Package 'meteoland'

May 29, 2019

Type Package

Title Landscape Meteorology Tools

Version 0.8.1

Date 2019-05-08

Description Functions to estimate weather variables at any position of a landscape [De Caceres et al. (2018) <doi:10.1016/j.envsoft.2018.08.003>].

License GPL (>= 2)

URL http://vegmod.ctfc.cat/meteolandweb

LazyData TRUE

Depends R (>= 3.4.0), sp

Imports spdep, rgdal, ncdf4, ncdf4.helpers, httr, jsonlite, methods, Rcpp (>= 0.12.12)

Suggests knitr

LinkingTo Rcpp

Encoding UTF-8

NeedsCompilation yes

VignetteBuilder utils, knitr

Author Miquel De Cáceres [aut, cre], Nicolas Martin [aut], Víctor Granda [aut], Antoine Cabon [aut]

Maintainer Miquel De Cáceres <miquelcaceres@gmail.com>

Repository CRAN

Date/Publication 2019-05-29 13:40:03 UTC

R topics documented:

AEMET download *Download data from AEMET*

Description

Download data from the Spanish National Meterology Agency (AEMET)

Usage

```
downloadAEMEThistoricalstationlist(api)
downloadAEMEThistorical(api, dates, station_id, export = FALSE, exportDir = getwd(),
                    exportFormat = "meteoland/txt", metadatafile = "MP.txt",
                    verbose = TRUE)
downloadAEMETcurrentday(api, daily = TRUE, verbose = TRUE)
```

Arguments

api	String with the AEMET API key (see https://opendata.aemet.es/).
dates	An object of class Date.
station_id	A string vector containing station ids (the list of stations for which historical climatic series are available is given by downloadAEMEThistoricalstationlist).
export	If export = FALSE the downloaded data is stored in memory. Otherwise the result is written on the disk (using the format specified in exportFormat).
exportDir	Output directory for downloaded meteorology.
exportFormat	Format of meteorological data. Current accepted formats are "castanea" and "meteoland".
metadatafile	The name of the file that will store the meta data describing all written files.
verbose	Boolean flag to print process information.
daily	Boolean flag. Are data to be returned at a daily or hourly scale?

Details

API key needs to be acquired from AEMET (https://opendata.aemet.es/)

Value

Function downloadAEMEThistoricalstationlist returns a SpatialPointsDataFrame-class object containing the list of AEMET weather stations for which historical climatic series are available and can be retrieved using downloadAEMEThistorical.

Function downloadAEMEThistorical downloads data for the specified AEMET stations and dates. Data are availables for dates up to 4 days before current date. If export = FALSE, function downloadAEMEThistorical returns a SpatialPointsMeteorology-class object with the downloaded meteorology for each station (point). Otherwise the function writes on the disk at the location specified by exportDir and solelely returns a SpatialPointsDataFrame-class object containing the files metadata.

Function downloadAEMETcurrentday downloads recent weather (the last 24h) from all currently available stations and returns data frame if daily = FALSE or a SpatialPointsDataFrame-class object with observations aggregated at the daily scale if else.

Note

The list of stations available in downloadAEMETcurrentday (current observations) is different from the list given by downloadAEMEThistoricalstationlist and available in downloadAEMEThistorical (stations with historical climate series).

Author(s)

Antoine Cabon, CTFC

Miquel De Cáceres Ainsa, CTFC

References

AEMET should be acknowledged as author of information when using this data.

See Also

SpatialPointsMeteorology-class

---

correctionpoints *Statistical correction of meteorological variables for a set of points*

Description

Functions correctionpoint and correctionpoints perform correction of predicted climatic data by applying statistical correction methods (unbiasing, scaling, or quantile mapping) to meteorological variables. Function correctionpoints.errors allows evaluating, for each point, the bias and mean absolute error (MAE) obtained before and after correcting the climate model for the historical period.

Usage

```
correctionpoint(obs, mod, proj, dates = NULL,
                params = defaultCorrectionParams(), verbose=TRUE)
correctionpoints(object, points, topodata = NULL, dates = NULL,
             export = FALSE, exportDir = getwd(), exportFormat = "meteoland/txt",
                metadatafile = "MP.txt", corrOut = FALSE, verbose = TRUE)
correctionpoints.errors(object, points, topodata = NULL,
                 error.type="residuals.cv",keep.data = FALSE, verbose = FALSE)
```

Arguments

obs	A data frame with observed meteorology.
mod, proj	Data frame with predicted meteorology for the reference and projection periods, respectively.
params	A list with correction params (see defaultCorrectionParams).
object	An object of class MeteorologyUncorrectedData-class containing the meteorology of more than one point.
points	An object of class SpatialPointsMeteorology-class with the coordinates and historical meteorological data of the locations for which correction of predicted climatic data has to be done. Alternatively, an object of class SpatialPointsDataFrame-class containing the meta data (columns dir, filename and possibly format) of meteorological files that will be read from the disk.
topodata	A data frame with topographic data for each point (i.e. three columns named elevation, slope and aspect). If topodata = NULL then Penman's potential evapotranspiration is not calculated.
dates	An object of class Date with a subset of dates of the projection period to be corrected. If dates = NULL then all dates in proj or the projection data of object are processed.
export	If export = FALSE the result of correction is stored in memory. Otherwise the result is written in the disk (using the format specified in exportFormat).
exportDir	Output directory for corrected meteorology.
metadatafile	The name of the file that will store the meta data describing all written files.
exportFormat	Export format for meteorological data (see writemeteorologypoint).
corrOut	Boolean flag to indicate that correction parameters (i.e. calculated biases) should be included with the output. Setting corrOut = TRUE changes the returned value.
verbose	Boolean flag to print process information.
error.type	String to specify the error to be evaluated, either "before" (before correction), "residual" (after correction) or "residual.cv" (after correction, but using cross-validation).
keep.data	Boolean flag to return the uncorrected/corrected data for the historical period.

Details

Function correctionpoints performs statistical correction of predicted climatic data for all points supplied in points whereas correctionpoint performs statistical correction of one single point. Observed meteorological data for each point typically comes from a nearby meteorological station, but they can be the result of interpolating the meteorology of several stations (see MeteorologyInterpolationData) or they can be extracted from reanalyzed meteorology (e.g. EU-WATCH) (see extractNetCDF).

For each target point, correctionpoints function first determines the predicted cell where the point falls. Then it calls correctionpoint. In turn, correctionpoint determines the dates that are shared in observed and predicted data for the historical period. These meteorological data of dates are used to conduct the correction of predicted climatic data for the future period. Corrections biases are calculated and applied for the twelve months separately. The user can control the

methods used for correction of each meteorological variable by changing the slot params in object (see class MeteorologyUncorrectedData-class) or the parameter params to correctionpoint. Three options are allowed (see defaultCorrectionParams): (a) 'unbias' for shifting the mean; (b) 'scaling' for multiplication by a factor; and (c) 'quantmap' for empirical quantile mapping between observed and modelled data (Déqué 2007).

A difficulty arises for quantile mapping when the variables bounded by zero, such as precipitation. As the models tend to drizzle (or may have lower frequency of precipitation events), the probability of precipitation in the model may be greater or lower than that observed. To correct this, when model precipitation is zero an observed value is randomly chosen in the interval where the observed cumulative frequency is less than or equal to the probability of no precipitation in the model. This procedure ensures that the probability of precipitation after correction is equal to that observed (Boé 2007).

Value

If export = FALSE, the function correctionpoints returns an object of class SpatialPointsMeteorology-class with the downscaled meteorology for each point. If export=TRUE function correctionpoints returns an object of class SpatialPointsDataFrame-class containing the meta data of the files written in the disk. If corrOut = TRUE the previous structures are embedded in a list which also contains an object with the calculated correction factors (biases, mappings) for each point and month. Function correctionpoints.errors (keep.data = FALSE) returns a data frame with the mean absolute error (MAE) and bias for each variable and point. If keep.data = TRUE then the function also returns a list of data frames with the uncorrected/corrected series used in the comparisons with observations.

Author(s)

Miquel De Cáceres Ainsa, Centre Tecnologic Forestal de Catalunya

Nicolas Martin, INRA-Avignon

References

Boé J, Terray L, Habets F, Martin E (2007) Statistical and dynamical downscaling of the Seine basin climate for hydro-meteorological studies. Int J Climatol 27:1643–1655. doi: 10.1002/joc.1602

De Caceres M, Martin-StPaul N, Turco M, Cabon A, Granda V (2018) Estimating daily meteorological data and downscaling climate models over landscapes. Environmental Modelling and Software 108: 186-196.

Déqué M (2007) Frequency of precipitation and temperature extremes over France in an anthropogenic scenario: Model results and statistical correction according to observed values. Glob Planet Change 57:16–26. doi: 10.1016/j.gloplacha.2006.11.030

See Also

penman, extractNetCDF, SpatialPointsMeteorology-class, writemeteorologypointfiles, MeteorologyUncorrectedData, MeteorologyInterpolationData

Examples

```
data(examplegridtopography)
data(exampleinterpolationdata)
data(examplecorrectiondata)

#Creates spatial topography points from the grid
p = 1:2
spt = as(examplegridtopography, "SpatialPointsTopography")[p]


#Interpolation of two points for the whole time period (2000-2003)
historical = interpolationpoints(exampleinterpolationdata, spt)

#Downscaling of future predictions (RCM models, year 2023)
predicted = correctionpoints(examplecorrectiondata, historical, spt@data)

#Plot predicted mean temperature for point 1
meteoplot(predicted, 1, "MeanTemperature", ylab="Temperature (Celsius)", ylim=c(-5,40))
meteoplot(predicted, 1, "MinTemperature", add=TRUE, col="blue")
meteoplot(predicted, 1, "MaxTemperature", add=TRUE, col="red")
#Add uncorrected mean temperature data (cell #3)
lines(examplecorrectiondata@dates,
      examplecorrectiondata@projection_data[[3]]$MeanTemperature,
      lty=3)
lines(examplecorrectiondata@dates,
      examplecorrectiondata@projection_data[[3]]$MinTemperature,
      col="blue", lty=3)
lines(examplecorrectiondata@dates,
      examplecorrectiondata@projection_data[[3]]$MaxTemperature,
      col="red", lty=3)
legend("topright", legend=c("corrected","uncorrected", "Maximum", "Mean", "Minimum"),
       col=c("black","black", "red","black","blue"), lty=c(1,3,1,1,1), bty="n")

#Scatter plot
plot(examplecorrectiondata@projection_data[[3]]$MeanTemperature,
     predicted@data[[1]]$MeanTemperature, cex=0.1, asp=1,
     ylab="Corrected mean temperature", xlab="Uncorrected mean temperature")
abline(a=0,b=1,col="gray")

#Plot predicted precipitation for point 1
meteoplot(predicted, 1, "Precipitation", ylab="Precipitation (mm)", ylim=c(0,120))
#Add uncorrected mean temperature data (cell #3)
lines(examplecorrectiondata@dates,
      examplecorrectiondata@projection_data[[3]]$Precipitation,
      col="red", lty=3)
legend("topleft", legend=c("corrected","uncorrected"), col=c("black","red"), lty=c(1,3), bty="n")

#Scatter plot
plot(examplecorrectiondata@projection_data[[3]]$Precipitation,
     predicted@data[[1]]$Precipitation, cex=0.1, asp=1,
     ylab="Corrected precipitation (mm)", xlab="Uncorrected precipitation (mm)")
abline(a=0,b=1,col="gray")
```

`correction_lowlevel` *Low-level correction functions*

Description

Low-level function to perform bias correction.

Usage

```
correction_series(obs, mod, proj = NULL, method = "unbias", isPrec=TRUE, qstep=0.01)
```

Arguments

`obs`	Observed series for the reference (historical) period.
`mod`	Modelled series for the reference (historical) period.
`proj`	Modelled series for the projected period. If missing, the reference (historical) period is corrected.
`method`	Correction method, either `"unbias"`, `"scaling"`, `"quantmap"`
`isPrec`	A flag to indicate that variable is precipitation (only relevant for quantile mapping).
`qstep`	Probability step for quantile mapping (see `defaultCorrectionParams`).

Value

Returns a vector with corrected values.

Author(s)

Miquel De Cáceres Ainsa, Centre Tecnologic Forestal de Catalunya

References

De Caceres M, Martin-StPaul N, Turco M, Cabon A, Granda V (2018) Estimating daily meteorological data and downscaling climate models over landscapes. Environmental Modelling and Software 108: 186-196.

See Also

`correctionpoints`, `defaultCorrectionParams`

`defaultCorrectionParams` *Default correction parameters*

Description

Returns a list with the default parameterization for staistical correction.

