `write.table`
- `...`: further arguments for `write.table`

See Also

- `write.table`
- `get.geotop.inpts.keyword.value`
declared.geotop.inpts.keywords

Examples

```r
library(geotopbricks)
data(bondone)
## Not Run - Uncomment te following line to run the example
## create.geotop.meteo.files(x=meteo)
```

Declared.geotop.inpts.keywords

Collects all keywords contained in the 'getop.inpts' configuration files and their values in a data frame object.

Description

Collects all keywords contained in the 'getop.inpts' configuration files and their values in a data frame object.

Usage

```r
declared.geotop.inpts.keywords(wpath, inpts.file = "geotop.inpts",
comment = "!", exceptions = "Date", warn = FALSE,
no.comment = c("!>", ">>"), ...)
```

Arguments

- **wpath**: working directory containing GEOtop files
- **inpts.file**: name of the GEOtop configuration file. Default is "geotop.inpts"
- **comment**: comment indicator character. Default is "!"
- **exceptions**: string vector. If keywords contain an element of this vector, the blank spaces in Value " " will not be removed.
- **warn**: logical argument of `readLines`. Default is FALSE.
- **no.comment**: string indicators read as comment ones by GEOtop but they do not indicate comments by "geotopbricks" package.
- ... further arguments of `readLines`

Value

A data frame with two columns: Keyword and Value

See Also

`get.geotop.inpts.keyword.value`
Description

getopbrick method bla bla bla

Usage

getopbrick(x = NULL, ...)

## Default S3 method:
getopbrick(x, ...)

## S3 method for class 'zoo'
getopbrick(x, layer = NULL, time = NULL, crs = NULL,
          timerange = NULL, ...)

## S3 method for class 'RasterLayer'
getopbrick(x, layer = NULL, time = NULL,
          ascpath = zoo(NULL), ...)

## S3 method for class 'RasterBrick'
getopbrick(x, layer = NULL, time = NULL,
          ascpath = zoo(NULL), ...)

## S3 method for class 'GeotopRasterBrick'
getopbrick(x, layer = NULL, time = NULL,
          crs = NULL, timerange = NULL, ascpath = NULL, ...)

Arguments

- **x**: a 'zoo' object returned by function `pointer.to.maps.xyz.time` or `pointer.to.maps.xy.time` or a `GeotopRasterBrick-class` object
- **layer**: layer at which raster maps are imported. If is NULL, maps are no-layer distributed and zoo must be returnend by `pointer.to.maps.xy.time`
- **time**: vector of time instants at which geotop maps are imported
- **crs**: coordinate system see `RasterBrick-class`
- **timerange**: two-elements vector containing the time range at which geotop maps are imported
- **ascpath**: NULL object or a "zoo" S3 object containing the names of ascii maps provided by GEOtop
Value

a `GeotopRasterBrick-class`

---

`GeotopRasterBrick-class`

`GeotopRasterBrick-class`

Description

A `GeotopRasterBrick`: an object to manage raster maps provided by GEOtop!!

Details

- `ascpath`: A "zoo" S3 object containing the names of ascii maps provided by GEOtop
- `index`: A "POSIXt" S3 object containing time or dates on which raster layers of `brick` are referred
- `layer`: character. Name of the vertical layer at which raster map are referred
- `brick`: A "RasterBrick-class" S4 object containing the Raster-Layer maps imported from GEOtop output files

#' @note A `GeotopRasterBrick` object can be created by `new("GeotopRasterBrick", ...)`

Author(s)

Emanuele Cordano

See Also

`Raster-class`

Examples

`showClass("GeotopRasterBrick")`
get.geotop.inpts.keyword.value

Importing a GEOTop Keyword and its Value into R

Description

It returns the values of a keyword of "geotop.inpts" file or data frame with the suitable format.

Usage

```r
get.geotop.inpts.keyword.value(keyword, inpts.frame = NULL,
   vector_sep = NULL, col_sep = " ", numeric = FALSE,
   format = "%d/%m/%Y %H:%M", date = FALSE, tz = "Etc/GMT-1",
   raster = FALSE, file_extension = ".asc", add_wpath = FALSE,
   wpath = NULL, use.read.raster.from.url = TRUE, data.frame = FALSE,
   formatter = "%O", level = 1, date_field = "Date",
   start_date = NULL, end_date = NULL, ContinuousRecovery = 0,
   ContinuousRecoveryFormatter = ",cereco%O", zlayer.formatter = NULL,
   z_unit = c("centimeters", "millimeters"),
   geotop_z_unit = "millimeters", add_suffix_dir = NULL, MAXNROW = 4,
   header.only = FALSE, ...)
```

