

# Package ‘geotopbricks’

March 4, 2024

**Maintainer** Emanuele Cordano <emanuele.cordano@gmail.com>

**License** GPL (>= 3)

**Title** An R Plug-in for the Distributed Hydrological Model GEOTop

**Type** Package

**Author** Emanuele Cordano

**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](http://www.geotop.org). Technical details about the model are available in Endrizzi et al (2014) <<https://gmd.copernicus.org/articles/7/2831/2014/gmd-7-2831-2014.html>>.

**Version** 1.5.8.2

**Date** 2024-03-04

**Repository** CRAN

**Depends** R (>= 2.10),methods,raster,stringr,zoo,sf

**Imports** terra

**Suggests** soilwater

**URL** <https://zenodo.org/record/8228413>, [www.geotop.org](http://www.geotop.org),  
<https://www.rendena100.eu/en/geotop-hydrological-model.html>,  
<https://github.com/ecor/geotopbricks>

**RoxygenNote** 7.3.1

**NeedsCompilation** no

**Date/Publication** 2024-03-04 20:40:03 UTC

## R topics documented:

argsParser . . . . .	2
bondone . . . . .	3

brick . . . . .	4
brick.decimal.formatter . . . . .	5
brickFromOutputSoil3DTensor . . . . .	7
create.geotop.inpts.keyword . . . . .	10
create.geotop.meteo.files . . . . .	12
declared.geotop.inpts.keywords . . . . .	13
geotopbrick . . . . .	14
GeotopRasterBrick-class . . . . .	15
get.geotop.inpts.keyword.value . . . . .	16
get.geotop.points . . . . .	20
get.geotop.recovery.state . . . . .	21
getProjection . . . . .	23
getvalues.brick.at.depth . . . . .	24
KML . . . . .	25
max_value . . . . .	26
min_value . . . . .	26
Ops . . . . .	27
plot . . . . .	27
pointer.to.maps.xyz.time . . . . .	28
read.ascii.vectorized.brick . . . . .	29
read.raster.from.url . . . . .	30
read.vectorized.geotop.recovery . . . . .	31
replace.keyword . . . . .	32
set.geotop.recovery.state . . . . .	33
vertical.aggregate.brick.within.depth . . . . .	34
write.ascii.vectorized.brick . . . . .	35
write.geotop.table . . . . .	37
write.vectorized.geotop.recovery . . . . .	38
write.vectorized.variable.in.string . . . . .	39
writeRasterxGEOtop . . . . .	40
zoo-class . . . . .	41
<b>Index</b>	<b>42</b>

---

argsParser

*Parser of an argument string*


---

## 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**

```
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

*Bondone Dataset*


---

**Description**

It contains hourly meteorological data observed at MeteoTrentino T0327 station located at Monte Bondone-Viotte (Trentino, Eastern 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

`Swg1ob` 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 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 (<https://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.

---

brick

*brick*

---

**Description**

Added implemetation 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 <a href="#">pointer.to.maps.xyz.time</a> or <a href="#">pointer.to.maps.xy.time</a> or a <a href="#">GeotopRasterBrick-class</a> object
layer	layer at which raster maps are imported. If is NULL, maps are no-zlayer distributed and zoo must be returned by <a href="#">pointer.to.maps.xy.time</a>
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 corresponding 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 <a href="#">RasterBrick-class</a>
use.read.raster.from.url	logical value. Default is TRUE. If TRUE the RasterLayer are read with <a href="#">read.raster.from.url</a> , instead of <a href="#">raster</a> (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.

**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" .
file_extension	string 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 <code>read.raster.from.url</code> , instead of <code>raster</code> (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 <code>raster(x)</code> does not always work and <code>use.read.raster.from.url</code> is necessary.
crs	coordinate system see <a href="#">RasterBrick-class,brick</a> , 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("example_files/examples/snowthickness",package="geotopbricks")
file <- paste(file,"SnowThickness0000L%04d.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)
```