Usage

```
defaultCorrectionParams()
```

Value

A list with the following items (default values in brackets):

- `methods`: A list with the correction method for each variable. Defaults are:
  - `MeanTemperature = "unbias"`
  - `MinTemperature = "quantmap"`
  - `MaxTemperature = "quantmap"`
  - `Precipitation = "quantmap"`
  - `MeanRelativeHumidity = "unbias"`
  - `Radiation = "unbias"`
  - `WindSpeed = "quantmap"`
- `fill_wind [= TRUE]`: A logical flag to fill wind speed values with uncorrected values when reference data is missing.
- `allow_saturated [= FALSE]`: A logical flag to indicate whether relative humidity values above saturation (>100%) are permitted (bias correction is performed on specific humidity).
- `wind_height [= 10]`: Wind measurement height (in m).
- `qstep [= 0.01]`: a numeric value between 0 and 1. Quantile mapping is fitted only for the quantiles defined by `quantile(0,1,probs=seq(0,1,by=qstep)`.

Author(s)

Miquel De Cáceres Ainsa, Centre de Ciència i Tecnologia Forestal de Catalunya

References

De Caceres M, Martin-StPaul N, Turco M, Cabon A, Granda V (2018) Estimating daily meteorological data and downscaling climate models over landscapes. Environmental Modelling and Software 108: 186-196.

See Also

`MeteorologyInterpolationData`

`defaultInterpolationParams` *Default interpolation parameters*

Description

Returns a list with the default parameterization for interpolation. Most parameter values are set according to Thornton et al. (1997).

Usage

```
defaultInterpolationParams()
```

Value

A list with the following items (default values in brackets):

- `initial_Rp [= 140000]`: Initial truncation radius.
- `iterations [= 3]`: Number of station density iterations.
- `alpha_MinTemperature [= 3.0]`: Gaussian shape parameter for minimum temperature.
- `alpha_MaxTemperature [= 3.0]`: Gaussian shape parameter for maximum temperature.
- `alpha_DewTemperature [= 3.0]`: Gaussian shape parameter for dew-point temperature.
- `alpha_PrecipitationEvent [= 5.0]`: Gaussian shape parameter for precipitation events.
- `alpha_PrecipitationAmount [= 5.0]`: Gaussian shape parameter for the regression of precipitation amounts.
- `alpha_Wind [= 3.0]`: Gaussian shape parameter for wind.
- `N_MinTemperature [= 30]`: Average number of stations with non-zero weights for minimum temperature.
- `N_MaxTemperature [= 30]`: Average number of stations with non-zero weights for maximum temperature.
- `N_DewTemperature [= 30]`: Average number of stations with non-zero weights for dew-point temperature.
- `N_PrecipitationEvent [= 5]`: Average number of stations with non-zero weights for precipitation events.
- `N_PrecipitationAmount [= 20]`: Average number of stations with non-zero weights for the regression of precipitation amounts.
- `N_Wind [= 2]`: Average number of stations with non-zero weights for wind.
- `St_Precipitation [= 5]`: Number of days for the temporal smoothing of precipitation.
- `St_TemperatureRange [= 15]`: Number of days for the temporal smoothing of temperature range.
- `pop_crit [= 0.50]`: Critical precipitation occurrence parameter.
- `f_max [= 0.6]`: Maximum value for precipitation regression extrapolations (0.6 equals to a maximum of 4 times extrapolation).
- `wind_height [= 10]`: Wind measurement height (in m).

Author(s)

Miquel De Cáceres Ainsa, Biodiversity and Landscape Ecology Laboratory, Centre Tecnologic Forestal de Catalunya

References

Thornton, P.E., Running, S.W., White, M. A., 1997. Generating surfaces of daily meteorological variables over large regions of complex terrain. J. Hydrol. 190, 214–251. doi:10.1016/S0022-1694(96)03128-9.

De Caceres M, Martin-StPaul N, Turco M, Cabon A, Granda V (2018) Estimating daily meteorological data and downscaling climate models over landscapes. Environmental Modelling and Software 108: 186-196.

See Also

MeteorologyInterpolationData

---

examplecorrectiondata *Example data set for statistical correction of RCM predictions*

Description

Example data set including the predictions of Regional Climate Model (CCLM4-8-17; driving global model CNRM-CERFACS-CNRM-CM5) for 3 model cells in a small area in Catalonia (NE Spain). Meteorological data covers an historical (reference) period (2000-2003) and a future (projection) period (2020-2023), the latter simulated under rcp4.5 scenario.

Usage

```
data("examplecorrectiondata")
```

Format

Formal class 'MeteorologyUncorrectedData-class'

Source

ESFG web site (http://esgf.llnl.gov/) that centralizes climate data from GCM and RCM uploaded in the frame of different international consortium, including the EURO-CORDEX regionalisation project.

Examples

```
data(examplecorrectiondata)
```

examplegridtopography *Example spatial grid topography*

Description

'SpatialGridTopography' object describing topographic features for a grid of 5 km x 5 km and cell size of 100 m in Catalonia (NE Spain).

Usage

```
data("examplegridtopography")
```

Format

Formal class 'SpatialGridTopography'

Source

'Institut Cartogràfic de Catalunya' (ICC)

Examples

```
data(examplegridtopography)
```

exampleinterpolationdata *Example data set for interpolation from weather stations*

Description

Example data set of spatial location, topography and daily meteorological records from 38 weather stations in Catalonia (NE Spain) corresponding to years 2000-2003.

Usage

```
data("exampleinterpolationdata")
```

Format

Formal class 'MeteorologyInterpolationData-class'

Source

'Servei Meteorològic de Catalunya' (SMC) and 'Agencia Española de Meteorología' (AEMET)

Examples

```
data(exampleinterpolationdata)
```

`extractNetCDF` *Extraction of climatic data from NetCDF files*

Description

This function reads a set of NetCDF files (one per variable) and extracts data for a set of NetCDF cells that are specified using a boundary box (in lon/lat format) or a set of (x,y) grid indices.

Usage

```
extractNetCDF(ncdf_files, bbox = NULL, offset = 0, cells = NULL, export = TRUE,
           exportDir = getwd(), exportFormat = "meteoland/txt", mpfilename = "MP.txt")
```

Arguments

`ncdf_files`	Character vector containing files to read
`bbox`	Boundary box (2 x 2 matrix) specifying the limit coordinates of a study area (in lon/lat format).
`offset`	A buffer to include NetCDF cells that are at a certain distance around the boundary box.
`cells`	A (n x 2) matrix specifying the x and y indices of n cells in a grid.
`export`	If `export = FALSE` the extracted data is stored in memory. Otherwise the result is written in the disk (using the format specified in `exportFormat`).
`exportFormat`	Export format for meteorological data (see `writemeteorologypoint`).
`exportDir`	Output directory for extracted meteorology.
`mpfilename`	The name of the file that will store the meta data describing all written files.

Details

Function `extractNetCDF` first identifies which cells in NetCDF data should be extracted according to `bbox` (or the cells are indicated by the user using `cells`), and the overall period (days). If neither `bbox` or `cells` is supplied, then all NetCDF cells will be processed. For each cell to be processed, the function loops over all files (which can describe different variables and time periods) and extracts the corresponding data. The function transforms units to the units used in `meteoland`. If specific humidity and mean temperature are available, the function calculates mean relative humidity.

Extracted meteorological data (a data frame with days in rows and meteorological variables in columns) can be stored in an object `SpatialPointsMeteorology-class` or it can be written in the disk (one file per cell). In the latter case, the output format can be chosen and the function also writes a supplementary file containing the meta data (i.e. the coordinates and filename of each file).

Humidity in climate model files is given as specific humidity. This is converted to relative humidity and the conversion may produce values above saturation (>100%) (see also `defaultCorrectionParams` for the same issue when performing bias correction).

Value

If `export = FALSE`, the function returns an object of class SpatialPointsMeteorology-class with the meteorological series for each cell (represented by a spatial point). Otherwise the function returns an object of class SpatialPointsDataFrame-class containing the meta data of the files written in the disk.

Author(s)

Miquel De Cáceres Ainsa, Centre Tecnologic Forestal de Catalunya

Nicolas Martin, INRA-Avignon

See Also

correctionpoints, writemeteorologypointfiles, SpatialPointsMeteorology-class

---

`extractpointdates` *Extracts meteorological data*

Description

Extracts meteorological data from an object.

Usage

```
extractpointdates(points, dates = NULL, verbose=FALSE)
extractgridpoints(griddata, points)
```

Arguments

`points`	For function `extractpointdates()`, an object of class SpatialPointsMeteorology. Alternatively, an object of class SpatialPointsDataFrame-class containing the meta data (columns `dir`, `filename` and possibly `format`) of meteorological files that will be sequentially read from the disk. For function `extractgridpoints()`, an object of class SpatialPoints.
`dates`	A vector of Date with a (subset) of dates to be extracted. If `NULL` all dates will be returned.
`griddata`	An object of class SpatialGridMeteorology-class or SpatialPixelsMeteorology-class with the meteorological data for a full grid or a subset of grid cells, respectively. Alternatively, a data.frame containing the meta data (columns `dir` and `filename`) of grid/pixels meteorological files that will be sequentially read from the disk.
`verbose`	Boolean flag to print process information.

Details

Function `extractpoints` is deprecated, because its functionality can be achieved using subsetting of spatial classes SpatialGridMeteorology and SpatialPixelsMeteorology.

Value

Function `extractpointdates()`, returns a `list` with the same length as `dates`. Each element of the list is an object of class `SpatialPointsDataFrame` with the meteorological data for all the points (in rows) and variables (in columns). Function `extractgridpoints()` returns an object of class `SpatialPointsMeteorology`.

Author(s)

Miquel De Cáceres Ainsa, Centre Tecnologic Forestal de Catalunya

---

`extractSpatialData` *Extracts spatial data from coordinates.*

Description

Extracts spatial data contained in a `SpatialGridDataFrame` or `SpatialPixelsDataFrame` corresponding to the coordinates of spatial classes.

Usage

```
extractSpatialData(x, y)
```

Arguments

`x`	An object of class `GridTopology`, `SpatialGrid` and `SpatialPoints`.
`y`	An object of class `SpatialGridDataFrame` or `SpatialPixelsDataFrame`

Value

A data frame with the contents of slot 'data' corresponding to the coordinates in `x`.

Author(s)

Miquel De Cáceres Ainsa, Biodiversity and Landscape Ecology Laboratory, Centre Tecnologic Forestal de Catalunya

See Also

`SpatialGridDataFrame`

Examples

```
# TO BE DONE
```

`interpolation.coverage` *Spatial and temporal coverage of interpolation data*

Description

Function `interpolation.coverage` calculates, for each meteorological variable, the number of stations with data per date or the number of dates with data per station in an object of class `MeteorologyInterpolationData-class`.

Usage

```
interpolation.coverage(object, type = "spatial", percent = FALSE)
```

Arguments

`object`	An object of class `MeteorologyInterpolationData-class`.
`type`	A string with the coverage summary to be produced (either "spatial" or "temporal").
`percent`	A boolean flag to indicate that percentages should be returned instead of counts.

Value

If `type = "spatial"` the function returns an object of class `SpatialPointsDataFrame` with the number (or percentage) of dates with data per station and meteorological variable. If `type = "temporal"` the function returns an object of class `data.frame` with the number (or percentage) of stations with data per day and meteorological variable.

Author(s)

Miquel De Cáceres Ainsa, Biodiversity and Landscape Ecology Laboratory, Centre Tecnologic Forestal de Catalunya

References

De Caceres M, Martin-StPaul N, Turco M, Cabon A, Granda V (2018) Estimating daily meteorological data and downscaling climate models over landscapes. Environmental Modelling and Software 108: 186-196.

See Also

`MeteorologyInterpolationData`

Examples

```
data(exampleinterpolationdata)

#Number of days with data per station
head(interpolation.coverage(exampleinterpolationdata))

#Number of stations with data per day
head(interpolation.coverage(exampleinterpolationdata, type = "temporal"))
```

---

interpolation.cv *Calibration and validation of interpolation procedures*

Description

Function `interpolation.calibration` determines optimal interpolation parameters 'N' and 'alpha' for a given meteorological variable. Optimization is done by minimizing mean absolute error (MAE) (Thornton et al. 1997). Function `interpolation.cv` calculates average mean absolute errors (MAE) for the prediction period of an object of class '`MeteorologyInterpolationData`'. Function `summary.interpolation.cv` returns a data.frame with cross-validation summaries and `plot.interpolation.cv` plots cross-validation results. In both calibration and validation procedures, predictions for each weather station are made using a leave-one-out procedure (i.e. after excluding the station from the predictive set).