Arguments

<table>
<thead>
<tr>
<th>Keyword</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>keyword</td>
<td>keyword name</td>
</tr>
<tr>
<td>inpts.frame</td>
<td>data frame returned by <code>declared.geotop.inpts.keywords</code> or NULL. Default is NULL.</td>
</tr>
<tr>
<td>vector_sep</td>
<td>character value for the separator character if Keyword Value must be returned as a vector, otherwise it is NULL. Default is NULL, but if numeric or date are FALSE, vector_sep is set &quot;,&quot; by default.</td>
</tr>
<tr>
<td>col_sep</td>
<td>character value for the separator character of columns. It is used if Keyword Value is returned as a data frame or zoo object or list of these objects. Default is NULL, but is set &quot;,&quot;.</td>
</tr>
<tr>
<td>numeric</td>
<td>logical value. If TRUE the Value has numeric type, otherwise it is a string or string vector. Default is FALSE.</td>
</tr>
<tr>
<td>format</td>
<td>string format representing the date, see <code>as.POSIXlt</code>, used if date is TRUE. Default is &quot;%d/%m/%Y %H:%M&quot; (which is the format used in geotop.inpts keyword InitDateDDMMYYYYhhmm)</td>
</tr>
<tr>
<td>date</td>
<td>logical value. If TRUE the Value is retured as <code>POSIXlt</code> date, otherwise it is a string or string vector. Default is FALSE.</td>
</tr>
<tr>
<td>tz</td>
<td>format string representing the time zone, see <code>as.POSIXlt</code>, used if date is TRUE. Default is &quot;Etc/GMT-1&quot; (until the previous version it was &quot;A&quot;) which means UTC +1.</td>
</tr>
</tbody>
</table>
raster logical value. Default is FALSE. If TRUE function returns directly the raster map as \texttt{Raster-class} object built with \texttt{raster} method.

file_extension Extension to be added to the keyword if keyword is a file name. Default is ".asc".

add_wpath logical value. Default is FALSE. If TRUE, the wpath string is attached to the keyword string value. It is automatically set TRUE if raster is TRUE.

wpath working directory containing GEOtop files (included the inpts file). It is mandatory if raster is TRUE. See \texttt{declared.geotop.inpts.keywords}.

use.read.raster.from.url logical value. Default is TRUE. If TRUE the RasterLayer are read with \texttt{read.raster.from.url}, instead of \texttt{raster} (otherwise). It is recommended in case the files whose paths are contained in x are remote and are 'http' addresses. In this cases the stand-alone method \texttt{raster(x)} does not always work and \texttt{use.read.raster.from.url} is necessary.

data.frame logical value. It is an option for tabular data. If TRUE function returns directly a data frame or a list of data frames as \texttt{data.frame} or \texttt{zoo} objects imported from the keyword-related files using \texttt{read.table} function. In this case the argument wpath (see \texttt{declared.geotop.inpts.keywords}) is mandatory. Default is FALSE.

formatter string value. It is the decimal formatter contained in the file name and used in case the tabular data are referred at several points. Default is "%04d". It is used in case \texttt{data.frame} is TRUE.

level integer values. Numbers indicating all the identification numbers of the files containing the requested data frames. Default is 1, corresponding to the decimal formatter "0001". See examples.

date_field string value. Default is "Date", otherwise defined by the value of \texttt{HeaderDateDDMMYYYYhhmmMeteo} geotop keyword. It is used only if the argument \texttt{data.frame} is TRUE. If it is NULL or NA the function returns a list of generic \texttt{data.frame} object(s), otherwise \texttt{link(zoo)} object(s). See the arguments \texttt{tz} and \texttt{format} for Date formatting.

isNA numeric value indicating NA in geotop ascii files. Default is -9999.00

matlab.syntax logical value. Default is FALSE. If TRUE a vector is written in a string according to *.m file syntax. Warning: this syntax is not read by GEOtop.

projfile filename of the GEOtop projection file. Default is \texttt{geotop.proj}.

start_date, end_date null objects or dates in POSIXlt format between which the variables are returned. It is enabled in case that \texttt{date_field} is not NULL or NA and \texttt{data.frame} is TRUE. Default is NULL.

ContinuousRecovery integer value. Default is 0. It is used for tabular output data and is the number of times GEOtop simulation broke during its running and was re-launched with 'Continuous Recovery' option.