---

```
brickFromOutputSoil3DTensor
      brickFromOutputSoil3DTensor
```

---

## 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%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,
  add_suffix_dir = NULL,
  ...
)

rasterFromOutput2DMap(x, when, ...)
```

## Arguments

x	string. GEOtop keyword related to the 3D or 2D variable to be imported in R.
when	<a href="#">POSIXct-class</a> 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 <a href="#">RasterLayer-class</a> object is imported, otherwise a <a href="#">RasterBrick-class</a> 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 <a href="#">get.geotop.inpts.keyword.value</a>
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 <a href="#">brick.decimal.formatter</a> . If crs is not NULL (Default), projfile is ignored.
projfile	name of the *.proj file containing CRS information. See <a href="#">get.geotop.inpts.keyword.value</a> . 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.
add_suffix_dir, ...	additional arguments for <a href="#">get.geotop.inpts.keyword.value</a> or <a href="#">brickFromOutputSoil3DTensor</a>

## 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 [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",  
"RelHumMapFiladd\_suffix\_dir=NULLle",  
"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](#)**Examples**

```

library(geotopbricks)
## Not run:
# The data containing in the link are only for educational use
wpath <- 'https://raw.githubusercontent.com/ecor/geotopbricks_doc/master/simulations/idroclim_test1'
## URL path (RAW VERSION) of
## https://github.com/ecor/geotopbricks_doc/tree/master/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 <- 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)

```

---

```
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",
  wpath = NULL,
  comment.lines = "default",
  header = "default",
  ...
)
```

**Arguments**

df	data frame returned by <a href="#">declared.geotop.inpts.keywords</a>
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 <a href="#">writeLines</a>

**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
wpath <-
'https://raw.githubusercontent.com/ecor/geotopbricks_doc/master/simulations/panola13_run2xC_test3'
## URL path (RAW VERSION) of
## https://github.com/ecor/geotopbricks_doc/tree/master/simulations/panola13_run2xC_test3
df <- declared.geotop.inpts.keywords(wpath=wpath)
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

```
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

<code>x</code>	'zoo' object or a list of 'zoo' object representing the meteorological station
<code>format</code>	string format representing the date, see <a href="#">as.POSIXlt</a> . Default is "%d/%m/%Y %H:%M" (which is the same format used in <code>geotop.inpts</code> keyword <code>InitDateDDMMYYYYhhmm</code> )
<code>file_prefix</code>	string containing file prefix (full path). It corresponds to the value of in <code>geotop.inpts</code> keyword <code>MeteoFile</code> )
<code>file_extension</code>	string containing the extensions of final files. Default is <code>c(".txt")</code>
<code>formatter</code>	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 <a href="#">sprintf</a> .
<code>na</code>	NA value indicator. Default is "-9999". See <a href="#">write.table</a> .
<code>col.names</code>	logical parameter. Default is TRUE. See <a href="#">write.table</a> .
<code>row.names</code>	logical parameter. Default is FALSE. See <a href="#">write.table</a> .
<code>date_field</code>	string value. Default is "Date", otherwise defined by the value of <code>HeaderDateDDMMYYYYhhmmMeteo</code> <code>geotop</code> keyword.
<code>sep</code>	string value. Default is ",". See <a href="#">write.table</a> .

level	integer argument. See <a href="#">get.geotop.inpts.keyword.value</a> for major details. Default is NULL and is ignored.
quote	logical parameter. Default is TRUE. See <a href="#">write.table</a> .
...	further arguments for <a href="#">write.table</a>

**See Also**

[write.table](#), [get.geotop.inpts.keyword.value](#)

**Examples**

```
library(geotopbricks)
data(bondone) ## It contains a "meteo" zoo object.

set.seed(12)

file_prefix <- paste(tempdir(),"meteo",sep="/")
level=2
out <- create.geotop.meteo.files(x=meteo,file_prefix=file_prefix,level=level)
## It exports the "meteo" zoo object into a ASCII file for GEOTop
head(readLines(out))
out
```

---

```
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**

```
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 charcater. 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 <a href="#">readLines</a> . Default is FALSE.
no.comment	string indicatos read as comment ones by GEOTop but they do not indicate comments by "geotopbricks" package.
...	further arguments of <a href="#">readLines</a>

