Package ‘geotopbricks’

August 29, 2016

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 2.0 are presented in the package. Any information about the GEOtop Distributed Hydrological Model source code is available on https://github.com/geotopmodel or https://github.com/se27xx/GEOtop. 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 1.4
Date 2016-07-07
Repository CRAN
Depends R (>= 2.10),methods,raster,stringr,zoo
Imports rgdal
Suggests soilwater
RoxygenNote 5.0.1
NeedsCompilation no
Date/Publication 2016-07-07 17:37:40

R topics documented:

argsParser ................................................................. 2
Parser of an argument string

This command parses ...

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"
value <- argsParser(option=option,args=args_b,sep=c(" ", " = " ))
```

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
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".
- **file_extension**: string value. Default is "asc"
brickFromOutputSoil3DTensor

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, insted of raster (other-
wise). It is recomended in case the files whose paths are contained in x are re-
ome 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

library(geotopbricks)
library(raster)
file <- system.file("doc/examples/snowthickness",package="geotopbricks")
file <- paste(file,"SnowThickness0000L%4d.asc",sep="/"
# nLayers=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

brickFromOutputSoil3DTensor(x, when, layers = "SoilLayerThicknesses",
one.layer = FALSE, suffix = "L%4dN%4d.asc",
time_formatter = "N%4d", suffix_one.layer = "N%4d.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 "%04d\n%.4.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 "%.4.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` or `rasterFromOutput2DMap`

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", 

"SoilLayerThicknesses", 

"SoilFile"
"SoilAveragedTempTensorFile",
"SoilLiqContentTensorFile",
"SoilAveragedLiqContentTensorFile",
"SoilIceContentTensorFile",
"SoilAveragedIceContentTensorFile",
"SoilLiqWaterPressTensorFile",
"SoilTotWaterPressTensorFile" for 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}

\begin{verbatim}
library(geotopbricks)
# The data containing in the link are only for educational use
wpath <- "http://www.rendena100.eu/public/geotopbricks/simulations/idroclim_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 <- \texttt{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 <- \texttt{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{verbatim}
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
This function is taken from http://stackoverflow.com/questions/9314658/colorbar-from-custom-colorrampalette.
Please visit the URL for major details and give your feedback if possible.

Author(s)
John Colby http://stackoverflow.com/users/412342/john-colby

References
http://stackoverflow.com/questions/9314658/colorbar-from-custom-colorrampalette

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

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

---

### Description

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

### Usage

```r
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 certain interval)
- `...`: arguments to be passed to `color.bar`

### See Also

`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' file the keyword and their values of a data.frame like the one returned by declared.geotop.inpts.keywords

Description

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

Usage

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

Arguments

- df: data frame returned by declared.geotop.inpts.keywords
- file: connection or file name where to write 'df'
- wpath: complete 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)

# Simulation working path
wpath <- 'http://www.rendena100.eu/public/geotopbricks/simulations/panola13_run2xC_test3'
df <- declared.geotop.inpts.keywords(wpath=wpath)
create.geotop.inpts.keyword(df=df)
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 corresponds to the value of in geotop.inpts 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

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

dec\nared.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

declared.geotop.inpts.keywords(wpath, inpts.file = "geotop.inpts",
comment = "، exceptions = "Date", warn = FALSE,
no.comment = c("!"!, ">",">>, """", ...)\n
Arguments

wpath working directory containing GEOTop files
inpts.file name of the GEOTop configuration file. Default is "geotop.inpts"
comment string 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

g\n\nget.geotop.inpts.keyword.value
Description

geotopbrick method bla bla bla

Usage

geotopbrick(x = NULL, ...)

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

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

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

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

## S3 method for class 'GeotopRasterBrick'
geotopbrick(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

... further arguments.

layer layer at which raster maps are imported. If is NULL, maps are no-zlayer distributed and zoo must be returned 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`

**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

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 = "%04d", level = 1, date_field = "Date", isNA = -9999, matlab南省yntax = TRUE, projfile = "geotop.proj", start_date = NULL, end_date = NULL, ContinuousRecovery = 0, ContinuousRecoveryFormatter = "._crec%04d", zlayer.Formatter = NULL, z_unit = c("centimeters", "millimeters"), geotop_z_unit = "millimeters", add_suffix_dir = NULL, ...

Arguments

<table>
<thead>
<tr>
<th>Argument</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 declared.geotop.inpts.keywords 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 as.POSIXlt, 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 returned as POSIXlt 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 as.POSIXlt, 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>
<tr>
<td>raster</td>
<td>logical value. Default is FALSE. If TRUE function returns directly the raster map as Raster-class object built with raster method.</td>
</tr>
</tbody>
</table>
get.geotop.inpts.keyword.value

**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 declared.geotop.inpts.keywords.

**use.read.raster.from.url**  logical value. Default is TRUE. If TRUE the RasterLayer are read with read.raster.from.url, instead of 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 raster(x) does not always work and 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 data.frame or zoo objects imported from the keyword-related files using read.table function. In this case the argument wpath (see 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 data.frame is TRUE.