ContinuousRecoveryFormatter character string. Default is '_crec%04d'. It is used only for tabular output data and if ContinuousRecovery is equal or greater than 1.
get.geotop.inpts.keyword.value

zlayer.formatter
decimal formatter. It is used if data.frame==TRUE and the columns refers to different soil depths. Default is NULL.

z_unit
z coordinate measurement unit. GEOtop values expressed in millimeters which are converted to centimeters by default. Default is c("centimeters","millimeters"). Otherwise can be the ratio between the unit and one meter. It is used if zlayer.formatter=="z%04d" or similar.

geotop_z_unit
z coordinate measurement unit used by GEOtop. Default is millimeters. It is used if zlayer.formatter=="z%04d" or similar.

add_suffix_dir
character string. Add a suffix at the directory reported in the keyword value

MAXNROW
maximum number accepted for data.frame output. Default is 4. It is used in case of data.frame==TRUE. In case the number of records in the function output is less than MAXNROW, function returns neither data.frame nor zo objects but only the keyword value.

header_only
logical value. Default is FALSE. If it is TRUE and data.frame==TRUE, only file header with variable names is returned by the function.

... further arguments of declared.geotop.inpts.keywords

Value
the keyword value

Note
If inpts.frame is NULL, inpts.frame will be obtained by calling the function declared.geotop.inpts.keywords with ... arguments.

Examples

library(geotopbricks)

#Simulation working path
## Not run:
wp <- 'https://www.rendena100.eu/public/geotopbricks/simulations/panola13_run2xC_test3'

prefix <- get.geotop.inpts.keyword.value("SoilLiqWaterPressTensorFile", wpath=wp)

slope <- get.geotop.inpts.keyword.value("SlopeMapFile", raster=TRUE, wpath=wp)
bedrock_depth <- get.geotop.inpts.keyword.value("BedrockDepthMapFile", raster=TRUE, wpath=wp)

layers <- get.geotop.inpts.keyword.value("SoilLayerThicknesses", numeric=TRUE, wpath=wp)
names(layers) <- paste("L",1:length(layers),sep="")

##### set van genuchten parameters to estimate water volume
theta_sat <- get.geotop.inpts.keyword.value("ThetaSat", numeric=TRUE, wpath=wp)
theta_res <- get.geotop.inpts.keyword.value("ThetaRes", numeric=TRUE, wpath=wp)
alphaVG <- get.geotop.inpts.keyword.value("AlphaVanGenuchten", 

get.geotop.inpts.keyword.value

numeric=TRUE, wpath=wpath) # expressed in mm^-1

nVG <- get.geotop.inpts.keyword.value("NVanGenuchten", numeric=TRUE, wpath=wpath)

###### end set van genuchten parameters to estimate water volume

###### set meteo data

start <- get.geotop.inpts.keyword.value("InitDateDDMMYYYYhhmm", date=TRUE, wpath=wpath, tz="A")
end <- get.geotop.inpts.keyword.value("EndDateDDMMYYYYhhmm", date=TRUE, wpath=wpath, tz="A")
nmeteo <- get.geotop.inpts.keyword.value("NumberOfMeteoStations", numeric=TRUE, wpath=wpath)
level <- 1:nmeteo

# Uncomment the following lines to run the R code:

## set meteo data

## End(Not run)

## Not run:
meteo <- get.geotop.inpts.keyword.value("MeteoFile", wpath=wpath, data.frame=TRUE, level=level, start_date=start, end_date=end)

## End(Not run)

##### end set meteo data

##### IMPORTING AN OUTPUT SOIL MOISTURE PROFILE:


## Not run:
SMC <- get.geotop.inpts.keyword.value("SoilLiqContentProfileFile", wpath=wpath, data.frame=TRUE, date_field="Date12.DDMMYYYYhhmm.", formatter="%04d")

SMCz <- get.geotop.inpts.keyword.value("SoilLiqContentProfileFile", wpath=wpath, data.frame=TRUE, date_field="Date12.DDMMYYYYhhmm.", formatter="%04d", zlayer.formatter="z%04d")

## End(Not run)
get.geotop.recovery.state

This function saves all spatially distributed information contained in the recovery folder into a comprehensive list object.

Description

This function saves all spatially distributed information contained in the recovery folder into a comprehensive list object.

Usage

```r
get.geotop.recovery.state(recFolder, xx = "0000", formatter = "L%04d",
extension = ".asc", nsoillayers = 10, layersFromDir = FALSE, ...)
```

Arguments

- `recFolder` directory when recovery maps are set. In GEOtop it is ...
- `xx` character String. Default is "0000"
- `formatter` string character for the the decimal formatter to be used. Default is "L%04d".
- `extension` file extension used for asciii recovery map files. It must contains "." as the first character. Default is ".asc".
- `nsoillayers` number of soil layers used in the GEOtop simulation.
- `layersFromDir` logical value. If is TRUE the number of soil/snow (vertical) layers used in the GEOtop simulation is automatically calculated and cannot be assigned through nsoillayers.
- `...` further arguments

Value

- a list object containing all recovery raster maps.