Usage

```
interpolation.calibration(object, stations = NULL, variable="Tmin",
                     N_seq = seq(5,30, by=5), alpha_seq = seq(0.25,10, by=0.25),
                        verbose = FALSE)
interpolation.calibration.fmax(object, stations = NULL,
                               fmax_seq = seq(0.05,0.95, by=0.05),
                               verbose = FALSE)
interpolation.cv(object, stations = NULL, verbose = FALSE)
## S3 method for class 'interpolation.cv'
summary(object, ...)
## S3 method for class 'interpolation.cv'
plot(x, type = "stations", ...)
```

Arguments

`object`	In the case of function `interpolation.cv`, an object of class `MeteorologyInterpolationData-class`. In the case of function `summary`, an object of class `interpolation.cv`
`stations`	A numeric vector containing the indices of stations to be used to calculate mean absolute errors (MAE) in the calibration or cross-validation analysis. All the stations with data are included in the training set but predictive MAE are calculated for the 'stations' subset only.

variable	A string indicating the meteorological variable for which interpolation parameters 'N' and 'alpha' will be calibrated. Accepted values are 'Tmin' (for minimum temperature), 'Tmax' (for maximum temperature), 'Tdew' (for dew-point temperature), 'PrecEvent' (for precipitation events),'PrecAmount' (for regression of precipitation amounts),'Prec' (for precipitation with the same values for precipitation events and regression of precipitation amounts).
N_seq	Set of average number of points to be tested.
alpha_seq	Set of alpha values to be tested.
fmax_seq	Set of f_max values to be tested.
verbose	A logical flag to generate additional console output.
x	A S3 object of class interpolation.cv with cross-validation results.
type	A string of the plot type to be produced (either "stations" or "dates").
...	Additional parameters passed to summary and plot functions.

Value

Function interpolation.calibration returns an object of class 'interpolation.calibration' with the following items:

- MAE: A numeric matrix with the mean absolute error values (averaged across stations) for each combination of parameters 'N' and 'alpha'.
- minMAE: Minimum MAE value.
- N: Value of parameter 'N' corresponding to the minimum MAE.
- alpha: Value of parameter 'alpha' corresponding to the minimum MAE.
- Observed: A matrix with observed values.
- Predicted: A matrix with predicted values for the optimum parameter combination.

Function interpolation.cv returns a list of class 'interpolation.cv' with the following items:

- stations: A data frame with as many rows as weather stations and the following columns:
  - MinTemperature-Bias: Bias (in degrees), calculated over the prediction period, of minimum temperature estimations in weather stations.
  - MinTemperature-MAE: Mean absolute errors (in degrees), averaged over the prediction period, of minimum temperature estimations in weather stations.
  - MaxTemperature-Bias: Bias (in degrees), calculated over the prediction period, of maximum temperature estimations in weather stations.
  - MaxTemperature-MAE: Mean absolute errors (in degrees), averaged over the prediction period, of maximum temperature estimations in weather stations.
  - Precipitation-Total: Difference in the total precipitation of the studied period.
  - Precipitation-DPD: Difference in the proportion of days with precipitation.
  - Precipitation-Bias: Bias (in mm), calculated over the days with precipitation, of precipitation amount estimations in weather stations.
  - Precipitation-MAE: Mean absolute errors (in mm), averaged over the days with precipitation, of precipitation amount estimations in weather stations.

  - RelativeHumidity-Bias: Bias (in percent), calculated over the prediction period, of relative humidity estimations in weather stations.
  - RelativeHumidity-MAE: Mean absolute errors (in percent), averaged over the prediction period, of relative humidity estimations in weather stations.
  - Radiation-Bias: Bias (in MJ/m2), calculated over the prediction period, of incoming radiation estimations in weather stations.
  - Radiation-MAE: Mean absolute errors (in MJ/m2), averaged over the prediction period, of incoming radiation estimations in weather stations.
- dates: A data frame with as many rows as weather stations and the following columns:
  - MinTemperature-Bias: Daily bias (in degrees), averaged over the stations, of minimum temperature estimations.
  - MinTemperature-MAE: Daily mean absolute error (in degrees), averaged over the stations, of minimum temperature estimations.
  - MaxTemperature-Bias: Daily bias (in degrees), averaged over the stations, of maximum temperature estimations.
  - MaxTemperature-MAE: Daily mean absolute error (in degrees), averaged over the stations, of maximum temperature estimations.
  - Precipitation-Bias: Daily bias (in mm), averaged over the stations, of precipitation amount estimations.
  - Precipitation-MAE: Daily mean absolute error (in mm), averaged over the stations, of precipitation amount estimations.
  - RelativeHumidity-Bias: Daily bias (in percent), averaged over the stations, of relative humidity estimations.
  - RelativeHumidity-MAE: Daily mean absolute error (in percent), averaged over the stations, of relative humidity estimations.
  - Radiation-Bias: Daily bias (in MJ/m2), averaged over the stations, of incoming radiation estimations.
  - Radiation-MAE: Daily mean absolute errors (in MJ/m2), averaged over the stations, of incoming radiation estimations.
- MinTemperature: A data frame with predicted minimum temperature values.
- MinTemperatureError: A matrix with predicted minimum temperature errors.
- MaxTemperature: A data frame with predicted maximum temperature values.
- MaxTemperatureError: A matrix with predicted maximum temperature errors.
- Precipitation: A data frame with predicted precipitation values.
- PrecipitationError: A matrix with predicted precipitation errors.
- RelativeHumidity: A data frame with predicted relative humidity values.
- RelativeHumidityError: A matrix with predicted relative humidity errors.
- Radiation: A data frame with predicted radiation values.
- RadiationError: A matrix with predicted radiation errors.

Author(s)

Miquel De Cáceres Ainsa, Biodiversity and Landscape Ecology Laboratory, Centre Tecnologic Forestal de Catalunya

References

Thornton, P.E., Running, S.W., 1999. An improved algorithm for estimating incident daily solar radiation from measurements of temperature, humidity, and precipitation. Agric. For. Meteorol. 93, 211–228. doi:10.1016/S0168-1923(98)00126-9.

De Caceres M, Martin-StPaul N, Turco M, Cabon A, Granda V (2018) Estimating daily meteorological data and downscaling climate models over landscapes. Environmental Modelling and Software 108: 186-196.

See Also

MeteorologyInterpolationData

Examples

```
data(exampleinterpolationdata)

#Calibration procedure
precEv_cal = interpolation.calibration(exampleinterpolationdata, variable="PrecEvent",
                                       stations = 1:5,
                                     N_seq=c(5,10,15), alpha_seq=seq(0.25,1.0, by=0.25),
                                       verbose = TRUE)

precAm_cal = interpolation.calibration(exampleinterpolationdata, variable="PrecAmount",
                                       stations = 1:5,
                                     N_seq=c(5,10,15), alpha_seq=seq(0.25,1.0, by=0.25),
                                       verbose = TRUE)

#Set 'alpha' and 'N' parameters to values found in calibration
exampleinterpolationdata@params$N_PrecipitationEvent = precEv_cal$N
exampleinterpolationdata@params$alpha_PrecipitationEvent = precEv_cal$alpha

exampleinterpolationdata@params$N_PrecipitationAmount = precAm_cal$N
exampleinterpolationdata@params$alpha_PrecipitationAmount = precAm_cal$alpha

#Run cross validation
cv = interpolation.cv(exampleinterpolationdata, stations = 1:5, verbose = TRUE)

#Print cross validation summaries
summary(cv)

#Plot results
plot(cv)
```

---

interpolationpoints *Interpolates daily meteorology over a landscape*

Description

Interpolates meteorological data on spatial points/pixels/grid using an object of class MeteorologyInterpolationData-cla

Usage

```
interpolationpoints(object, points, dates = NULL, export = FALSE,
                    exportDir = getwd(), exportFormat = "meteoland/txt",
                    metadatafile = "MP.txt", verbose=TRUE)
interpolationpixels(object, pixels, dates, export = FALSE,
                  exportDir=getwd(), exportFormat = "netCDF",
                  metadatafile = "MG.txt", verbose=TRUE)
interpolationgrid(object, grid, dates, export = FALSE,
                  exportDir=getwd(), exportFormat = "netCDF",
                  metadatafile = "MG.txt", verbose=TRUE)
```

Arguments

object	An object of class MeteorologyInterpolationData-class.
points	An object of class SpatialPointsTopography-class.
pixels	An object of class SpatialPixelsTopography-class representing the target landscape.
grid	An object of class SpatialGridTopography-class representing the target landscape.
dates	An object of class Date. If this is NULL then all dates in object are processed.
export	If export = FALSE the result of interpolation is stored in memory. Otherwise the result is written in the disk (using the format specified in exportFormat).
exportDir	Output directory for interpolated meteorology data.
exportFormat	Export format for meteorological data (see writemeteorologypoint).
metadatafile	The name of the ascii text file that will store the meta data describing all written files.
verbose	Boolean flag to print process information.

Details

If CRS projection is different between object and points/pixels/grid, the function transforms the coordinates of points/pixels/grid to adapt them to the CRS of object.

Value

If export = FALSE, function interpolationpoints returns an object of SpatialPointsMeteorology-class. If export = TRUE, the function returns an object of class SpatialPointsDataFrame-class containing the meta data of the files written in the disk.

If export = FALSE, function interpolationpixels returns an object of SpatialPixelsMeteorology-class. If export = TRUE, the function writes the results in files and a data.frame with columns 'dir' and 'filename' is returned.

If export = FALSE, function interpolationgrid returns an object of SpatialGridMeteorology-class. If export = TRUE, the function writes the results in files and a data.frame with columns 'dir' and 'filename' is returned.

Author(s)

Miquel De Cáceres Ainsa, CTFC

References

Thornton, P.E., Running, S.W., White, M. A., 1997. Generating surfaces of daily meteorological variables over large regions of complex terrain. J. Hydrol. 190, 214–251. doi:10.1016/S0022-1694(96)03128-9.

De Caceres M, Martin-StPaul N, Turco M, Cabon A, Granda V (2018) Estimating daily meteorological data and downscaling climate models over landscapes. Environmental Modelling and Software 108: 186-196.

See Also

penman, SpatialPointsTopography-class, SpatialGridTopography, SpatialPixelsTopography, MeteorologyInterpolationData

Examples

```
data(examplegridtopography)
data(exampleinterpolationdata)

####### INTERPOLATION on particular POINTS

#Creates spatial topography points from the grid
p = 1:2
spt = as(examplegridtopography, "SpatialPointsTopography")[p]

#Interpolation of two points for the whole time period (2000-2003)
mp = interpolationpoints(exampleinterpolationdata, spt)

#Plot interpolated meteorological series
meteoplot(mp,1, ylab="Mean temperature")

####### INTERPOLATION on PIXELS
# Creates spatial topography pixels as a subset of grid pixels
# and select pixels at maximum distance of 2km from center
spt = as(examplegridtopography,"SpatialPointsTopography")
cc = spt@coords
center = 5160
d = sqrt((cc[,1]-cc[center,1])^2+(cc[,2]-cc[center,2])^2)
spxt = as(spt[which(d<2000)], "SpatialPixelsTopography")

# Interpolation of meteorology over pixels for two days
ml = interpolationpixels(exampleinterpolationdata, spxt,
                         as.Date(c("2001-02-03", "2001-06-03")))

#Plot PET corresponding to 2001-06-03
spplot(ml,2,"PET")

####### INTERPOLATION over a complete GRID
```

```
#Interpolation of meteorology over a grid for two days
ml = interpolationgrid(exampleinterpolationdata, examplegridtopography,
                       as.Date(c("2001-02-03", "2001-06-03")))
#Plot PET corresponding to 2001-06-03
spplot(ml,2,"PET")
```

`interpolation_lowlevel` *Low-level interpolation functions*

Description

Low-level functions to interpolate meteorology (one day) on a set of points.

Usage

```
interpolation_dewtemperature(Xp, Yp, Zp, X, Y, Z, T,
                             iniRp = 140000, alpha = 3.0, N = 30, iterations = 3)
interpolation_temperature(Xp, Yp, Zp, X, Y, Z, T,
                          iniRp = 140000, alpha = 3.0, N = 30, iterations = 3)
interpolation_precipitation(Xp, Yp, Zp, X, Y, Z, P, Psmooth,
                        iniRp = 140000, alpha_event = 6.25, alpha_amount = 6.25,
                      N_event = 20, N_amount = 20, iterations = 3, popcrit = 0.5,
                           fmax = 0.95)
interpolation_wind(Xp, Yp, WS, WD, X, Y,
                   iniRp = 140000, alpha = 2.0, N = 1, iterations = 3,
                   directionsAvailable = TRUE)
```

Arguments

`Xp, Yp, Zp`	Spatial coordinates and elevation (Zp; in m.a.s.l) of target points.
`X, Y, Z`	Spatial coordinates and elevation (Zp; in m.a.s.l) of reference locations (e.g. meteorological stations).
`T`	Temperature (e.g., minimum, maximum or dew temperature) at the reference locations (in degrees).
`P`	Precipitation at the reference locations (in mm).
`Psmooth`	Temporally-smoothed precipitation at the reference locations (in mm).
`WS, WD`	Wind speed (in m/s) and wind direction (in degrees from north clock-wise) at the reference locations.
`iniRp`	Initial truncation radius.
`iterations`	Number of station density iterations.
`alpha, alpha_amount, alpha_event`	Gaussian shape parameter.
`N, N_event, N_amount`	Average number of stations with non-zero weights.

popcrit — Critical precipitation occurrence parameter.

fmax — Maximum value for precipitation regression extrapolations (0.6 equals to a maximum of 4 times extrapolation).

directionsAvailable — A flag to indicate that wind directions are available (i.e. non-missing) at the reference locations.