**Value**

a data frame with two columns: Keyword and Value

**See Also**

[get.geotop.inpts.keyword.value](#)

---

geotopbrick

*geotopbrick*

---

**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 <a href="#">pointer.to.maps.xyz.time</a> or <a href="#">pointer.to.maps.xy.time</a> or a <a href="#">GeotopRasterBrick-class</a> object
...	further arguments.
layer	layer at which raster maps are imported. If is NULL, maps are no-layer distributed and zoo must be returned by <a href="#">pointer.to.maps.xy.time</a>
time	vector of time instants at which geotop maps are imported
crs	coordinate system see <a href="#">RasterBrick-class</a>
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**

```
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.syntax = 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,
MAXNROW = 4,
header.only = FALSE,
...
)

```

### Arguments

keyword	keyword name
inpts.frame	data frame returned by <a href="#">declared.geotop.inpts.keywords</a> or NULL. Default is NULL.
vector_sep	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 ", " by default.
col_sep	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 ", ".
numeric	logical value. If TRUE the Value has numeric type, otherwise it is a string or string vector. Default is FALSE.
format	string format representing the date, see <a href="#">as.POSIXlt</a> , used if date is TRUE. Default is "%d/%m/%Y %H:%M" (which is the format used in geotop.inpts keyword InitDateDDMMYYYYhhmm)
date	logical value. If TRUE the Value is returned as <a href="#">POSIXlt</a> date, otherwise it is a string or string vector. Default is FALSE.
tz	format string representing the time zone, see <a href="#">as.POSIXlt</a> , used if date is TRUE. Default is "Etc/GMT-1" (until the previous version it was "A") which means UTC +1.
raster	logical value. Default is FALSE. If TRUE function returns directly the raster map as <a href="#">Raster-class</a> object built with <a href="#">raster</a> 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 <a href="#">declared.geotop.inpts.keywords</a> .
use.read.raster.from.url	logical value. Default is TRUE. If TRUE the RasterLayer are read with <a href="#">read.raster.from.url</a> , instead of <a href="#">raster</a> (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.

<code>data.frame</code>	logical value. It is an option for tabular data. If TRUE function returns directly a data frame or a list of data frames as <code>data.frame</code> or <code>zoo</code> objects imported from the keyword-related files using <code>read.table</code> function. In this case the argument <code>wpath</code> (see <code>declared.geotop.inpts.keywords</code> ) is mandatory. Default is FALSE.
<code>formatter</code>	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 <code>data.frame</code> is TRUE.
<code>level</code>	integer values. Numbers incating all the identification numbers of the files containing the requested data frames. Default is 1, correspondig to the decimal formatter "0001". See examples.
<code>date_field</code>	string value. Default is "Date", otherwise defined by the value of <code>HeaderDateDDMMYYYYhhmmMeteo</code> geotop keyword. It is used only if the argument <code>data.frame</code> is TRUE. If it is NULL or NA the function return a list of generic <code>data.frame</code> object(s), otherwise <code>link{zoo}</code> object(s). See the arguments <code>tz</code> and <code>format</code> for Date formatting.
<code>isNA</code>	numeric value indicating NA in geotop ascii files. Default is -9999.00
<code>matlab.syntax</code>	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.
<code>projfile</code>	filename of the GEOTop projection file. Default is <code>geotop.proj</code> .
<code>start_date, end_date</code>	null objects or dates in POSIXlt format between which the variables are returned. It is enabled in case that <code>date_field</code> is not NULL or NA and <code>data.frame</code> is TRUE. Default is NULL.
<code>ContinuousRecovery</code>	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 'Contiuous Recovery' option.
<code>ContinuousRecoveryFormatter</code>	character string. Default is '_crec%04d'. It is used only for tabular output data and if <code>ContinuousRecovery</code> is equal or greater than 1.
<code>zlayer.formatter</code>	decimal formatter. It is used if <code>data.frame==TRUE</code> and the columns refers to different soil depths. Default is NULL.
<code>z_unit</code>	z coordinate measurement unit. GEOTop values expressed in millimeters which are converted to centimeters by default. Default is <code>c("centimeters", "millimeters")</code> . Otherwise can be the ratio between the unit and one meter. It is used if <code>zlayer.formatter=="z%04d"</code> or similar.
<code>geotop_z_unit</code>	z coordinate measurement unit used by GEOTop. Default is millimeters. It is used if <code>zlayer.formatter=="z%04d"</code> or similar.
<code>add_suffix_dir</code>	character string. Add a suffix at the directory reported in the keyword value
<code>MAXNROW</code>	maximum number accepted for <code>data.frema</code> output. Default is 4. It is used in case of <code>data.frame==TRUE</code> . In case the number of records in the function output is less than <code>MAXNROW</code> , function returns neither <code>data.frame</code> nor <code>zoo</code> 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