Note

This function has been used with the built 1.225-9 of GEOtop.

Author(s)

Emanuele Cordano

See Also

- `brick.decimal.formatter`,
- `raster.set.geotop.recovery.state`,
- `write.vectorized.geotop.recovery.read`,
**getProjection**

**Examples**

```r
library(geotopbricks)
exampe_Rscript <- system.file('template/example.geotop.recovery.state.R', package = "geotopbricks")
exampe_Rscript

# Not Run because it elapses too long time!!!
# Please Uncomment the following line to run by yourself!!!
# source(exampel_Rscript)
```

---

**getProjection**  
It reads the CRS metadata utilized in a GEOtop Simulation

**Description**

It reads the CRS metadata utilized in a GEOtop Simulation

**Usage**

```r
getProjection(x, cond = TRUE, ...)
```

**Arguments**

- `x` name and full path of the file containing CRS information
- `cond` logical value. If FALSE the function returns NA. Default is TRUE.
- `...` further arguments

**Value**

A string corresponding the projection and CRS if the argument cond is TRUE.

**Examples**

```r
library(geotopbricks)
wpah <- "https://www.rendena100.eu/public/geotopbricks/simulations/idroclim_test1"
## Not run:

x <- paste(wpah,"geotop.proj",sep="/")

crs <- getProjection(x)

## End(Not run)
```
getvalues.brick.at.depth

Interpolates the values of a 'brick' at a certain depth and returns the map of brick values at the "depth" level

Description

Interpolates the values of a 'brick' at a certain depth and returns the map of brick values at the "depth" level

Usage

getvalues.brick.at.depth(x, depth, layers, i0 = NULL, verify = FALSE, ...)

Arguments

x a 'RasterBrick' or a three-dimensional array
depth depth map, generally a 'RasterLayer' object
layers vector of layer thickness
i0 a 'Raster' containing the number of soil layer just over the bedrock. Default is NULL and is then calculated.
verify logical. Default is FALSE. If it is TRUE, it verifies that function is working correctly.
... further argument

Value

a list of 'Raster' maps:
i0 a 'Raster' containing the number of soil layer just over the bedrock
val_z0 a 'Raster' containing the values of x at the i0-th layer
val_z1 a 'Raster' containing the values of x at the (i0+1)-th layer
z0 a 'Raster' containing the depth of the center of the i0-th layer
z1 a 'Raster' containing the depth of the center of the (i0+1)-th layer

Note

x and depth or i0 must cover the same spatial region.

See Also

codevertical.aggregate.brick.within.depth
max_value

Examples

library(geotopbricks)
# The examples is the following R script conteined in a 'inst' directory of the package source
f <- system.file("doc/examples/example.getvalues.brick.at.depth.R",package="geotopbricks")
# source(f) # Uncomment this line to run the example.
# You can copy the example file using file.copy(from=f,to=....,...) See file.copy documentation

KML               KML

Description

KML method for a GeotopRasterBrick object

Usage

## S4 method for signature 'GeotopRasterBrick'
KML(x, filename,
    crs = as.character("+proj=longlat +ellps=WGS84 +datum=WGS84 +no_defs"),
    ...)

Arguments

   x        the GeotopRasterBrick object
    filename mane of the KML file to produce
    crs      character string containing the LatLon reference system. Default is "+proj=longlat +ellps=WGS84 +datum=WGS84 +no_defs"
               (see http://spatialreference.org/ref/epsg/4326/).
          ...
               further argument for S4 method KML for Raster object.

Note

A coordinate transformation is made with projectRaster.

max_value

Description

  Gets the maximum (scalar) values of a GeotopRasterBrick object

Usage

  max_value(x)
min_value

Arguments
  x a GeotopRasterBrick object
  ... further arguments

Value
  the maximum (scalar) values of a GeotopRasterBrick object

min_value min_value

Description
  Gets the minimum (scalar) values of a GeotopRasterBrick object

Usage
  min_value(x)

Arguments
  x a GeotopRasterBrick object
  ... further arguments

Value
  the minimum (scalar) values of a GeotopRasterBrick object

Ops Ops

Description
  Ops

Usage
  ## S4 method for signature 'GeotopRasterBrick,GeotopRasterBrick'
  Ops(e1, e2)