**level**  integer values. Numbers incating all the identandification numbers of the files containing the requested data frames. Default is 1, correspondig to the decimal formatter "0001". See examples.

**date_field**  string value. Default is "Date", otherwise defined by the value of HeaderDateDDMMYYYYhhmmMeteogeo top keyword. It is used only if the argument data.frame is TRUE. If it is NULL or NA the function return a list of generic data.frame object(s), otherwise link(zoo) object(s). See the arguments tz and 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 synstax is not read by GEOtop.

**projfile**  fileneme of the GEOtop projection file. Default is 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 date_field is not NULL or NA and 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.

**zlayer.Formatter**  decimal formatter. It is used if data.frame==TRUE and the columns refers to different soil depths. Default is NULL.
**get.geotop.inpts.keyword.value**

- **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
  further arguments of declared.geotop.inpts.keys

**Value**

the keyword value

**Note**

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

**Examples**

```r
library(geotopbricks)

#Simulation working path
wpath <- 'http://www.rendena100.eu/public/geotopbricks/simulations/panola13_run2xC_test3'
prefix <- get.geotop.inpts.keyword.value("SoilLiqWaterPressTensorFile",wpath=wpath)

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

layers <- get.geotop.inpts.keyword.value("SoilLayerThicknesses",numeric=TRUE,wpath=wpath)
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=wpath)
theta_res <- get.geotop.inpts.keyword.value("ThetaRes",numeric=TRUE,wpath=wpath)
alphaVG <- get.geotop.inpts.keyword.value("AlphaVanGenuchten",
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)
```
get.geotop.recovery.state

level <- 1:nmeteo

# Uncomment the following lines to run the R code:

## set meteo data

## 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:

wpath <- 'http://www.rendena100.eu/public/geotopbricks/simulations/Muntatschini_pnt_1_225_B2_004'

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

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

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 ascii recovery map files. It must contains '.' as the first character. Default is "asc".

nsollayers 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 nsollayers.

... further arguments

Value

a list object containining 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,

eraster.set.geotop.recovery.state,

write.vectorized.geotop.recovery.read.vectorized.geotop.recovery

Examples

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

# Not Run because it elapses too long time!!!
# Please Uncomment the following line to run by yourself!!!
# source(example_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)
wpath <- "http://www.rendena100.eu/public/geotopbricks/simulations/idroclim_test1"
x <- paste(wpath,"geotop.proj",sep="/"")

crs <- getProjection(x)
```

---

**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**

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

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

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.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
```
listFromOutputSoil3DTensor

Description

Extracts a list of files pointing to an output 3D Tensor or 2D map respectively

Usage

```r
KML(x, filename, 
    crs = as.character("+proj=longlat +ellps=WGS84 +datum=WGS84 +no_defs"), ...)
```

Arguments

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

Note

A coordinate transformation is made with `projectRaster`.

Examples

```r
library(geotopbricks)
# The examples is the following R script contented in a 'inst' directory of the package source
f <- system.file("doc/examples/example.KML.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
```
Usage

```r
listFromOutputSoil3DTensor(x, when, layers = "SoilLayerThicknesses",
one.layer = FALSE, suffix = "L%04dN%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, ...)
```

Arguments

- **x**  
  string. GEOtop keyword related to the 3D or 2D variable to be imported in R.
- **when**  
  `POSIXlt-class` for date and time on which the variable `x` is requested.
- **layers**  
  number of soil layer or geo top 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.
- **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 geo top 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.
- **...**  
  additional arguments for `get.geotop.inpts.keyword.value` or `brickFromOutputSoil3DTensor`

Details

This function is experimental and documentation partially exhaustive. 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 rasterFromOutput2DMap.

Author(s)
Emanuele Cordano

See Also
get.geotop.inpts.keyword.value.brick.decimal.formatter.brickFromOutputSoil3DTensor

Examples

tz <- "Etc/GMT+1"
start <- as.POSIXct("2002-03-22", tz=tz)
end <- as.POSIXct("2002-03-25", tz=tz)
day <- 3600*24
when <- seq(from=start, to=end, by=day)
# The data containing in the link are only for educational use
wpath <- "http://www.rendena100.eu/public/geotopbricks/simulations/idroclim_test1"
x <- "SoilLiqContentTensorFile"
when <- as.POSIXct("2002-03-22 UTC", tz="A")