wpath <-
'https://raw.githubusercontent.com/ecor/geotopbricks_doc/master/simulations/panola13_run2xC_test3'
## URL path (RAW VERSION) of
## https://github.com/ecor/geotopbricks_doc/tree/master/simulations/panola13_run2xC_test3
prefix <- get.geotop.inpts.keyword.value("SoilLiqWaterPressTensorFile",wpath=wpath)

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

tz <- "Etc/GMT-1" ## See help(timezones) In particular:
## Most platforms support time zones of the form Etc/GMT+n
## and Etc/GMT-n (possibly also without prefix Etc/),
```

```

## which assume a fixed offset from UTC (hence no DST).
## Contrary to some expectations
## (but consistent with names such as PST8PDT), negative offsets are times ahead of (east of) UTC,
## positive offsets are times behind (west of) UTC.
start <- get.geotop.inpts.keyword.value("InitDateDDMMYYYYhhmm",
date=TRUE,wpath=wpath,tz=tz)
end <- get.geotop.inpts.keyword.value("EndDateDDMMYYYYhhmm",
date=TRUE,wpath=wpath,tz=tz)

nmeteo <- get.geotop.inpts.keyword.value("NumberOfMeteoStations",
numeric=TRUE,wpath=wpath)
level <- 1:nmeteo

## set meteo data

meteo <- get.geotop.inpts.keyword.value("MeteoFile",wpath=wpath,data.frame=TRUE,
level=level,start_date=start,end_date=end,tz=tz)

##### end set meteo data

## IMPORTING AN OUTPUT SOIL MOISTURE PROFILE:

wpath <- paste0(
'https://raw.githubusercontent.com/ecor/geotopbricks_doc/',
'master/simulations/Muntatschini_pnt_1_225_B2_004')
## URL Path (RAW VERSION) of
## https://github.com/ecor/geotopbricks_doc/tree/master/simulations/Muntatschini_pnt_1_225_B2_004

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")

```

---

get.geotop.points

*Get a **sf** object for Meteorological Stations or Control Points in a GEOtop simulation*


---

**Description**

Get a `sf` object for Meteorological Stations or Control Points in a GEOtop simulation

**Usage**

```
get.geotop.points(
  prefix = c("MeteoStation", "CoordinatePoint"),
  suffixes = c("Code", "Elevation", "Source"),
  coord_suffixes = list(MeteoStation = c("CoordinateX", "CoordinateY"), CoordinatePoint =
    c("X", "Y")),
  wpath,
  ...,
  vector_sep = ", "
)
```

**Arguments**

prefix	keyword prefix
suffixes	keyword suffixes
coord_suffixes	coordinate keyword suffixes. Default is <code>c("PointX", "PointY")</code>
wpath	GEOtop simulation path
vector_sep, ...	further arguments for <code>get.geotop.inpts.keyword.value</code>

**Examples**

```
###See simulation template: "https://github.com/ecor/geotopbricks_doc/tree/master/template/sumava"
wpath <- "https://raw.githubusercontent.com/ecor/geotopbricks_doc/master/template/sumava/"
## system.file("template/sumava",package="geotopbricks")
out <- get.geotop.points(wpath=wpath)
out <- get.geotop.points(prefix="CoordinatePoint",suffix=c("Code","Source"),wpath=wpath)
out <- get.geotop.points(prefix="MeteoStation",suffix=c("Code","Source"),wpath=wpath)
```

---