  ## S4 method for signature 'GeotopRasterBrick,numeric'
  Ops(e1, e2)

  ## S4 method for signature 'numeric,GeotopRasterBrick'
  Ops(e1, e2)
Arguments

e1, e2 the `GeotopRasterBrick` or numeric objects

Details

Ops method for a GeotopRasterBrick object

Note

If e1 or e2 time index is not taken into account.

plot

Description

plot method for a GeotopRasterBrick object

Usage

```r
## S4 method for signature 'GeotopRasterBrick,ANY'
plot(x, y = NULL, ...)
```

Arguments

- `x` the `GeotopRasterBrick` object
- `y` further argument
- `...` further argument for S4 method plot for Raster object.

See Also

KML

Examples

```r
library(geotopbricks)
# The examples is the following R script conteined in a 'inst' directory of the package source
f <- system.file("doc/examples/example.plot.GeotopRasterBrick.R",package="geotopbricks")
# source(f) # Uncomment this line to run the example.
# You can copy the example file using file.copy(from=f,to=.\.\...) See file.copy documentation
```
`pointer.to.maps.xyz.time` function

**Usage**

```r
pointer.to.maps.xyz.time(wpath, map.prefix = "thetaliq",
                        suffix = "L%04d%04d.asc", zoo.index = NULL, ntime, nlayers)
```

**Arguments**

- `wpath` complete working path to *.asc* maps are saved
- `map.prefix` string prefix name map before
- `suffix` z-time or time suffix plus file extention character string. Default for GEOtop application is "L%04d%04d.asc" for xy+z+time maps or "N%04d.asc" for xy+time maps.
- `zoo.index` time or date index. Default is NULL, otherwise function returns a zoo object with zoo.index as index.
- `ntime` number of time instant. If zoo.index is not NULL, it is calculated from zoo.index length.
- `nlayers` number of vertical layers.

**Value**

A data.frame or zoo object containig the paths to maps fpr each time and z layer.

**Author(s)**

Emanuele Cordano

---

`read.ascii.vectorized.brick`

*Read a text file containing values and matedata of a z-layer brick referred to a time instant (e.g. date). The file is formatted like an ascii format like 'geotop.inpts' file.*

**Description**

Read a text file containing values and matedata of a z-layer brick referred to a time instant (e.g. date). The file is formatted like an ascii format like 'geotop.inpts' file.
Usage

```r
read.ascii.vectorized.brick(file = NULL, comment = "!", crs = "", NAflag = -9999, matlab.syntax = FALSE, ...)
```

Arguments

- `file`: file name to write
- `comment`: character. Comment indicator. Default is "!".
- `crs`: Character or object of class CRS. PROJ4 type description of a Coordinate Reference System (map projection) (optional). See `brick` or `raster`.
- `NAflag`: numeric. Default is -9999, see `writeRasterxGEOTop`.
- `matlab.syntax`: logical value. Default is FALSE. If TRUE the file syntax is like the one of a *.m Matlab script file.
- `...`: further arguments inserted as attribute

Value

the `RasterBrick-class` object

See Also

- `write.ascii.vectorized.brick`

Examples

```r
# see the examples of read.ascii.vectorized.brick
```

---

**read.raster.from.url**

It imports a 'RasterLayer' object in Escri-Asci format from a URL 'http://....<FILENAME>.asc'

---

**Description**

It imports a 'RasterLayer' object in Escri-Asci format from a URL 'http://....<FILENAME>.asc'

Usage

```r
read.raster.from.url(x, header_nrow = 6, ...)
```

Arguments

- `x`: the character string containing the URL address
- `...`: additional arguments
Value

a ‘RasterLayer’ object

Note

This function reads a local or remote text files formatted as http://en.wikipedia.org/wiki/Esri_grid and creates a ‘RasterLayer’ object.

See Also

raster.readLines

---

Value

a list object like get.geotop.recovery.state
replace.keyword

See Also

write.vectorized.geotop.recovery

Examples

# see the examples of read.ascii.vectorized.brick

replace.keyword = It replaces some keyword values of geotop.inpts file with the ones of another *.inpts value

Description

It replaces some keyword values of geotop.inpts file with the ones of another *.inpts value

Usage

replace.keyword(x, y = "geotop.inpts", file.output = NULL,
write.file.output = TRUE, wpath = NULL, ...)