# Not Run because it elapses too long time!!!
# Please Uncomment the following lines to run by yourself!!!

# wpath <- '/Users/ecor/attivita/2013/fem-Idroclima/Trentino_500_dstr_GE0top_1.225.9_002'
# kpsi <- "SoilLiqWaterPressTensorFile" ## soil water pressure head
out <- listFromOutputSoil3DTensor(x, when=when, wpath=wpath, tz=tz, use.read.raster.from.url=FALSE)
Description

Gets the maximum (scalar) values of a `GeotopRasterBrick` object

Usage

```
max_value(x)
```

Arguments

- `x` a `GeotopRasterBrick` object
- `...` further arguments

Value

the maximum (scalar) values of a `GeotopRasterBrick` object

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
### Description

Ops method for a GeotopRasterBrick object

### Usage

```r
## 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

### Note

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

### 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.GeoRaterBrick.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**

-pointer.to.maps.xyz.time-

**Description**

'pointer.to.maps.xyz.time' function

**Usage**

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

**Arguments**

<table>
<thead>
<tr>
<th>Argument</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>wpath</td>
<td>complete working path to *.asc maps are saved</td>
</tr>
<tr>
<td>map.prefix</td>
<td>string prefix name map before</td>
</tr>
<tr>
<td>suffix</td>
<td>z-time or time suffix plus file extension character string. Default for GEOtop app-</td>
</tr>
<tr>
<td></td>
<td>lication is &quot;L%04dN%04d.asc&quot; for xy+z+time maps or &quot;N%04d.asc&quot; for xy+time maps.</td>
</tr>
<tr>
<td>zoo.index</td>
<td>time or date index. Default is NULL, otherwise function returns a zoo object with</td>
</tr>
<tr>
<td></td>
<td>zoo.index as index.</td>
</tr>
<tr>
<td>ntime</td>
<td>number of time instant. If zoo.index is not NULL, it is calculated from zoo.index</td>
</tr>
<tr>
<td>nlayers</td>
<td>number of vertical layers.</td>
</tr>
</tbody>
</table>
read.ascii.vectorized.brick

Value

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

Author(s)

Emanuele Cordano

Description

Read a text file containing values and metadata 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

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

# 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

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

Arguments

x the character string containing the URL address
header_nrow Number of header in the ASCII grid format. Default is 6. See http://en.wikipedia.org/wiki/Esri_grid
... 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

read.vectorized.geotop.recovery

Reads a text file like the one generated by write.vectorized.geotop.recovery

Description

Reads a text file like the one generated by write.vectorized.geotop.recovery

# containing values and metadata 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

read.vectorized.geotop.recovery(file = file, comment = "!", matlb.syntax = TRUE, xx = "0000", formatter = "L%04d", extension = ".asc", NAflag = -9999, crs = "", ...)  

Arguments

file       file name to write  
comment    character. Comment indicator. Default is ".!".  
matlb.syntax logical value. Default is TRUE. If TRUE the file syntax is like the one of a *.m Matlab script file.  
formatter, extension, xx  
    see get.geotop.recovery.state.  
NAflag      numeric. Default is -9999, see writeRasterxGE0top.  
crs         Character or object of class CRS. PROJ4 type description of a Coordinate Reference System (map projection) (optional). See brick or raster.  
...         further arguments inserted as attribute  

Value

a list object like get.geotop.recovery.state  

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 meteorological station metadata, for instance, when the meteorological station of a study case is 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"
zl <- 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

`set.geotop.recovery.state(rec, newRecFolder, ...)"
vertical.aggregate.brick.within.depth

Arguments

- `rec` a list object returnened 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

# 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.
### Description

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

### Examples

```r
library(geotopbricks)
# The examples is the following R script contained
# 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

Usage

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

Arguments

- `b`: a `RasterBrick-class` or `GeotopRasterBrick-class` object
- `file`: file name to write
- `header`: character string vector for header text lines. If missing, a default header is written. Default is c("! header").
- `overwrite`: logical. Default is TRUE, see `writeRaster`.
- `NAflag`: numeric. Default is -9999, see `writeRasterxGE0top`.
- `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 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,"SnowThickness0000%04d.asc",sep="/"")
## b <- brick.decimal.formatter(file=file,nlayers=15)
## nlayers(b)
## names(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]]-b[[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

write.geotop.table(x, file, wpath = NULL, tz = "Etc/GMT-1",
         date_field = "Date12.DDMYYYhhmm.", 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.

tz  

time zone. Default is "Etc/GMT-1". See get.geotop.inpts.keyword.value for further details.

date_field  

string used for date-time field. Default is "Date12.DDMYYYhhmm.". See get.geotop.inpts.keyword.value for further details.

file_end  

suffix of the file name (file) (optional). Default is ",".

sep  

separator character. Default is ",,". See write.table for further details.

format  

date time format. Default is "%d/%m/%Y %H:%M". See get.geotop.inpts.keyword.value for further details.

na  

string for unassigned values. Default is "-9999". See write.table for further details.