Value

All functions return a vector with interpolated values for the target points.

Author(s)

Miquel De Cáceres Ainsa, Centre Tecnologic Forestal de Catalunya

References

Thornton, P.E., Running, S.W., White, M. A., 1997. Generating surfaces of daily meteorological variables over large regions of complex terrain. J. Hydrol. 190, 214–251. doi:10.1016/S0022-1694(96)03128-9.

De Caceres M, Martin-StPaul N, Turco M, Cabon A, Granda V (2018) Estimating daily meteorological data and downscaling climate models over landscapes. Environmental Modelling and Software 108: 186-196.

See Also

defaultInterpolationParams

Examples

```
data("exampleinterpolationdata")
mxt100 = exampleinterpolationdata@MaxTemperature[,100]
Psmooth100 = exampleinterpolationdata@SmoothedPrecipitation[,100]
P100 = exampleinterpolationdata@Precipitation[,100]
mismxt = is.na(mxt100)
misP = is.na(P100)
Z = exampleinterpolationdata@elevation
X = exampleinterpolationdata@coords[,1]
Y = exampleinterpolationdata@coords[,2]
Zpv = seq(0,1000, by=100)
xp = 360000
yp = 4640000
xpv = rep(xp, 11)
ypv = rep(yp, 11)

interpolation_temperature(xpv, ypv, Zpv,
                          X[!mismxt], Y[!mismxt], Z[!mismxt],
                          mxt100[!mismxt])
interpolation_precipitation(xpv, ypv, Zpv,
                            X[!misP], Y[!misP], Z[!misP],
                            P100[!misP], Psmooth100[!misP])
```

`meteocomplete` *Complete daily meteorological variables*

Description

Fills missing values of relative humidity, radiation and potential evapotranspiration from a data frame with daily values of minimum/maximum/mean temperature and precipitation.

Usage

```
meteocomplete(x, latitude, elevation, slope, aspect)
```

Arguments

`x`	A data frame with dates as row names and columns named 'MeanTemperature', 'MaxTemperature', 'MinTemperature' and 'Precipitation'
`latitude`	Latitude in degrees North.
`elevation`	Elevation in m.a.s.l.
`slope`	Slope in degrees.
`aspect`	Aspect in degrees from North.

Details

The function fills values for humidity, radiation and PET only if they are missing in the input data frame. If a column 'SpecificHumidity' is present in the input data, relative humidity is calculated from it. Otherwise, relative humidity is calculated assuming that dew point temperature equals the minimum temperature. Potential solar radiation is calculated from latitude, slope and aspect. Incoming solar radiation is then corrected following Thornton & Running (1999) and potential evapotranspiration following Penman (1948).

Value

A data frame copied from `x` but with filled values for variables:

- `MeanRelativeHumidity`: Mean daily relative humidity (in percent).
- `MinRelativeHumidity`: Minimum daily relative humidity (in percent).
- `MaxRelativeHumidity`: Maximum daily relative humidity (in percent).
- `Radiation`: Incoming solar radiation (MJ/m2).
- `PET`: Potential evapotranspiration (in mm of water).

Author(s)

Miquel De Cáceres Ainsa, CTFC

References

Thornton, P.E., Running, S.W., 1999. An improved algorithm for estimating incident daily solar radiation from measurements of temperature, humidity, and precipitation. Agric. For. Meteorol. 93, 211-228.

Penman, H. L. 1948. Natural evaporation from open water, bare soil and grass. Proceedings of the Royal Society of London. Series A. Mathematical and Physical Sciences, 193, 120-145.

See Also

penman, radiation_solarRadiation

`meteoplot` *Plots point meteorological series*

Description

Simple plotting of a meteorological series for a given point.

Usage

```
meteoplot(object, index=1, var="MeanTemperature",
          fun=NULL, freq=NULL, dates = NULL, months = NULL, add = FALSE,...)
```

Arguments

`object`	A data frame with daily meteorological data (in this case `index` is not used) or an object of class `SpatialPointsMeteorology`. Alternatively, an object of class `SpatialPointsDataFrame` containing the meta data (columns `dir`, `filename` and possibly `format`) of meteorological files.
`index`	An integer to indicate the point in the `SpatialPointsMeteorology` object (or the `SpatialPointsDataFrame` object).
`var`	The meteorological variable to be plotted.
`fun`	The name of a function to be calculated for summaries (only valid if `freq` is specified).
`freq`	A string giving an interval specification for summaries (e.g., `"week"`, `"month"`, `"quarter"` or `"year"`).
`dates`	An object of class `Date` to define the period to be plotted. If `dates = NULL` then all dates in `object` are processed.
`months`	A numeric vector to indicate the subset of months for which plotting is desired (e.g. `c(7,8)` for July and August). When combined with `fun` and `freq`, this parameter allows plotting summaries for particular seasons. For example `fun = "sum" freq = "years"` and `months = 6:8` leads to plotting the sum over summer months of each year.
`add`	A flag to indicate wether drawing should be done on the current plot (using function `lines`).
`...`	Additional parameters for functions `plot` or `lines`.

Details

Daily precipitation is plotted using bars (i.e. `type = "h"` when calling `plot`). Otherwise the function draws lines (i.e. `type = "l"` when calling `plot`). If `object` is of class `SpatialPointsDataFrame-class` then the function reads the meteorological data to be plotted from the disk.

Author(s)

Miquel De Cáceres Ainsa, Biodiversity and Landscape Ecology Laboratory, Centre Tecnologic Forestal de Catalunya

See Also

`summarypoints`

Examples

```
data(examplegridtopography)
data(exampleinterpolationdata)

#Creates spatial topography points from the grid
p = 1:2
spt = as(examplegridtopography, "SpatialPointsTopography")[p]

#Interpolation of two points for the whole time period (2000-2003)
mp = interpolationpoints(exampleinterpolationdata, spt)

#Plot interpolated meteorological series
meteoplot(mp,1, ylab="Daily mean temperature")

meteoplot(mp,1, ylab="Monthly mean temperature", fun=mean, freq="months")
```

MeteorologyInterpolationData

Creates an object of class 'MeteorologyInterpolationData'

Description

Initializes an object for meteorology interpolation over landscapes using weather station data and the methods described in Thornton et al. (1997) and Thornton & Running (1999).

Usage

```
MeteorologyInterpolationData(points, elevation = NULL, slope = NULL, aspect = NULL,
                             MinTemperature = NULL, MaxTemperature = NULL,
                             Precipitation = NULL, RelativeHumidity = NULL,
                             Radiation = NULL, WindSpeed = NULL,
                             WindDirection = NULL, WindFields = NULL,
                             params = defaultInterpolationParams())
```

Arguments

points	An object of class SpatialPointsMeteorology, an object of SpatialPointsTopography or an object of class SpatialPoints (see 'Details").
elevation	A numeric vector with elevation values of weather stations (in meters).
slope	A numeric vector with slope values of weather stations (in degrees). Needed for cross-validation of interpolation routines.
aspect	A numeric vector with aspect values of weather stations (in degrees from North). Needed for cross-validation of interpolation routines.
MinTemperature	A matrix with minimum temperature recordings (in degrees Celsius) for all weather stations (in rows) and all days (in columns).
MaxTemperature	A matrix with maximum temperature recordings (in degrees Celsius) for all weather stations (in rows) and all days (in columns).
Precipitation	A matrix with precipitation recordings (in mm of water) for all weather stations (in rows) and all days (in columns).
RelativeHumidity	A matrix with (mean) relative humidity recordings (in percent) for all weather stations (in rows) and all days (in columns).
Radiation	A matrix with relative radiation recordings (in MJ/m2) for all weather stations (in rows) and all days (in columns). Needed for cross-validation only.
WindSpeed	A matrix with wind speed recordings (in m/s) for all weather stations (in rows) and all days (in columns).
WindDirection	A matrix with wind direction recordings (in degrees from North) for all weather stations (in rows) and all days (in columns).
WindFields	Object of class "list". See function readWindNinjaWindFields.
params	A list containing interpolation parameters.

Details

There are three ways of building an object of MeteorologyInterpolationData:

1. The first way is using an object of SpatialPointsMeteorology containing both the coordinates and meteorological series of stations. In this case elevation has to be provided, but aspect and slope may be ommitted. Parameters MinTemperature to WindDirection can be left as NULL.
2. The second way is using an object of class of SpatialPointsTopography containing the coordinates of stations and topographic variables. In this case parameters MinTemperature, MaxTemperature and Precipitation will need to be supplied, each being a matrix with weather stations in rows and days in columns, but RelativeHumidity to WindDirection may be left as NULL.
3. The third way is using an object of SpatialPoints containing the coordinates of stations only. In this case elevation has to be provided, but aspect and slope may be ommitted. As in the second case, parameters MinTemperature, MaxTemperature and Precipitation will need to be supplied, each being a matrix with weather stations in rows and days in columns, but RelativeHumidity to WindDirection may be left as NULL.

Value

An object of class MeteorologyInterpolationData.

Author(s)

Miquel De Cáceres Ainsa, Biodiversity and Landscape Ecology Laboratory, Centre Tecnologic Forestal de Catalunya

References

Thornton, P.E., Running, S.W., 1999. An improved algorithm for estimating incident daily solar radiation from measurements of temperature, humidity, and precipitation. Agric. For. Meteorol. 93, 211–228. doi:10.1016/S0168-1923(98)00126-9.

Thornton, P.E., Running, S.W., White, M. a., 1997. Generating surfaces of daily meteorological variables over large regions of complex terrain. J. Hydrol. 190, 214–251. doi:10.1016/S0022-1694(96)03128-9.

See Also

MeteorologyInterpolationData, defaultInterpolationParams.

Examples

```
## TO BE DONE ##
```

MeteorologyInterpolationData-class

Class "MeteorologyInterpolationData"

Description

An S4 class to interpolate meteorology over a landscape.

Objects from the Class

Objects can be created by calls of the form `new("MeteorologyInterpolationData", ...)`, or by calls to the function MeteorologyInterpolationData.

Slots

dates: Object of class "Date" describing the time period for which meteorological estimates are possible.

bbox: Object of class "matrix" with the boundary box that sets meteorological estimation boundaries.

proj4string: Object of class "CRS" with the projection string of station spatial coordinates.

`coords`: Object of class `"matrix"` containing the coordinates of weather stations (each row is a point).

`elevation`: A numeric vector with elevation values of weather stations (in meters).

`slope`: A numeric vector with slope values of weather stations (in degrees). Needed for cross-validation only.

`aspect`: A numeric vector with aspect values of weather stations (in degrees from North). Needed for cross-validation only.

`MinTemperature`: Object of class `"data.frame"` with minimum temperature recordings (in degrees Celsius) for all weather stations (in rows) and all days (in columns).

`MaxTemperature`: Object of class `"data.frame"` with maximum temperature recordings (in degrees Celsius) for all weather stations (in rows) and all days (in columns).

`SmoothedTemperatureRange`: Object of class `"matrix"` with temporally smoothed temperature range recordings (in degrees Celsius) for all weather stations (in rows) and all days (in columns).

`Precipitation`: Object of class `"matrix"` with precipitation recordings (in mm of water) for all weather stations (in rows) and all days (in columns).

`SmoothedPrecipitation`: Object of class `"matrix"` with temporally smoothed precipitation recordings (in mm of water) for all weather stations (in rows) and all days (in columns).

`RelativeHumidity`: Object of class `matrix` with relative humidity recordings (in percent) for all weather stations (in rows) and all days (in columns).

`Radiation`: Object of class `matrix` with relative radiation recordings (in MJ/m2) for all weather stations (in rows) and all days (in columns). Needed for cross-validation only.

`WindSpeed`: Object of class `"matrix"` with wind speed recordings (in m/s) for all weather stations (in rows) and all days (in columns).

`WindDirection`: Object of class `"matrix"` with wind direction recordings (in degrees from North) for all weather stations (in rows) and all days (in columns).

`WindFields`: Object of class `"list"`. See function `readWindNinjaWindFields`.

`WFIndex`: Object of class `"matrix"` with the closest windfield index for each stations in each day.

`WFFactor`: Object of class `"matrix"` with the multiplication factor for the wind speed of each stations in each day.

`params`: A `"list"` containing interpolation parameters.

