Package ‘getMet’

March 21, 2016

Title  Get Meteorological Data for Hydrologic Models
Version  0.3.2
Description  Hydrologic models often require users to collect and format input meteorological data. This package contains functions for sourcing, formatting, and editing meteorological data for hydrologic models.
Depends  R (>= 3.0.0)
Imports  EcoHydRology, jsonlite
License  GPL-3
LazyData  true
LazyLoad  true
RoxygenNote  5.0.0
NeedsCompilation  no
Author  Andrew Sommerlot [aut, cre],
      Daniel Fuka [aut],
      Zachary Easton [aut]
Maintainer  Andrew Sommerlot <andrewrs@vt.edu>
Repository  CRAN
Date/Publication  2016-03-21 00:31:42

R topics documented:

   genSWATdates  ......................................................... 2
   getMet ................................................................. 2
   getSWATcfsr .......................................................... 3
   getSWATdates .......................................................... 4
   getSWATwunderForecast ............................................. 4
   SWATsubGage ......................................................... 5

Index  7
getMet

Gets met data from a specified source and creates model input files in the specified format

Usage

gGetMet(locations, dataSource = "cfsr", outFormat = "swat",
outDir = getwd(), apiKey = "")
getSWATcfsr

Arguments

locations - data.frame object or location of csv file with two columns, the first being latitude and second being longitude in decimal degrees. These locations will be the locations of the collected met data.
dataSource - Source of met data from predefined source list. Currently only 'cfsr' is supported.
outFormat - Format of met data output from predefined source. Currently only 'swat' is supported.
outdir - Directory where output files will be saved
apiKey - String input of api key for selected data source if required.

Value

returns specified met data in specified format

Examples

## Not run:
locations = data.frame(lat = 38, lon = 79)
outdir = "test"
getMet(locations=locations, outDir=outDir, dataSource = 'cfsr', outFormat = 'swat')

## End(Not run)

---

getSWATcfsr

*Gets met data from the redimensioned CFSR data set and outputs SWAT IO format meteorological input files*

Description

Gets met data from the redimensioned CFSR data set and outputs SWAT IO format meteorological input files

Usage

getSWATcfsr(centroids, outDir = getwd())

Arguments

centroids - data.frame object or location of csv file with two columns, the first being latitude and second being longitude in decimal degrees. These should be the centroids of the swat model subbasins and are the locations of the output met data.
outdir - Directory where output files will be saved.

Value

returns cfsr met data in swat IO format
getSWATwunderForecast

Examples

```r
## Not run:
centroids = data.frame(lat = 38, lon = 79)
outdir = "test"
getSWATcfsr(centroids=centroids, outdir=outdir)

## End(Not run)
```

getSWATdates

Converts a date time series to SWAT IO format dates

Description

Converts a date time series to SWAT IO format dates

Usage

```r
getSWATdates(dates, dateformat = "yyyy-mm-dd")
```

Arguments

- `dates`: Input vector of dates in as Date or character class.
- `dateformat`: Format of input dates. Can be 'yyyy-mm-dd', 'mm-dd-yyyy', 'mm/dd/yyyy', 'yyyy/mm/dd','dd/mm/yyyy', or 'dd-mm-yyyy'. Default is 'yyyy-mm-dd'.

Value

returns a time series of swat IO format dates from the input

Examples

```r
dates = c("2000-12-28", "2000-12-29")
getSWATdates(dates)
```

getSWATwunderForecast

Gets met data from the wunderground 10 day forecast and outputs SWAT IO format meteorological input files

Description

Gets met data from the wunderground 10 day forecast and outputs SWAT IO format meteorological input files

Usage

```r
getSWATwunderForecast(centroids, outDir = getwd(), apiKey)
```
Arguments

- **centroids** - data.frame object or location of csv file with two columns, the first being latitude and second being longitude in decimal degrees. These should be the centroids of the swat model subbasins and are the locations of the output met data.
- **outdir** - Directory where output files will be saved.
- **apiKey** - String input of your freely available api key from wunderground.com. See https://www.wunderground.com/weather/api/d/pricing.html for more details. Select the ANVIL version of the developer plan, record the resulting key and use it in this function.

Value

returns wunderground 10 day forecast met data in swat IO format

Examples

```r
## Not run:
centroids = data.frame(lat = 38, lon = 79)
outdir = "test"
getSWATwunderForecast(centroids=centroids, outdir=outdir)

## End(Not run)
```

SWATsubGage

Formats SWAT subbasin files to user specifications. Defaults support use of getSWATcfsr.

Description

Formats SWAT subbasin files to user specifications. Defaults support use of getSWATcfsr.

Usage

```r
SWATsubGage(wd, outDir = "", numPars = 5, basinCentroid = FALSE)
```

Arguments

- **wd** - String. location of the subbasin files to be formatted. Generally this is the TxtInOut folder generated when building a SWAT project.
- **outdir** - String. The location to write the formatted subbasin files. By default, this is the same as the wd, and will overwrite existing subbasin files.
- **numPars** - Integer. Number of measured weather parameters used in the model. Ranges from 1 to 5 with 1 as precipitation only, 2 as precipitation and temperature, 3 as precipitation, temperature, and solar radiation, 4 as precipitation, temperature, solar radiation, and relative humidity, and 5 as precipitation, temperature, solar radiation, relative humidity, and wind speed. Default is 5, or all parameters are formatted as measured inputs.
basinCentroid - Logical. If TRUE, then all gage location flags are set to "1", meaning there is a single time series for each of the specified measured meteorological inputs. Only use if a subbain center approximation of meteorological data is being used. Defaults to FALSE, or each subbasin has a corresponding time series in each measured meteorological input file.

Value

returns formatted subbasin files in the outDir location. If outDir is not specified files are saved to the wd location and any existing files are overwritten.

Examples

```r
## Not run:
SWATsubGaged(wd = '~/SWAT_PROJECT_FOLDER/TxtInOut')

## End(Not run)
```
Index

genSWATdates, 2
getMet, 2
getSWATcfsr, 3
getSWATdates, 4
getSWATwunderForecast, 4

SWATsubGage, 5