Arguments

x filename of the *.inpts with the "new" keyword value
y filename of the *.inpts with the "old" keyword value. Default is "geotop.inpts".
file.output filename where to write the comprehensive new geotop.inpts file. If it is NULL (default), the filename is assigned by y.
write.file.output logical value. If it is TRUE, the output of the function is written in the file file.output.
wpath working path to the GEOtop simulation folder containing the x and y files.
... further arguments

Details

This function replaces some keyword values of y with the ones indicated in y. It is useful to replace the meteo station metadata, for instance, when the meteorological station of a study cases are modified. The function returns the new geotop.inpts file as a vector of character strings. If write.file.output==TRUE, the output is written in an external file, e.g. "geotop.inpts" newly (this option is suggested).

Author(s)

Emanuele Cordano
Examples

```r
library(geotopbricks)
wpath <- system.file('template/meteo_ex', package = "geotopbricks")
x <- "meteo.inpts"
z1 <- replace.keyword(x, wpath = wpath, write.file.output = FALSE)
```

---

**set.geotop.recovery.state**

*This function re-writes the recovery ascii raster maps in a given folder*

Description

This function re-writes the recovery ascii raster maps in a given folder

Usage

```r
set.geotop.recovery.state(rec, newRecFolder, ...)
```

Arguments

- `rec`: a list object returned by `get.geotop.recovery.state`
- `newRecFolder`: directory where to write all recovery raster ascii maps
- `...`: further arguments

Author(s)

Emanuele Cordano

See Also

`get.geotop.recovery.state`, `writeRasterxGE0top`

Examples

```r
# See the examples of the 'get.geotop.recovery.state' function
```
vertical.aggregate.brick.within.depth

Aggregates with a mean or an addition on the vertical profile the values of a 'brick' within a certain depth and returns the vertical aggregated map

Description
Aggregates with a mean or an addition on the vertical profile the values of a 'brick' within a certain depth and returns the vertical aggregated map

Usage
vertical.aggregate.brick.within.depth(x, depth = NULL, layers = NULL, 
i0 = NULL, verify = FALSE, FUN = identity, 
divide.by.depth = FALSE, ...)

Arguments
  x  a 'RasterBrick' or a three-dimensional array
  depth  depth map, generally a 'RasterLayer' object
  layers  vector of layer thickness
  i0  a 'Raster' containing the number of soil layer just over the bedrock. Default is NULL and is then calculated.
  verify  logical. Default is FALSE. If it is TRUE, it verifies that function is working correctly.
  FUN  function used for aggregation. If missing, identity is the default value.
  divide.by.depth  logical. If TRUE the function returns the 'mean' value, otherwise a a cumulate value. Default is FALSE.
  ...  further argument for FUN

Value
  a list of 'Raster' maps:
    i0  a 'Raster' containing the number of soil layer just over the bedrock
    z0  a 'Raster' containing the depth of the center of the i0-th layer
    result  a 'Raster' containing the aggregated map

Note
  x and depth or i0 must cover the same spatial region.
write.ascii.vectorized.brick

See Also

getvalues.brick.at.depth.brick

Examples

library(geotopbricks)
# The examples is the following R script contenued
# in a 'inst' directory of the package source
f <- system.file("doc/examples/example.vertical.aggregate.brick.within.depth.R",
package="geotopbricks")
# source(f) # Uncomment this line to run the example.
# You can copy the example file using file.copy(from=f,to=........) See file.copy documentation

write.ascii.vectorized.brick

Write a z-layer brick referred to a time instant (e.g. date) in an ascii format like 'geotop.inpts' file.

Description

Writes a z-layer brick referred to a time instant (e.g. date) in an ascii format like 'geotop.inpts' file.

Usage

write.ascii.vectorized.brick(b, file = NULL, header = NULL,
overwrite = TRUE, NAflag = -9999, matlab.syntax = FALSE, ...)

Arguments

b
file
header
overwrite
NAflag
matlab.syntax
...
a RasterBrick-class or GeotopRasterBrick-class object
file name to write
character string vector for header text lines. If missing, a default header is written. #Default is c("!
header").
logical. Default is TRUE, see writeRaster.
numeric. Default is -9999, see writeRasterxGEotop.
logical value. Default is FALSE. If TRUE the file syntax is like the one of a *.m Matlab script file.
further arguments inserted as attribute
**Value**

the string vector possibly written in file.

**Note**

Add Quote if necessary. This function is NOT maintained and will be DEPRECATED.