```
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,
  ...
)

```

**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 estension used for ascii recovery map files. It must contains ' .' as the first character. Defaut 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 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](#),  
[raster,set.geotop.recovery.state](#),  
[write.vectorized.geotop.recovery,read.vectorized.geotop.recovery](#)

**Examples**

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

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

---

getProjection	<i>It reads the CRS metadata utilized in a GEOtop Simulation</i>
---------------	--

---

**Description**

It reads the CRS metadata utilized in a GEOtop Simulation

**Usage**

```
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**

```
library(geotopbricks)

wpath <- 'https://raw.githubusercontent.com/ecor/geotopbricks_doc/master/simulations/idroclim_test1'
## URL path (RAW VERSION) of
## https://github.com/ecor/geotopbricks_doc/tree/master/simulations/idroclim_test1
## Not run:

x <- paste(wpath, "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

[vertical.aggregate.brick.within.depth](#)



## Examples

```
library(geotopbricks)
# The examples is the following R script contained 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 <a href="#">GeotopRasterBrick</a> 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 <a href="https://spatialreference.org/ref/epsg/4326/">https://spatialreference.org/ref/epsg/4326/</a> ).
...	further argument for S4 method KLM for Raster object.

## Note

A coordinate transformation is made with [projectRaster](#).

## Examples

```
library(geotopbricks)
# The examples is the following R script contained in a 'inst' directory of the package source
f <- system.file("doc/examples/example.KML.GeotopRasterBrick.R", package="geotopbricks")
## Not run:
source(f) # Uncomment this line to run the example.
```

```
## End(Not run)
# You can copy the example file using file.copy(from=f,to=...,...) See file.copy documentation
```

---

<code>max_value</code>	<i>max_value</i>
------------------------	------------------

---

### Description

Gets the maximum (scalar) values of a [GeotopRasterBrick](#) object

### Usage

```
max_value(x, na.rm = TRUE, ...)
```

### Arguments

`x` a [GeotopRasterBrick](#) object.  
`na.rm, ...` further arguments for `max`.

### Value

the maximum (scalar) values of a [GeotopRasterBrick](#) object

---

<code>min_value</code>	<i>min_value</i>
------------------------	------------------

---

### Description

Gets the minimum (scalar) values of a [GeotopRasterBrick](#) object

### Usage

```
min_value(x, na.rm = TRUE, ...)
```

### Arguments

`x` a [GeotopRasterBrick](#) object.  
`na.rm, ...` further arguments for `min`.

### 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

*plot*


---

**Description**

plot method for a GeotopRasterBrick object

**Usage**

```
## 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**

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

*pointer.to.maps.xyz.time*

---

**Description**

'pointer.to.maps.xyz.time' function (obsolete)

**Usage**

```
pointer.to.maps.xyz.time(  
  wpath,  
  map.prefix = "thetaliq",  
  suffix = "L%04dN%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%04dN%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**

```
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 <a href="#">brick</a> or <a href="#">raster</a> .
NAflag	numeric. Dafauli is -9999, see <a href="#">writeRasterxGEOtop</a> .
matlab.syntax	logical value. Default is FALSE. If TRUE the file syntax is like the one of a *.m Matlab script file.
...	further aguments 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-Ascii format from a URL
                    'http(s)://...<FILENAME>.asc
```

---

**Description**

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

**Usage**

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

**Arguments**

<code>x</code>	the charcater string containing the URL address
<code>header_nrow</code>	Number of header in the ASCII grid format. Deafault is 6. See <a href="https://en.wikipedia.org/wiki/Esri_grid">https://en.wikipedia.org/wiki/Esri_grid</a>
<code>...</code>	additional arguments

**Value**

a 'RasterLayer' object

**Note**

This function reads a local or remote text files formatted as [https://en.wikipedia.org/wiki/Esri\\_grid](https://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