Extends

Class `"MeteorologyProcedureData"`, directly. Class `"Spatial"`, by class "MeteorologyProcedureData", distance 2

Methods

subsample `signature(object = "MeteorologyInterpolationData")`: Generates a `MeteorologyInterpolationData` object for a smaller area and a subset of dates.

Author(s)

Miquel De Cáceres Ainsa, Centre Tecnologic Forestal de Catalunya

See Also

MeteorologyInterpolationData, MeteorologyProcedureData-class, subsample

Examples

```
#Structure of the S4 object
showClass("MeteorologyInterpolationData")
```

MeteorologyProcedureData-class
Class "MeteorologyProcedureData"

Description

A virtual class for estimating meteorology over landscapes

Objects from the Class

A virtual Class: No objects may be created from it.

Slots

dates: Object of class "Date" describing the time period for which meteorological estimates are possible.

bbox: Object of class "matrix" with the boundary box that sets meteorological estimation boundaries.

proj4string: Object of class "CRS" with the projection string of accepted coordinates.

Methods

subsample signature(object = "MeteorologyProcedureData"): Generates MeteorologyProcedureData objects for a smaller area and a subset of dates.

Extends

Class "Spatial", directly.

Author(s)

Miquel De Cáceres Ainsa, Biodiversity and Landscape Ecology Laboratory, Centre Tecnologic Forestal de Catalunya

See Also

MeteorologyInterpolationData-class, MeteorologyUncorrectedData-class

Examples

```
showClass("MeteorologyProcedureData")
```

`MeteorologyUncorrectedData` *Creates an object of class 'MeteorologyUncorrectedData'*

Description

Initializes an object for statistical correction of meteorological data over landscapes.

Usage

```
MeteorologyUncorrectedData(points, reference_data, projection_data, dates,
                           params = defaultCorrectionParams())
```

Arguments

`points`	An object of class `SpatialPoints`.
`reference_data`	Reference (historic) meteorological data used to calibrate correction factors when compared with observations. A vector of data frames (one per point) or a single data frame containing the meta data (columns `dir` and `filename`) of meteorological files that will be read from the disk.
`projection_data`	Projected meteorological data to be corrected. A vector of data frames (one per point) or a single data frame containing the meta data (columns `dir` and `filename`) of meteorological files that will be read from the disk.
`dates`	Object of class `"Date"` describing the time period for which meteorological correction is possible (corresponding to `projection_data`).
`params`	A `"list"` containing correction parameters.

Details

See correction details in vignettes or in `correctionpoints`.

Value

An object of class `MeteorologyUncorrectedData`.

Author(s)

Miquel De Cáceres Ainsa, Biodiversity and Landscape Ecology Laboratory, Centre Tecnologic Forestal de Catalunya

See Also

`MeteorologyUncorrectedData`, `examplecorrectiondata`, `defaultCorrectionParams`.

MeteorologyUncorrectedData-class
Class "MeteorologyUncorrectedData"

Description

An S4 class to conduct statistical correction of meteorology over a landscape.

Objects from the Class

Objects can be created by calls of the form `new("MeteorologyUncorrectedData", ...)`, or by calls to the function `MeteorologyUncorrectedData`.

Slots

`dates`: Object of class `"Date"` describing the time period for which meteorological estimates are possible.

`bbox`: Object of class `"matrix"` with the boundary box that sets meteorological estimation boundaries.

`proj4string`: Object of class `"CRS"` with the projection string of station spatial coordinates.

`coords`: Object of class `"matrix"` containing the coordinates of weather stations (each row is a point).

`reference_data`: Reference (historic) meteorological data used to calibrate correction factors when compared with observations. A vector of data frames (one per point) or a single data frame containing the meta data (columns `dir` and `filename`) of meteorological files that will be read from the disk.

`projection_data`: Projection meteorological data to be corrected. A vector of data frames (one per point) or a single data frame containing the meta data (columns `dir` and `filename`) of meteorological files that will be read from the disk.

`params`: A `"list"` containing correction parameters.

Extends

Class `"MeteorologyProcedureData"`, directly. Class `"Spatial"`, by class "MeteorologyProcedureData", distance 2

Methods

subsample `signature(object = "MeteorologyUncorrectedData")`: Generates a `MeteorologyUncorrectedData` object for a smaller area and a subset of dates.

Author(s)

Miquel De Cáceres Ainsa, Biodiversity and Landscape Ecology Laboratory, Centre Tecnologic Forestal de Catalunya

See Also

MeteorologyUncorrectedData, MeteorologyProcedureData-class, examplecorrectiondata, subsample

Examples

```
#Structure of the S4 object
showClass("MeteorologyUncorrectedData")
```

---

penman *Potential evapotranspiration*

Description

Functions to calculate potential evapotranspiration using Penman or Penman-Monteith.

Usage

```
penman(latrad, elevation, slorad, asprad, J,
            Tmin, Tmax, RHmin, RHmax, R_s,  u,
            z=2.0, z0 = 0.001, alpha = 0.08, windfun="1956")
penmanmonteith(rc, elevation, Tmin, Tmax, RHmin, RHmax,
               Rn, u = NA)
```

Arguments

latrad	Latitude in radians.
elevation	Elevation (in m).
slorad	Slope (in radians).
asprad	Aspect (in radians from North).
J	Julian day, number of days since January 1, 4713 BCE at noon UTC.
Tmax	Maximum temperature (degrees Celsius).
Tmin	Minimum temperature (degrees Celsius).
RHmin	Minimum relative humidity (percent).
RHmax	Maximum relative humidity (percent).
R_s	Solar radiation (MJ/m2).
u	With wind speed (m/s).
z	Wind measuring height (m).
z0	Roughness height (m).
alpha	Albedo.
windfun	Wind speed function version, either "1948" or "1956".
rc	Canopy vapour flux (stomatal) resistance (s·m-1).
Rn	Daily net radiation (MJ·m-2·day-1).

Details

The code was adapted from package 'Evapotranspiration', which follows McMahon et al. (2013). If wind speed is not available, an alternative formulation for potential evapotranspiration is used as an approximation (Valiantzas 2006)

Value

Potential evapotranspiration (in mm of water).

Author(s)

Miquel De Cáceres Ainsa, Biodiversity and Landscape Ecology Laboratory, Centre Tecnologic Forestal de Catalunya

References

Penman, H. L. 1948. Natural evaporation from open water, bare soil and grass. Proceedings of the Royal Society of London. Series A. Mathematical and Physical Sciences, 193, 120-145.

Penman, H. L. 1956. Evaporation: An introductory survey. Netherlands Journal of Agricultural Science, 4, 9-29.

McMahon, T.A., Peel, M.C., Lowe, L., Srikanthan, R., McVicar, T.R. 2013. Estimating actual, potential, reference crop and pan evaporation using standard meteorological data: a pragmatic synthesis. Hydrology \& Earth System Sciences 17, 1331–1363. doi:10.5194/hess-17-1331-2013.

See Also

`interpolationpoints`

---

precipitationconcentration

Daily precipitation concentration

Description

Calculates daily precipitation concentration (Martin-Vide et al. 2004)

Usage

```
precipitationconcentration(p)
```

Arguments

p	A numeric vector with daily precipitation values.

Value

An value between 0 (equal distribution of rainfall) and 1 (one day concentrates all rainfall).

Author(s)

Miquel De Cáceres Ainsa, Centre Tecnologic Forestal de Catalunya

References

Martin-Vide J (2004) Spatial distribution of a daily precipitation concentration index in peninsular Spain. International Journal of Climatology 24, 959–971. doi:10.1002/joc.1030.

---

radiation *Solar radiation utility functions*

Description

Set of functions used in the calculation of incoming solar radiation and net radiation.

Usage

```
radiation_dateStringToJulianDays(dateStrings)
radiation_daylength(latrad, slorad, asprad, delta)
radiation_daylengthseconds(latrad, slorad, asprad, delta)
radiation_directDiffuseInstant(solarConstant, latrad, slorad, asprad, delta,
                               hrad, R_s, clearday)
radiation_directDiffuseDay(solarConstant, latrad, slorad, asprad, delta,
                           R_s, clearday, nsteps = 24)
radiation_potentialRadiation(solarConstant, latrad, slorad, asprad, delta)
radiation_julianDay(year, month, day)
radiation_skyLongwaveRadiation(Tair, vpa, c)
radiation_outgoingLongwaveRadiation(solarConstant, latrad, elevation,  slorad,
                                    asprad, delta, vpa, tmin, tmax, R_s)
radiation_netRadiation(solarConstant, latrad, elevation, slorad, asprad,  delta,
                    vpa, tmin, tmax, R_s, alpha = 0.08)
radiation_solarConstant(J)
radiation_solarDeclination(J)
radiation_solarElevation(latrad, delta, hrad)
radiation_solarRadiation(solarConstant, latrad, elevation, slorad, asprad, delta,
                         diffTemp, diffTempMonth, vpa, precipitation)
radiation_sunRiseSet(latrad, slorad, asprad, delta)
```

Arguments

dateStrings	A character vector with dates in format "YYYY-MM-DD".
latrad	Latitude (in radians North).
slorad	Slope (in radians).
asprad	Aspect (in radians from North).
delta	Solar declination (in radians).

solarConstant	Solar constant (in kW·m-2).
hrad	Solar hour (in radians).
R_s	Daily incident solar radiation (MJ·m-2).
clearday	Boolean flag to indicate a clearsky day (vs. overcast).
nsteps	Number of daily substeps.
J	Julian day (integer), number of days since January 1, 4713 BCE at noon UTC.
year, month, day	Year, month and day as integers.
alpha	Surface albedo (from 0 to 1).
Tair	Air temperature (in degrees Celsius).
vpa	Average daily vapor pressure (kPa).
c	Proportion of sky covered by clouds [0-1].
tmin, tmax	Minimum and maximum daily temperature (ºC).
elevation	Elevation above sea level (in m).
precipitation	Precipitation (in mm).
diffTemp	Difference between maximum and minimum temperature (ºC).
diffTempMonth	Difference between maximum and minimum temperature, averaged over 30 days (ºC).

Value

Values returned for each function are:

- radiation_dateStringToJulianDays: A vector of Julian days (i.e. number of days since January 1, 4713 BCE at noon UTC).
- radiation_daylength: Day length (in hours).
- radiation_daylengthseconds: Day length (in seconds).
- radiation_directDiffuseInstant: A vector with instantaneous direct and diffusive radiation rates (for both SWR and PAR).
- radiation_directDiffuseDay: A data frame with instantaneous direct and diffusive radiation rates (for both SWR and PAR) for each subdaily time step.
- radiation_potentialRadiation: Daily (potential) solar radiation (in MJ·m-2).
- radiation_julianDay: Number of days since January 1, 4713 BCE at noon UTC.
- radiation_skyLongwaveRadiation: Instantaneous incoming (sky) longwave radiation (W·m-2).
- radiation_outgoingLongwaveRadiation: Daily outgoing longwave radiation (MJ·m-2·day-1).
- radiation_netRadiation: Daily net solar radiation (MJ·m-2·day-1).
- radiation_solarConstant: Solar constant (in kW·m-2).
- radiation_solarDeclination: Solar declination (in radians).
- radiation_solarElevation: Angle of elevation of the sun with respect to the horizon (in radians).
- radiation_solarRadiation: Daily incident solar radiation (MJ·m-2·day-1).
- radiation_sunRiseSet: Sunrise and sunset hours in hour angle (radians).

Note

Code for `radiation_julianDay()`, `radiation_solarConstant()` and `radiation_solarDeclination()` was translated to C++ from R code in package 'insol' (by J. G. Corripio).

Author(s)

Miquel De Cáceres Ainsa, Centre Tecnologic Forestal de Catalunya

References

Danby, J. M. Eqn. 6.16.4 in Fundamentals of Celestial Mechanics, 2nd ed. Richmond, VA: Willmann-Bell, p. 207, 1988.

Garnier, B.J., Ohmura, A., 1968. A method of calculating the direct shortwave radiation income of slopes. J. Appl. Meteorol. 7: 796-800

McMahon, T. A., M. C. Peel, L. Lowe, R. Srikanthan, and T. R. McVicar. 2013. Estimating actual, potential, reference crop and pan evaporation using standard meteorological data: a pragmatic synthesis. Hydrology & Earth System Sciences 17:1331–1363. See also: http://www.fao.org/docrep/x0490e/x0490e06.htm.

Reda, I. and Andreas, A. 2003. Solar Position Algorithm for Solar Radiation Applications. 55 pp.; NREL Report No. TP-560-34302, Revised January 2008. http://www.nrel.gov/docs/fy08osti/34302.pdf

Spitters, C.J.T., Toussaint, H.A.J.M. and Goudriaan, J. (1986). Separating the diffuse and direct components of global radiation and its implications for modeling canopy photosynthesis. I. Components of incoming radiation. Agricultural and Forest Meteorology, 38, 231–242.