...  

further arguments for write.table.

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

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

Arguments

<table>
<thead>
<tr>
<th>Argument</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>rec</td>
<td>a list object returned by <code>get.geotop.recovery.state</code></td>
</tr>
<tr>
<td>file</td>
<td>ascii text file name where to write the string vector</td>
</tr>
<tr>
<td>header</td>
<td>character string vector for header text lines. If missing, a default header is written. Default is c(&quot;! header&quot;) or the one assigned by <code>matlab.syntax</code>.</td>
</tr>
<tr>
<td>overwrite</td>
<td>logical. Default is TRUE, see <code>writeRaster</code>.</td>
</tr>
<tr>
<td>NAflag</td>
<td>numeric. Default is -9999, see <code>writeRasterxGeotop</code>.</td>
</tr>
<tr>
<td>matlab.syntax</td>
<td>logical value. Default is TRUE. If TRUE the file syntax is like the one of a *.m Matlab script file.</td>
</tr>
<tr>
<td>...</td>
<td>further arguments inserted as attribute</td>
</tr>
</tbody>
</table>

Value

a string vector containing the rec variables.

Note

Add Quote if necessary

See Also

- `get.geotop.recovery.state`
- `set.geotop.recovery.state`
- `write.vectorized.variable.in.string`

Examples

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

---

**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

See Also

read.ascii.vectorized.brick

Examples

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

writeRasterxGEOTop

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

Description

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

Usage

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

```
library(geotopbricks)

## Simulation working path

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

snowfile <- "/temporary/snow.asc"

dir.create("/temporary")
writeRasterxGEOtop(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

```
showClass("zoo")
```
Index

*Topic classes
  GeotopRasterBrick-class, 16
*Topic dataset
  bondone, 3
*Topic methods
  Ops, 29
  argsParser, 2
  as.POSIXlt, 13, 17
  bondone, 3
  brick, 4, 6, 31, 33, 36
  brick, GeotopRasterBrick-method (brick), 4
  brick, zoo-method (brick), 4
  brick.decimal.formatter, 5, 7, 9, 21, 25, 27, 41
  brickFromOutputSoil3DTensor, 6, 7, 8, 25–27
  color.bar, 10, 11
  color.bar.raster, 11
  create.geotop.inpts.keyword, 12
  create.geotop.meteo.files, 13
  data.frame, 18
  declared.geotop.inptskeywords, 12, 14, 17–19
  geotopbrick, 15
  GeotopRasterBrick, 24, 28, 29
  GeotopRasterBrick
    (GeotopRasterBrick-class), 16
  GeotopRasterBrick-class, 16
  get.geotop.inpts.keyword.value, 7, 9, 13, 14, 17, 25, 27, 38, 41
  get.geotop.recovery.state, 20, 33, 35, 38, 39
  getProjection, 22
  getvalues.brick.at.depth, 5, 22, 36
  identity, 35
  KML, 24, 30
  KML, GeotopRasterBrick-method (KML), 24
  list, 33
  listFromOutputSoil3DTensor, 24
  max_value, 28
  meteo(bondone), 3
  min_value, 28
  Ops, 29
  Ops, GeotopRasterBrick, GeotopRasterBrick-method (Ops), 29
  Ops, GeotopRasterBrick, numeric-method (Ops), 29
  Ops, numeric, GeotopRasterBrick-method (Ops), 29
  plot, 29
  plot, GeotopRasterBrick, ANY-method (plot), 29
  pointer.to.maps.xy.time, 4, 5, 15
  pointer.to.maps.xy.time
    (pointer.to.maps.xyz.time), 30
  pointer.to.maps.xyz.time, 4, 5, 15, 30
  POSIXlt, 17
  projectRaster, 24
  raster, 5, 6, 17, 18, 21, 31–33
  rasterFromOutput2DMap, 9, 27
  rasterFromOutput2DMap
    (brickFromOutputSoil3DTensor), 6
  read.ascii.vectorized.brick, 31, 37, 40
  read.raster.from.url, 5, 6, 18, 32
  read.table, 18
  read.vectorized.geotop.recovery, 21, 32
  readLines, 14, 32
  readLines, 14, 32
  replace.keyword, 33
set.geotop.recovery.state, 21, 34, 39
setMinMax, 11
sprintf, 13
system, 41
vertical.aggregate.brick.within.depth, 5, 23, 35
write.ascii.vectorized.brick, 31, 36
write.geotop.table, 38
write.table, 13, 38
write.vectorized.geotop.recovery, 21, 32, 33, 38
write.vectorized.variable.in.string, 39, 39
writeLines, 12
writeRaster, 37, 39–41
writeRasterxGEOTop, 31, 33, 35, 37, 39, 40, 40
zoo, 3, 18
zoo-class, 42