#. 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 = "!",
    matlab.syntax = TRUE,
    xx = "0000",
    formatter = "%04d",
    extension = ".asc",
    NAflag = -9999,
    crs = "",
    ...
)
```

### Arguments

file	file name to write
comment	character. Comment indicator. Default is "!".
matlab.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 <a href="#">get.geotop.recovery.state</a> .
NAflag	numeric. Default is -9999, see <a href="#">writeRasterxGEOtop</a> .
crs	Character or object of class CRS. PROJ4 type description of a Coordinate Reference System (map projection) (optional). See <a href="#">brick</a> or <a href="#">raster</a> .
...	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	<i>It replaces some keyword values of geotop.inpts file with the ones of anoter *.inpts value</i>
-----------------	---

---

**Description**

It replaces some keyword values of geotop.inpts file with the ones of anoter \*.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 x. It is useful to replace the meteorological 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

```
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

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

## Arguments

rec	a list object returned by <a href="#">get.geotop.recovery.state</a>
newRecFolder	directory where to write all recovery raster ascii maps
...	further arguments

## Author(s)

Emanuele Cordano

## See Also

[get.geotop.recovery.state](#), [writeRasterxGEOtop](#)

## 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, <code>identity</code> 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.

**See Also**

[getvalues.brick.at.depth,brick](#)

**Examples**

```
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
```

*Writes 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	a <a href="#">RasterBrick-class</a> or <a href="#">GeotopRasterBrick-class</a> object
file	file name to write
header	character string vector for header text lines. If missing, a default header is written. #Default is <code>c("! header")</code> .
overwrite	logical. Default is TRUE, see <a href="#">writeRaster</a> .
NAflag	numeric. Default is -9999, see <a href="#">writeRasterxGEOtop</a> .
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**

```
## Not Run
## library(geotopbricks)
## library(raster)
## file <- system.file("doc/examples/snowthickness", package="geotopbricks")
## file <- paste(file,"SnowThickness0000L%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	<i>Writes an R object (data.frame or zoo) into a CSV file readable by GEOtop.</i>
--------------------	---

---

### 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.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.
tz	time zone. Default is "Etc/GMT-1". See <a href="#">get.geotop.inpts.keyword.value</a> for further details.
date_field	string used for date-time field. Default is "Date12.DDMMYYYYhhmm.". See <a href="#">get.geotop.inpts.keyword.value</a> for further details.
file_end	suffix of the file name (file) (optional). Default is "".
sep	separator character. Default is ",". See <a href="#">write.table</a> for further details.
format	date time format. Default is "%d/%m/%Y %H:%M". See <a href="#">get.geotop.inpts.keyword.value</a> for further details.
na	string for unassigned values. Default is "-9999". See <a href="#">write.table</a> for further details.
...	further arguments for <a href="#">write.table</a> .

---

```
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

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

### Arguments

rec	a list object returned by <a href="#">get.geotop.recovery.state</a>
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 <a href="#">writeRaster</a> .
NAflag	numeric. Default is -9999, see <a href="#">writeRasterxGEOtop</a> .
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.

### 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 list object contained the variables (scalars, vectors or Rasters) which will be written in a string each.
NAflag	numeric. Default is -9999, see <a href="#">writeRasterxGEOtop</a> .
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	<i>This function uses <a href="#">writeRaster</a> to create .asc maps which can be read by <a href="#">GEOtop</a></i>
--------------------	---

---

**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 <a href="#">writeRaster</a> . It can be also a <a href="#">RasterBrick-class</a> object.
filename	see <a href="#">writeRaster</a> . It is a vector of string or one string containing a decimal formatter (see <a href="#">brick.decimal.formatter</a> ) in case x is a <a href="#">RasterBrick-class</a> object.
overwrite	logical. Default is TRUE, see <a href="#">writeRaster</a> .
NAflag	numeric. Default is -9999, see <a href="#">writeRaster</a> .
use.decimal.formatter	logical value. Default is FALSE. If it is TRUE or x is a <a href="#">RasterBrick-class</a> object with <code>nlayers(x) != length(filename)</code> , filename is considered as one string containing a decimal formatter (e.g. "%04d", see <a href="#">brick.decimal.formatter</a> ). 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 ex-
                    tension 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 <- paste0("https://raw.githubusercontent.com/ecor/geotopbricks_doc/master/template/",
#"rendena100/SnowDepthMapFile-2014-MA-mean-winter-2013-2014.asc")
file <- system.file("ex/elev.tif", package="terra")
elev <- raster(file)