See Also

`interpolationpoints`

---

readmeteorologygrid *Reads grid meteorology from the disk*

Description

Functions to read grid meteorological data from the disk.

Usage

```
readmeteorologygrid(file, format = "netCDF")
readmeteorologygridfiles(files, format = "netCDF")
readmeteorologygridcells(files, cellIndices, format = "netCDF")
readmeteorologypixels(file, format = "netCDF")
readmeteorologypixelsfiles(files, format = "netCDF")
```

Arguments

file	A string of the file name to be read.
format	Format of meteorological data. The only accepted format is "netCDF".
files	Either a vector of filename strings or a data frame with two columns: 'dir' and 'filename.
cellIndices	An integer vector with grid cell indices indicating the cells for which meteorological data has to be read.

Details

Function readmeteorologygrid reads a file containing the meteorology over a grid for a single day. Function readmeteorologygridfiles reads several files, each containing the meteorology over the same grid for a different day. Function readmeteorologygridcell also reads several grid meteorology files, but it keeps the meteorology of a set of cells only. Function readmeteorologypixels reads a file containing the meteorology over a grid for a single day and filters those pixels with missing data. Function readmeteorologypixelsfiles reads several files, each containing the meteorology over the same grid for a different day, and filters those pixels with missing data.

Value

Function readmeteorologygrid returns an object SpatialGridDataFrame-class where the data frame has grid cells as rows and meteorological variables as columns. Function readmeteorologygridfiles returns an object SpatialGridMeteorology-class and readmeteorologygridcells returns an object SpatialPointsMeteorology-class. Function readmeteorologypixels returns an object SpatialPixelsDataFrame-class where the data frame has pixels as rows and meteorological variables as columns. Function readmeteorologypixelsfiles returns an object SpatialPixelsMeteorology-class.

Author(s)

Miquel De Cáceres Ainsa, Centre Tecnologic Forestal de Catalunya

See Also

writemeteorologygrid, writemeteorologypixels, SpatialGridMeteorology-class, SpatialPixelsMeteorology-cl

---

readmeteorologypoint *Reads point meteorology from the disk*

Description

Functions to read point meteorological data from ascii files in different formats.

Usage

```
readmeteorologypoint(file, dates = NULL, format = "meteoland/txt", sep="\t")
readmeteorologypointfiles(points, files=NULL, dates = NULL, format="meteoland/txt",
                          sep="\t")
```

Arguments

file	A string of the file to be read.
points	An object of class SpatialPoints-class (in this case files cannot be NULL) or object of class SpatialPointsDataFrame-class with two data columns: 'dir' and 'filename' (and possibly 'format').
files	A vector of strings to be read (when points is of class SpatialPoints-class). Length and order must match points.
dates	Object of class "Date" describing a subset of dates to be extracted from meteorological series. If NULL the whole period read from files is kept.
format	Format of meteorological data. Current accepted formats are "meteoland/txt", "meteoland/rds", "castanea/txt" and "castanea/rds".
sep	The field separator character for ascii text files (see read.table).

Details

In readmeteorologypointfiles the value of format is used as default but can be overloaded if points includes a column 'format'.

Author(s)

Miquel De Cáceres Ainsa, Centre Tecnologic Forestal de Catalunya

Nicolas Martin, INRA-Avignon

See Also

writemeteorologypoint, read.table, SpatialPointsMeteorology-class

---

readWindNinjaWindFields

Reads WindNinja results

Description

Reads the wind fields generated by 'WindNinja' (http://www.firelab.org/project/windninja) for combinations of domain-level wind speed and wind direction classes.

Usage

```
readWindNinjaWindFields(filebase, resolution = "100m",
                        directionClasses = c(0, 45, 90, 135, 180, 225, 270, 315),
                    speedClasses = c(5, 15, 25), proj4string = CRS(as.character(NA)))
```

Arguments

filebase	A string to indicate the template for accessing WindNinja files. Resolution, wind directions and wind speed class values are appended to this string to obtain the filename to read.
resolution	Resolution string.
directionClasses	A vector of wind speed directions (in degrees).
speedClasses	A vector of wind class values (in m/s).
proj4string	Object of class "CRS" with the projection string of wind field rasters.

Value

A list with the following items:

- directionClasses: The vector of wind direction classes.
- speedClasses: The vector of wind speed classes.
- indexTable: A numeric matrix indicating the raster index of each combination of domain-level wind directions and wind speed classes.
- windSpeed: An object of class SpatialGridDataFrame containing wind speed rasters (in m/s) for each combination of domain-level wind direction and wind speed.
- windDirection: An object of class SpatialGridDataFrame containing wind direction rasters (in degrees from North) for each combination of domain-level wind direction and wind speed.

Note

WindNinja should be run with m/s as wind speed units and for all the combinations of domain-level wind speed and wind direction required.

Author(s)

Miquel De Cáceres Ainsa, Biodiversity and Landscape Ecology Laboratory, Centre Tecnologic Forestal de Catalunya

References

Forthofer, J.M., Butler, B.W., Wagenbrenner, N.S., 2014. A comparison of three approaches for simulating fine-scale surface winds in support of wildland fire management. Part I. Model formulation and comparison against measurements. Int. J. Wildl. Fire 23, 969–981.

See Also

MeteorologyInterpolationData

Examples

```
## TO BE DONE ##
```

`reshapeworldmet` *Reshapes data from 'worldmet' or 'weathercan'*

Description

Reshapes weather station data acquired using the 'worldmet' or 'weathercan' R packages into formats useful for meteoland

Usage

```
reshapeworldmet(hourly_data, output = "SpatialPointsMeteorology", complete = TRUE,
                verbose = TRUE)
reshapeweathercan(hourly_data, daily_data = NULL, output = "SpatialPointsMeteorology",
                  complete = TRUE, verbose = TRUE)
```

Arguments

`hourly_data`	Hourly weather data. In the case of `reshapeworldmet`, a tibble or data frame returned by function 'importNOAA'. In the case of `reshapeweathercan`, a tibble or data frame returned by function 'weather_dl' with 'interval="hour"'.
`daily_data`	Daily weather data (only for `reshapeweathercan`), a tibble or data frame returned by function 'weather_dl' with 'interval="day"'.
`output`	Kind of output desired. Either 'SpatialPointsTopography', 'SpatialPointsMeteorology' or 'MeteorologyInterpolationData'.
`complete`	A flag to indicate that missing variables should be completed using function `meteocomplete`
`verbose`	A flag to show information of the reshape process in the console output.

Details

Note that to have precipitation included in downloads from 'worldmet' you should set 'precip = TRUE' when calling function 'importNOAA'. In the case of weathercan, precipitation is only provided for daily data (i.e. setting 'interval="day"' when calling 'weather_dl'), whereas wind speed and relative humidity are only availave for hourly data (i.e., setting 'interval="hour"' when calling 'weather_dl'). Hence, in `meteoland` we recommend downloading both daily and hourly data and then calling function `reshapeweathercan` to merge the two sources.

Value

An object of the class indicated in `output`.

Author(s)

Miquel De Cáceres Ainsa, CTFC

See Also

meteocomplete

SMC download *Download data from SMC*

Description

Download data from the Catalan automatic weather station network (XEMA from Servei Meteorològic de Catalunya)

Usage

```
downloadSMCvarmetadata(api, type = "current")
downloadSMCstationlist(api, date = NULL)
downloadSMCcurrentday(api, daily_meteoland = TRUE, variable_code=NULL,
                      station_id=NULL, date = Sys.Date(), verbose=TRUE)
downloadSMChistorical(api, dates, station_id=NULL, variable_code=NULL, export = FALSE,
                      exportDir = getwd(), exportFormat = "meteoland/txt",
                      metadatafile = "MP.txt", verbose=TRUE)
```

Arguments

api	String with the SMC API key (the procedure to apply for an api key is explained in https://apidocs.meteocat.gencat.cat/).
type	Either 'current' or 'historical' for current day variable metadata or historical variable metadata, respectively.
daily_meteoland	When daily_meteoland = TRUE all the usual variables in meteoland are downloaded and aggregated at the daily scale. When daily_meteoland = FALSE, the user can choose the desired variables using variable_code.
variable_code	A character vector with variable codes to be queried. If NULL in downloadSMChistorical all the usual variables for meteoland are downloaded. Otherwise, the results are returned as a dara frame.
dates	An object of class Date with dates comprised within the last two years.
date	An object of class Date. By default the current day in the case of downloadSMCcurrentday(). In the case of downloadSMCstationlist() a date for which operational stations are queried.
station_id	A string vector containing station ids (the list of stations for which climatic series are available is given by downloadSMCstationlist()). If NULL, all available stations are queried. Otherwise, only the data corresponding to the specified stations will be returned.
export	If export = FALSE the downloaded data is stored in memory. Otherwise the result is written on the disk (using the format specified in exportFormat).

exportDir	Output directory for downloaded meteorology.
exportFormat	Format of meteorological data. Current accepted formats are "castanea" and "meteoland".
metadatafile	The name of the file that will store the meta data describing all written files.
verbose	Boolean flag to print process information.

Details

API key needs to be requested from SMC (https://apidocs.meteocat.gencat.cat/).

Value

Function downloadSMCstationlist returns a SpatialPointsDataFrame-class object containing the list of SMC operational weather stations for the date given.

Function downloadSMCvarmetadata returns a data frame with weather variables, their units and acronym to be used in queries (see parameter variable_code).

Function downloadSMCcurrentday downloads recent weather (the last 24h or the weather for a given date) from all currently available stations and returns data frame if daily_meteoland = FALSE or a SpatialPointsDataFrame-class object with observations aggregated at the daily scale otherwise.

Function downloadSMChistorical downloads historical daily weather corresponding to a given time period from a set (or all currently available) stations. Results are returned (or exported) after formatting data as a SpatialPointsMeteorology-class if variable_code = NULL, or as a data frame otherwise.

Author(s)

Antoine Cabon, CTFC

Miquel De Cáceres Ainsa, CTFC

References

Servei Meteorològic de Catalunya (SMC) should be acknowledged as author of information when accessing weather data with these functions.

See Also

SpatialPointsMeteorology-class

SpatialGridMeteorology *Creates a 'SpatialGridMeteorology'*

Description

Initializes an object of class SpatialGridMeteorology-class

Usage

```
SpatialGridMeteorology(grid, proj4string=CRS(as.character(NA)), data, dates)
```

Arguments

grid	An object of class GridTopology-class
proj4string	Object of class "CRS" with the projection string.
data	A vector of data frames (one per date).
dates	Object of class "Date" describing the time period of meteorological estimates.

Value

An object of class SpatialGridMeteorology-class

Author(s)

Miquel De Cáceres Ainsa, Centre Tecnologic Forestal de Catalunya

See Also

SpatialGridMeteorology-class

SpatialGridMeteorology-class *Class* "SpatialGridMeteorology"

Description

An S4 class that represents a spatial grid with meteorology daily data.

Objects from the Class

Objects can be created by calls of the form new("SpatialGridMeteorology", ...), or by calls to the function SpatialGridMeteorology.

Slots

`dates`: Object of class `"Date"` describing the time period for which meteorological estimates are available.

`data`: A vector of `"data.frame"` objects, each one containing the grid data for one date.

`bbox`: Object of class `"matrix"` with the boundary box.

`proj4string`: Object of class `"CRS"` with the projection string.

Extends

Class `"SpatialGrid"`, directly. Class `"Spatial"`, by class "SpatialGrid", distance 2.

Methods

`[ signature(x = "SpatialGridMeteorology", i = "ANY", j = "ANY", drop = "ANY")`: subsets the grid and associated meteorology; only rows (x values) and columns (y values) can be subsetted.

Author(s)

Miquel De Cáceres Ainsa, Centre Tecnologic Forestal de Catalunya

See Also

`SpatialGridTopography`, `SpatialGridDataFrame-class`

Examples

```
#Structure of the S4 object
showClass("SpatialGridMeteorology")
```

SpatialGridTopography *Creates a 'SpatialGridTopography'*

Description

Function `SpatialGridTopography` creates an object of class `SpatialGridTopography-class` containing topographic variables over a landscape.

Usage

```
SpatialGridTopography(grid, elevation, slope = NULL, aspect = NULL,
                      proj4string = CRS(as.character(NA)))
```

Arguments

grid	An object of class GridTopology-class or SpatialGrid-class.
elevation	A vector of elevation values for all cells of the grid (in m.a.s.l.).
slope	A vector of slope angles for all cells of the grid (in degrees). If slope=NULL, slope is calculated as indicated in details.
aspect	A vector of aspect angles for all cells of the grid (in degrees from North clockwise). aspect=NULL, aspect values are calculated as indicated in details.
proj4string	An object of class CRS-class.

Details

Slope and aspect calculations were adapted from functions in package 'SDMTools', which used the approach described in Burrough & McDonell (1998).