**See Also**

read.ascii.vectorized.brick

**Examples**

```r
## Not Run
## library(geotopbricks)
## library(raster)
## file <- system.file("doc/examples/snowthickness", package="geotopbricks")
## file <- paste(file, "SnowThickness000L%04d.asc", sep="/")
## b <- brick.decimal.formatter(file=file,nlayers=15)
## nlabels(b)
## file <- "snow.txt"
## btext <- write.ascii.vectorized.brick(b,Date="1/1/2009",file="snow.txt")
## The printed object
## str(btext)
## bb <- read.ascii.vectorized.brick(file = file)
## bf <- abs(as.matrix(bb[[1]]-bb[[1]]))<.Machine$double.eps^0.5
```

---

**write.geotop.table**  
*Writes an R object (data.frame or zoo) into a CSV file readable by GEOtop.*

**Description**

Writes an R object (data.frame or zoo) into a CSV file readable by GEOtop.

**Usage**

```r
write.geotop.table(x, file, wpath = NULL, tz = "Etc/GMT-1",
                   date_field = "Date12.DDMMYYYYhhmm.", file_end = "",
                   sep = ",", format = "%d/%m/%Y %H:%M", na = "-9999", ...)
```

**Arguments**

- `x`: R object (data.frame or zoo) to be exported and written.
- `file`: filename
- `wpath`: working path to the GEOtop simulation. If `wpath` is not `NULL`, filename will be put in `wpath`. 
**write.vectorized.geotop.recovery**

It writes a list object returned by `get.geotop.recovery.state` as a string vector or in a text file, following `.inpts` or Matlab-like syntax.

**Description**

It writes a list object returned by `get.geotop.recovery.state` as a string vector or in a text file, following `.inpts` or Matlab-like syntax.

**Usage**

```r
write.vectorized.geotop.recovery(rec, file = NULL, header = NULL,
                                  overwrite = TRUE, NAflag = -9999, matlab.syntax = TRUE, ...)
```

**Arguments**

- `rec`  
a list object returned by `get.geotop.recovery.state`
- `file`  
ascii text file name where to write the string vector
- `header`  
character string vector for header text lines. If missing, a default header is written. Default is c("! header") or the one assigned by `matlab.syntax`.
- `overwrite`  
logical. Default is `TRUE`, see `writeRaster`.
- `NAflag`  
numeric. Default is `-9999`, see `writeRasterxGEotop`.
- `matlab.syntax`  
logical value. Default is `TRUE`. If `TRUE` the file syntax is like the one of a `*.m` Matlab script file.
- `...`  
further arguments inserted as attribute

**Value**

a string vector containing the `rec` variables.
write.vectorized.variable.in.string

Note
Add Quote if necessary

See Also
get.geotop.recovery.state,set.geotop.recovery.state,write.vectorized.variable.in.string

Examples
# See the examples of the 'get.geotop.recovery.state' function

write.vectorized.variable.in.string
Writes one or more variables (scalars, vectors or Rasters) in a string each, following *.inpts or Matlab-like syntax.

Description
Writes one or more variables (scalars, vectors or Rasters) in a string each, following *.inpts or Matlab-like syntax.

Usage
write.vectorized.variable.in.string(l, NAflag = -9999, matlab.syntax = FALSE, ...)

Arguments
l
a codelist object contained the variables (scalars, vectors or Rasters) which will be written in a string each.
NAflag numeric. Default is -9999, see writeRasterxGEotop.
matlab.syntax logical value. Default is FALSE. If TRUE the file syntax is like the one of a *.m Matlab script file.
...
further arguments

Value
the string vector <NAME_VARIABLE>=<VALUES_VARIABLE>.

Note
Add Quote if necessary
writeRasterxGEOtop

See Also

read.ascii.vectorized.brick

Examples

```r
a <- 1:5
l <- list(v=a,a=a)
out <- write.vectorized.variable.in.string(l,matlab.syntax=TRUE)
out
```

Description

This function uses `writeRaster` to create .asc maps which can be read by GEOtop.

Usage

```r
writeRasterxGEOtop(x, filename = NULL, overwrite = TRUE,
NAflag = -9999, use.decimal.formatter = FALSE,
start.from.zero = FALSE, keyword, wpath, suffix.ext = ".asc", ...)
```

Arguments

- **x**: a Raster object, see `writeRaster`. It can be also a `RasterBrick-class` object.
- **filename**: see `writeRaster`. It is a vector of string or one string containing a decimal formatter (see `brick.decimal.formatter`) in case `x` is a `RasterBrick-class` object.
- **overwrite**: logical. Default is `TRUE`, see `writeRaster`.
- **NAflag**: numeric. Default is `-9999`, see `writeRaster`.
- **use.decimal.formatter**: logical value. Default is `FALSE`. If it is `TRUE` or `x` is a `RasterBrick-class` object with `nlayers(x)`! = `length(filename)`, `filename` is considered as one string containing a decimal formatter (e.g. `"%04d"`, see `brick.decimal.formatter`). Otherwise, if `filename` is considered as a vector string.
- **start.from.zero**: logical value. Default is `FALSE`. If `TRUE` the formatter starts from `0000`, otherwise it starts from `0001`.
- **keyword**: geotop keyword to be used to extract the raster file name from `geotop.inpts` file. This is enabled if `filename` is equal to `NULL`.
- **wpath**: simulation folder containing `geotop.inpts` file.
suffix.ext  character string to be added to the keyword value, e.g., possible suffix and extension of the raster file name. Default is ".asc".