elevfile <- rasterTmpFile()
extension(elevfile) <- ".asc"

writeRasterxGEOtop(x=elev,file=elevfile)

```

---

zoo-class	<i>A GeotopRasterBrick: an object to manage raster maps provided by GEOtop!!</i>
-----------	--

---

**Description**

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

**Examples**

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

# Index

- \* **classes**
  - GeotopRasterBrick-class, 15
- \* **dataset**
  - bondone, 3
- \* **methods**
  - Ops, 27
- argsParser, 2
- as.POSIXlt, 12, 17
- bondone, 3
- brick, 4, 6, 29, 31, 35
- brick, GeotopRasterBrick-method (brick), 4
- brick, zoo-method (brick), 4
- brick.decimal.formatter, 5, 8, 10, 22, 40
- brickFromOutputSoil3DTensor, 7, 8, 9
- create.geotop.inpts.keyword, 10
- create.geotop.meteo.files, 12
- data.frame, 18
- declared.geotop.inpts.keywords, 10, 11, 13, 17–19
- geotopbrick, 14
- GeotopRasterBrick, 25–27
- GeotopRasterBrick
  - (GeotopRasterBrick-class), 15
- GeotopRasterBrick-class, 15
- get.geotop.inpts.keyword.value, 8, 10, 13, 14, 16, 21, 37, 41
- get.geotop.points, 20
- get.geotop.recovery.state, 21, 31, 33, 38
- getProjection, 23
- getvalues.brick.at.depth, 5, 24, 35
- identity, 34
- KML, 25, 28
- KML, GeotopRasterBrick-method (KML), 25
- list, 31
- max, 26
- max\_value, 26
- meteo (bondone), 3
- min, 26
- min\_value, 26
- Ops, 27
- Ops, GeotopRasterBrick, GeotopRasterBrick-method (Ops), 27
- Ops, GeotopRasterBrick, numeric-method (Ops), 27
- Ops, numeric, GeotopRasterBrick-method (Ops), 27
- plot, 27
- plot, GeotopRasterBrick, ANY-method (plot), 27
- pointer.to.maps.xy.time, 5, 15
- pointer.to.maps.xy.time
  - (pointer.to.maps.xyz.time), 28
- pointer.to.maps.xyz.time, 5, 15, 28
- POSIXlt, 17
- projectRaster, 25
- raster, 5, 6, 17, 22, 29–31
- rasterFromOutput2DMap, 9
- rasterFromOutput2DMap
  - (brickFromOutputSoil3DTensor), 7
- read.ascii.vectorized.brick, 29, 36, 39
- read.raster.from.url, 5, 6, 17, 30
- read.table, 18
- read.vectorized.geotop.recovery, 22, 31
- readLines, 14, 30
- replace.keyword, 32
- set.geotop.recovery.state, 22, 33, 38
- sf, 20, 21
- sprintf, 12

system, [41](#)

vertical.aggregate.brick.within.depth,  
[5](#), [24](#), [34](#)

write.ascii.vectorized.brick, [30](#), [35](#)

write.geotop.table, [37](#)

write.table, [12](#), [13](#), [37](#)

write.vectorized.geotop.recovery, [22](#),  
[31](#), [38](#)

write.vectorized.variable.in.string,  
[38](#), [39](#)

writeln, [11](#)

writeRaster, [36](#), [38](#), [40](#), [41](#)

writeRasterxGEOTop, [29](#), [31](#), [33](#), [36](#), [38](#), [39](#),  
[40](#)

zoo, [3](#), [18](#)

zoo-class, [41](#)