The rate of change (delta) of the surface in the horizontal (dz/dx) and vertical (dz/dy) directions from the center cell determines the slope and aspect. The values of the center cell and its eight neighbors determine the horizontal and vertical deltas. The neighbors are identified as letters from 'a' to 'i', with 'e' representing the cell for which the aspect is being calculated. The rate of change in the x direction for cell 'e' is calculated with the algorithm:

```
[dz/dx] = ((c + 2f + i) - (a + 2d + g) / (8 * x_cell_size)
```

The rate of change in the y direction for cell 'e' is calculated with the following algorithm:

```
[dz/dy] = ((g + 2h + i) - (a + 2b + c)) / (8 * y_cell_size)
```

The algorithm calculates slope as: `rise_run = sqrt ( [dz/dx]2 + [dz/dy]2 ])`.

From this value , one can calculate the slope in degrees or radians as:

```
slope_degrees = ATAN (rise_run) * 57.29578
```

```
slope_radians = ATAN (rise_run)
```

Taking the rate of change in both the x and y direction for cell 'e', aspect is calculated using:

```
aspect = 57.29578 * atan2 ([dz/dy], -[dz/dx])
```

The aspect value is then converted to compass direction values (0-360 degrees).

Value

Function SpatialGridTopography returns an object 'SpatialGridTopography-class'.

Author(s)

Miquel De Cáceres Ainsa, Biodiversity and Landscape Ecology Laboratory, Centre Tecnologic Forestal de Catalunya

References

Burrough, P. A. and McDonell, R.A., 1998. Principles of Geographical Information Systems (Oxford University Press, New York), p. 190.

See Also

extractSpatialData, SpatialGridTopography-class

Examples

```
data(examplegridtopography)

#Display data
spplot(examplegridtopography, variable="elevation", scales=list(draw=TRUE))

#Grids can be subsetted
sgt = examplegridtopography[1:50, 1:50]
spplot(sgt, variable="elevation", scales=list(draw=TRUE))
```

SpatialGridTopography-class

Class "SpatialGridTopography"

Description

An S4 class that represents topography over a grid of coordinates.

Objects from the Class

Objects can be created by calls of the form `new("SpatialGridTopography", ...)`, or by calls to the function SpatialGridTopography.

Slots

grid: Object of class GridTopology.

data: Object of class "data.frame" containing the elevation (in m), slope (in degrees) and aspect (in degrees from North) of every cell.

bbox: Object of class "matrix" with the boundary box.

proj4string: Object of class "CRS" with the projection string.

Extends

Class "SpatialGridDataFrame", directly. Class "SpatialGrid", by class "SpatialGridDataFrame", distance 2. Class "Spatial", by class "SpatialGridDataFrame", distance 3.

Methods

[`signature(x = "SpatialGridTopography", i = "ANY", j = "ANY", drop = "ANY")`: subsets the grid and associated topography; only rows (x values) and columns (y values) can be subsetted.

Author(s)

Miquel De Cáceres Ainsa, Biodiversity and Landscape Ecology Laboratory, Centre Tecnologic Forestal de Catalunya

See Also

SpatialGridTopography, SpatialGridDataFrame-class

Examples

```
#Structure of the S4 object
showClass("SpatialGridTopography")
```

SpatialPixelsMeteorology

Creates a 'SpatialPixelsMeteorology'

Description

Initializes an object of class SpatialPixelsMeteorology-class

Usage

```
SpatialPixelsMeteorology(points, data, dates,
                         tolerance = sqrt(.Machine$double.eps),
                         proj4string = CRS(as.character(NA)), round = NULL,
                         grid = NULL)
```

Arguments

points	An object of class SpatialPoints-class.
data	A vector of data frames (one per date).
dates	Object of class "Date" describing the time period of meteorological estimates.
tolerance	Precision up to which extent points should be exactly on a grid.
proj4string	Object of class CRS in the first form only used when points does not inherit from Spatial.
round	default NULL, otherwise a value passed to as the digits argument to round for setting cell size.
grid	Grid topology using an object of class GridTopology; a value of NULL implies that this will be derived from the point coordinates.

Value

An object of class SpatialPixelsMeteorology-class

Author(s)

Miquel De Cáceres Ainsa, Centre Tecnologic Forestal de Catalunya

See Also

SpatialPixelsMeteorology-class

SpatialPixelsMeteorology-class

Class "SpatialPixelsMeteorology"

Description

An S4 class that represents meteorology data that has locations on a regular grid.

Objects from the Class

Objects can be created by calls of the form `new("SpatialPixelsMeteorology", ...)`, or by calls to the function SpatialPixelsMeteorology.

Slots

`dates`: Object of class "Date" describing the time period for which meteorological estimates are available.

`data`: A vector of "data.frame" objects, each one containing the pixel data for one date.

`grid`: Grid parameters (see GridTopology).

`grid.index`: Index of points in full grid.

`coords`: Object of class "matrix" with the spatial coordinates.

`bbox`: Object of class "matrix" with the boundary box.

`proj4string`: Object of class "CRS" with the projection string.

Extends

Class "SpatialPixels", directly. Class "SpatialPoints", by class "SpatialPixels", distance 2. Class "Spatial", by class "SpatialPixels", distance 3.

Methods

[`signature(x = "SpatialPixelsMeteorology", i = "ANY", ..., drop = "ANY")`: subsets the pixels and associated topography; only one dimension can be subsetted. If `drop = TRUE` the boundary box is recalculated.

spplot `signature(object = "SpatialPixelsTopography")`: allows plotting topography maps.

Author(s)

Miquel De Cáceres Ainsa, Centre Tecnologic Forestal de Catalunya

See Also

SpatialPixelsTopography, SpatialPixelsDataFrame-class

Examples

```
#Structure of the S4 object
showClass("SpatialPixelsMeteorology")
```

SpatialPixelsTopography *Creates a 'SpatialPixelsTopography'*

Description

Function SpatialPixelsTopography creates an object of class SpatialPixelsTopography-class containing topographic variables for a set of points.

Usage

```
SpatialPixelsTopography(points, elevation, slope, aspect,
                        tolerance = sqrt(.Machine$double.eps),
                        proj4string = CRS(as.character(NA)), round = NULL,
                        grid = NULL)
```

Arguments

points	An object of class SpatialPoints-class or a numeric matrix of coordinates.
elevation	Elevation values (in m) of the points.
slope	Slope values (in degrees) of the points.
aspect	Aspect values (in degrees from North) of the points.
tolerance	Precision up to which extent points should be exactly on a grid.
proj4string	Object of class CRS in the first form only used when points does not inherit from Spatial.
round	default NULL, otherwise a value passed to as the digits argument to round for setting cell size.
grid	Grid topology using an object of class GridTopology; a value of NULL implies that this will be derived from the point coordinates.

Value

Function SpatialPixelsTopography returns an object 'SpatialPixelsTopography-class'.

Author(s)

Miquel De Cáceres Ainsa, Biodiversity and Landscape Ecology Laboratory, Centre Tecnologic Forestal de Catalunya

See Also

SpatialPixelsTopography-class

Examples

```
data(examplegridtopography)

#Creates spatial topography pixels as a subset of points in the grid
spt = as(examplegridtopography,"SpatialPointsTopography")
cc = spt@coords
center = 5160
d = sqrt((cc[,1]-cc[center,1])^2+(cc[,2]-cc[center,2])^2)
p = which(d<3000) #Select points at maximum distance of 3km from center
spxt = SpatialPixelsTopography(spt[p], spt$elevation[p],
                               spt$slope[p],
                               spt$aspect[p])

#Alternatively, use coercing and subsetting (drop = TRUE causes grid to be recalculated)
spxt = as(examplegridtopography, "SpatialPixelsTopography")[p, drop=TRUE]

#Display data
spplot(spxt, variable="elevation", scales=list(draw=TRUE))
spplot(spxt, variable="slope", scales=list(draw=TRUE))
spplot(spxt, variable="aspect", scales=list(draw=TRUE))
```

SpatialPixelsTopography-class

Class "SpatialPixelsTopography"

Description

An S4 class that represents topography that has locations on a regular grid.

Objects from the Class

Objects can be created by calls of the form `new("SpatialPixelsTopography", ...)`, or by calls to the function SpatialPixelsTopography.

Slots

data: Object of class "data.frame" containing the elevation (in m), slope (in degrees) and aspect (in degrees from North) of every cell.

coords.nrs: Inherited from SpatialPointsDataFrame but not used.

grid: Object of class GridTopology.

grid.index: Index of points in full grid.

coords: Object of class "matrix" with the spatial coordinates.

bbox: Object of class "matrix" with the boundary box.

proj4string: Object of class "CRS" with the projection string.

Extends

Class "SpatialPixelsDataFrame", directly. Class "SpatialPixels", by class "SpatialPixelsDataFrame", distance 2. Class "SpatialPointsDataFrame", by class "SpatialPixelsDataFrame", distance 2. Class "SpatialPoints", by class "SpatialPixelsDataFrame", distance 3. Class "Spatial", by class "SpatialPixelsDataFrame", distance 4.

Methods

[`signature(x = "SpatialGridTopography", i = "ANY", ..., drop = "ANY")`: subsets the pixels and associated topography; only one dimension can be subsetted. If `drop = TRUE` the boundary box is recalculated.

spplot `signature(object = "SpatialPixelsTopography")`: allows plotting topography maps.

Author(s)

Miquel De Cáceres Ainsa, Biodiversity and Landscape Ecology Laboratory, Centre Tecnologic Forestal de Catalunya

See Also

SpatialPixelsTopography, SpatialPixelsDataFrame-class

Examples

```
#Structure of the S4 object
showClass("SpatialPixelsTopography")
```

SpatialPointsMeteorology

Creates a 'SpatialPointsMeteorology'

Description

Initializes an object of class SpatialPointsMeteorology-class

Usage

```
SpatialPointsMeteorology(points, data, dates, dataByDate = FALSE)
```

Arguments

points	An object of class SpatialPoints-class. Row names of point coordinates are used to identify points.
data	A list of data frames. If `dataByDate = FALSE` the elements of data are assumed to correspond to points. If `dataByDate = TRUE` the elements of data are assumed to correspond to dates (see 'Details').
dates	Object of class "Date" describing the time period of meteorological estimates.

`dataByDate` A flag to indicate that elements of `data` correspond to dates, as opposed to the default (`dataByDate = FALSE`) which assumes that elements correspond to points (see 'Details').

Details

There are two ways of building an object of of class `SpatialPointsMeteorology-class`. The first way (`dataByDate = FALSE`) is to supply as value for `data` a vector of data frames with one data frame per spatial point, with dates as rows and meteorological variables as columns. In this case all data frames must have the same number of rows (dates) and columns (variables). The second way (if `dataByDate = TRUE`) is to supply as value for `data` a vector of data frames with one data frame per date, with points as rows and meteorological variables as columns. In this case, the data frames may have different rows and different columns. Only the information corresponding to `points` will be taken and some variables may be missing.

Value

An object of class `SpatialPointsMeteorology-class`

Author(s)

Miquel De Cáceres Ainsa, Centre Tecnologic Forestal de Catalunya

See Also

`SpatialPointsMeteorology-class`

SpatialPointsMeteorology-class

Class "SpatialPointsMeteorology"

Description

An S4 class that represents a set of points with meteorology data series.

Objects from the Class

Objects can be created by calls of the form `new("SpatialPointsMeteorology", ...)`, or by calls to the function `SpatialPointsMeteorology`.

Slots

`dates`: Object of class "Date" describing the time period for which meteorological estimates are available.

`data`: A vector of "data.frame" objects, each one corresponding to one spatial point.

`coords`: Object of class "matrix" with the spatial coordinates.

`bbox`: Object of class "matrix" with the boundary box.

`proj4string`: Object of class "CRS" with the projection string.

Extends

Class "SpatialPoints", directly. Class "Spatial", by class "SpatialPoints", distance 2.

Methods

[signature(x = "SpatialPointsMeteorology", i = "ANY", j = "ANY", drop = "ANY"): subsets the spatial points and the corresponding list of meteorological data; only rows (points) can be subsetted.

Author(s)

Miquel De Cáceres Ainsa, Centre Tecnologic Forestal de Catalunya

See Also

SpatialPointsTopography-class, SpatialPoints-class

Examples

```
#Structure of the S4 object
showClass("SpatialPointsMeteorology")
```

SpatialPointsTopography *Creates a 'SpatialPointsTopography'*

Description

Function SpatialPointsTopography creates an object of class SpatialPointsTopography-class containing topographic variables for a set of points.

Usage

```
SpatialPointsTopography(points, elevation, slope = NULL, aspect = NULL,
                        proj4string = CRS(as.character(NA)))
```

Arguments

points	An object of class SpatialPoints-class.
elevation	Elevation values (in m) of the points.
slope	Slope values (in degrees) of the points.
aspect	Aspect values (in degrees from North) of the points.
proj4string	Object of class CRS in the first form only used when points does not inherit from Spatial.