... further arguments of `get.geotop.inpts.keyword.value` or `writeRaster`

Note

It makes use of `system` functions. It uses `*.asc` format for raster files. In case the file name `filename` is missing and then NULL, it must be imported by the simulation `geotop.inpts` file.

Examples

```r
library(geotopbricks)

## Simulation working path

file <- system.file("renden100/SnowDepthMapFile-2014-MA-mean-winter-2013-2014.asc", package="geotopbricks")
snow <- raster(file)

c <- RasterBrick(snow, file = file)

dir.create("./temporary")
writeRaster(cGEotop(x = snow, file = snowfile))
```

---

**zoo-class**

*A GeotopRasterBrick: an object to manage raster maps provided by GEOtop!!*

**Description**

A GeotopRasterBrick: an object to manage raster maps provided by GEOtop!!

**Examples**

```r
showClass("zoo")
```
# Index

## Topic classes
- GeotopRasterBrick-class, 16

## Topic dataset
- bondone, 3

## Topic methods
- Ops, 25
  - argsParser, 2
  - as.POSIXlt, 13, 17
  - bondone, 3
  - brick, 4, 6, 28, 29, 33
  - brick, GeotopRasterBrick-method (brick), 4
  - brick, zoo-method (brick), 4
  - brick.decimal.formatter, 5, 7, 9, 21, 37
  - brickFromOutputSoil3DTensor, 6, 7, 8
  - color.bar, 10, 11
  - color.bar.raster, 11
  - create.geotop.inpts.keyword, 12
  - create.geotop.meteo.files, 13
  - data.frame, 18
  - declared.geotop.inpts.keywords, 12, 14, 17–19
  - geotopbrick, 15
  - GeotopRasterBrick, 24–26
  - GeotopRasterBrick (GeotopRasterBrick-class), 16
  - GeotopRasterBrick-class, 16
  - get.geotop.inpts.keyword.value, 7, 9, 13, 14, 17, 35, 38
  - get.geotop.recovery.state, 21, 29, 31, 35, 36
  - getProjection, 22
  - getvalues.brick.at.depth, 5, 23, 33
  - identity, 32

## KML
- GeotopRasterBrick-method (KML), 24
- list, 29
- max_value, 24
- meteo (bondone), 3
- min_value, 25
- Ops, 25
- Ops, GeotopRasterBrick, GeotopRasterBrick-method (Ops), 25
- Ops, GeotopRasterBrick, numeric-method (Ops), 25
- Ops, numeric, GeotopRasterBrick-method (Ops), 25

## plot
- plot, 26
  - plot, GeotopRasterBrick, ANY-method (plot), 26
  - pointer.to.maps.xyz.time, 4, 5, 15
  - pointer.to.maps.xy.time (pointer.to.maps.xyz.time), 27
  - pointer.to.maps.xyz.time, 4, 5, 15, 27
  - POSIXlt, 17
  - projectRaster, 24

## raster
- 5, 6, 18, 21, 28, 29
- rasterFromOutput2DMap, 9
- rasterFromOutput2DMap (brickFromOutputSoil3DTensor), 6
- read.ascii.vectorized.brick, 27, 34, 37
- read.raster.from.url, 5, 6, 18, 28
- read.table, 18
- read.vectorized.geotop.recovery, 21, 29
- readLines, 14, 29
- replace.keyword, 30

## set.geotop.recovery.state
- 21, 31, 36

## setMinMax
- 11
sprintf, 13
system, 38

vertical.aggregate.brick.within.depth,
   5, 23, 32
write.ascii.vectorized.brick, 28, 33
write.geotop.table, 34
write.table, 13, 35
write.vectorized.geotop.recovery, 21,
   29, 30, 35
write.vectorized.variable.in.string,
   36, 36
writeLines, 12
writeRaster, 33, 35, 37, 38
writeRasterxGE0top, 28, 29, 31, 33, 35, 36,
   37

zoo, 3, 18
zoo-class, 38