Details

If either `slope = NULL` or `aspect = NULL` then when estimating weather on the object locations radiation will be calculated assuming a flat surface.

Value

Function `SpatialPointsTopography` returns an object 'SpatialPointsTopography-class'.

Author(s)

Miquel De Cáceres Ainsa, Biodiversity and Landscape Ecology Laboratory, Centre Tecnologic Forestal de Catalunya

See Also

SpatialPointsTopography-class

Examples

```
data(examplegridtopography)

#Creates spatial topography points from the grid
p = 1:2
points = as(examplegridtopography,"SpatialPoints")[p]
spt = SpatialPointsTopography(points, examplegridtopography$elevation[p],
                              examplegridtopography$slope[p],
                              examplegridtopography$aspect[p])
spt

#Alternatively, use coercing and subsetting
spt = as(examplegridtopography, "SpatialPointsTopography")[p]
spt
```

SpatialPointsTopography-class

Class "SpatialPointsTopography"

Description

An S4 class that represents topography over a grid of coordinates.

Objects from the Class

Objects can be created by calls of the form `new("SpatialPointsTopography", ...)`, or by calls to the function SpatialPointsTopography.

Slots

`data`: Object of class `"data.frame"` containing the elevation (in m), slope (in degrees) and aspect (in degrees from North) of every point.

`coords`: Object of class `"matrix"` with the spatial coordinates.

`bbox`: Object of class `"matrix"` with the boundary box.

`proj4string`: Object of class `"CRS"` with the projection string.

Extends

Class `"SpatialPointsDataFrame"`, directly. Class `"SpatialPoints"`, by class "SpatialPointsDataFrame", distance 2. Class `"Spatial"`, by class "SpatialPointsDataFrame", distance 3.

Methods

`[ signature(x = "SpatialPointsTopography", i = "ANY", j = "ANY", drop = "ANY")`: subsets the spatial points and associated topography; only rows (points) can be subsetted.

Author(s)

Miquel De Cáceres Ainsa, Biodiversity and Landscape Ecology Laboratory, Centre Tecnologic Forestal de Catalunya

See Also

`SpatialPointsTopography`, `SpatialPointsDataFrame-class`

Examples

```
#Structure of the S4 object
showClass("SpatialPointsTopography")
```

---

spplot *Spatial grid plots*

Description

Function `spplot` for `SpatialGridTopography-class` and `SpatialPixelsTopography-class` objects allows drawing maps of topographic attributes. Function `spplot` for `SpatialGridMeteorology-class` and `SpatialPixelsMeteorology-class` objects allows drawing maps of meteorological variables corresponding to specific dates.

Usage

```
## S4 method for signature 'SpatialGridTopography'
spplot(obj, variable="elevation",...)
## S4 method for signature 'SpatialGridMeteorology'
spplot(obj, date, variable="MeanTemperature", ...)
## S4 method for signature 'SpatialPixelsTopography'
spplot(obj, variable="elevation",...)
## S4 method for signature 'SpatialPixelsMeteorology'
spplot(obj, date, variable="MeanTemperature", ...)
```

Arguments

obj	An object of class SpatialGridTopography.
variable	A string of the variable to be plotted (only type="elevation", type="slope", type="aspect" are allowed).
...	Additional parameters to function spplot.
date	A string or an integer for the date to be plotted.

Author(s)

Miquel De Cáceres Ainsa, Biodiversity and Landscape Ecology Laboratory, Centre Tecnologic Forestal de Catalunya

See Also

meteoplot

Examples

```
data(examplegridtopography)

#Display data
spplot(examplegridtopography, type="elevation", scales=list(draw=TRUE))
spplot(examplegridtopography, type="slope", scales=list(draw=TRUE))
spplot(examplegridtopography, type="aspect", scales=list(draw=TRUE))
```

---

subsample *Sub-sampling procedure data*

Description

Generates a spatial and/or temporal subset of procedure data

Usage

```
## S4 method for signature 'MeteorologyUncorrectedData'
subsample(object, bbox = NULL, stations = NULL, dates = NULL, buffer = 0)
## S4 method for signature 'MeteorologyInterpolationData'
subsample(object, bbox = NULL, stations = NULL, dates = NULL, buffer = 0)
```

Arguments

object	An object of a sub-class MeteorologyProcedureData-class.
bbox	A 2x2 numeric matrix with the boundaries of the target area. If NULL, the original boundary box is kept (except if stations is specified).
stations	A numeric, character or logical vector specifying a subset of weather stations. If NULL all original weather stations are kept (except if bbox is specified).
dates	A vector of Date with the subset of dates of interest. If NULL, all dates are kept.
buffer	A buffer to put around bbox for the spatial selection of data.

Value

An object of the same class as object.

Methods

subsample signature(object = "MeteorologyUncorrectedData"): Generates a MeteorologyUncorrectedData object for a smaller area and a subset of dates.

subsample signature(object = "MeteorologyInterpolationData"): Generates a MeteorologyInterpolationData object for a smaller area and a subset of dates.

Examples

```
data(exampleinterpolationdata)

oridates = exampleinterpolationdata@dates

#Interpolation data using the first ten dates (same boundary box)
subdata = subsample(exampleinterpolationdata, dates = oridates[1:10])
```

---

summarypoints *Summaries of meteorological data*

Description

Summarizes the meteorology of a single location, a set of spatial points, pixels in a grid, or weather stations of interpolation data.

Usage

```
summarypoint(x, var, fun = mean, freq = NULL, dates = NULL, months = NULL, ...)
summarypoints(points, var, fun = mean, freq = NULL, dates = NULL, months = NULL, ...)
summarygrid(griddata, dates = NULL)
summarypixels(pixelsdata, dates = NULL)
summaryinterpolationdata(object, var, fun = mean, freq = NULL, dates = NULL,
                         months = NULL, ...)
```

Arguments

x	A data frame with dates in rows and meteorological variables in columns.
points	An object of class SpatialPointsMeteorology-class with the coordinates and meteorological data of the locations for which summaries are desired. Alternatively, an object of class SpatialPointsDataFrame-class containing the meta data (columns dir, filename and possibly format) of meteorological files that will be sequentially read from the disk.
var	The name of the meteorological variable to be summarized.
fun	The function to be calculated on values of each point. If freq is specified, the function will be calculated by intervals.
freq	A string giving an interval specification (e.g., "week", "month", "quarter" or "year"). If NULL then no intervals are defined.
dates	An object of class Date to define the period to be summarized. If dates = NULL then all dates in points are processed.
months	A numeric vector to indicate the subset of months for which summary is desired (e.g. c(7,8) for July and August). This parameter allows studing particular seasons, when combined with freq. For example freq = "years" and months = 6:8 leads to summarizing summer months of each year.
...	Additional parameters to fun.
griddata	An object of class SpatialGridMeteorology-class with the meteorological data for a grid. Alternatively, a data.frame containing the meta data (columns dir and filename) of grid meteorological files that will be sequentially read from the disk.
pixelsdata	An object of class SpatialPixelsMeteorology-class with the meteorological data for a set of grid cells. Alternatively, a data.frame containing the meta data (columns dir and filename) of grid meteorological files that will be sequentially read from the disk.
object	An object of class MeteorologyInterpolationData-class.

Value

Function summarypoint returns a named vector of values with dates as names. Functions summarypoints and summaryinterpolatoiondata return an object of class SpatialPointsDataFrame containing summaries (either one variable or several if freq is specified). Functions summarygrid and summarypixels return an object of class SpatialGridDataFrame and SpatialPixelsDataFrame, respectively, containing the summary of all meteorogical variables (sums or precipitation and PET; means for the other variables) calculated over the specified period.

Author(s)

Miquel De Cáceres Ainsa, CTFC

Antoine Cabon,CTFC

See Also

SpatialPointsMeteorology-class

Examples

```
data(examplegridtopography)
data(exampleinterpolationdata)

#Creates spatial topography points from the grid
p = 1:2
spt = as(examplegridtopography, "SpatialPointsTopography")[p]

#Interpolation of two points for the whole time period (2000-2003)
mp = interpolationpoints(exampleinterpolationdata, spt)

#PET sums by months
mp.sum = summarypoints(mp, var="PET", freq="months")

mp.sum
```

---

utils *Physical utility functions*

Description

Set of functions used in the calculation of physical variables.

Usage

```
utils_airDensity(temperature, Patm)
utils_atmosphericPressure(elevation)
utils_averageDailyVP(Tmin, Tmax, RHmin, RHmax)
utils_averageDaylightTemperature(Tmin, Tmax)
utils_latentHeatVaporisation(temperature)
utils_latentHeatVaporisationMol(temperature)
utils_psychrometricConstant(temperature, Patm)
utils_saturationVP(temperature)
utils_saturationVaporPressureCurveSlope(temperature)
```

Arguments

`temperature`	Air temperature (ºC).
`Tmin, Tmax`	Minimum and maximum daily temperature (ºC).
`RHmin, RHmax`	Minimum and maximum relative humidity (%).
`Patm`	Atmospheric air pressure (in kPa).
`elevation`	Elevation above sea level (in m).

Value

Values returned for each function are:

- `utils_airDensity`: air density (in kg·m-3).
- `utils_atmosphericPressure`: Air atmospheric pressure (in kPa).
- `utils_averageDailyVP`: average (actual) vapour pressure (in kPa).
- `utils_averageDaylightTemperature`: average daylight air temperature (in ºC).
- `utils_latentHeatVaporisation`: Latent heat of vaporisation (MJ·kg-1).
- `utils_latentHeatVaporisationMol`: Latent heat of vaporisation (J·mol-1).
- `utils_psychrometricConstant`: Psychrometric constant (kPa·ºC-1).
- `utils_saturationVP`: saturation vapour pressure (in kPa).
- `utils_saturationVaporPressureCurveSlope`: Slope of the saturation vapor pressure curve (kPa·ºC-1).

Author(s)

Miquel De Cáceres Ainsa, Centre Tecnologic Forestal de Catalunya

References

McMurtrie, R. E., D. A. Rook, and F. M. Kelliher. 1990. Modelling the yield of Pinus radiata on a site limited by water and nitrogen. Forest Ecology and Management 30:381–413.

McMahon, T. A., M. C. Peel, L. Lowe, R. Srikanthan, and T. R. McVicar. 2013. Estimating actual, potential, reference crop and pan evaporation using standard meteorological data: a pragmatic synthesis. Hydrology & Earth System Sciences 17:1331–1363. See also: http://www.fao.org/docrep/x0490e/x0490e06.htm

writemeteorologygrid *Writes grid meteorology to the disk*

Description

Functions to write grid meteorological data to the file system.

Usage

```
writemeteorologygrid(object, date, file, format = "netCDF")
writemeteorologygridfiles(object, dir=getwd(), format = "netCDF",
                          metadatafile = "MG.txt")
```

Arguments

object	An object of class SpatialGridMeteorology-class with the meteorological data to be written.
date	A Date object or a character string indicating the date of the meteorological grid to be written.
file	A string with the file name to be written.
format	Format of meteorological data. The only accepted format is "netCDF".
dir	Output directory for meteorology data.
metadatafile	The name of the file that will store the meta data describing all written files.

Author(s)

Miquel De Cáceres Ainsa, Centre Tecnologic Forestal de Catalunya

See Also

readmeteorologygrid, SpatialGridMeteorology-class

writemeteorologypixels *Writes pixel meteorology to the disk*

Description

Functions to write meteorological data located on pixels of a grid onto the file system.

Usage

```
writemeteorologypixels(object, date, file, format = "netCDF")
writemeteorologypixelsfiles(object, dir=getwd(), format = "netCDF",
                          metadatafile = "MG.txt")
```

Arguments

object	An object of class SpatialPixelsMeteorology-class with the meteorological data to be written.
date	A Date object or a character string indicating the date of the meteorological data to be written.
file	A string with the file name to be written.
format	Format of meteorological data. The only accepted format is "netCDF".
dir	Output directory for meteorology data.
metadatafile	The name of the file that will store the meta data describing all written files.

Author(s)

Miquel De Cáceres Ainsa, Centre Tecnologic Forestal de Catalunya

See Also

readmeteorologypixels, SpatialPixelsMeteorology-class

---

writemeteorologypoint *Writes point meteorology to the disk*

Description

Functions to write point meteorological data to ascii files in different formats.

Usage

```
writemeteorologypoint(data, file, format = "meteoland/txt")
writemeteorologypointfiles(object, dir=getwd(), format = "meteoland/txt",
                           metadatafile="MP.txt")
```

Arguments

data	An data frame with meteorological data.
file	A string with the file name to be written.
format	Output format of meteorological data. Current accepted formats are "meteoland/txt", "meteoland/rds", "castanea/txt" and "castanea/rds".
object	An object of class SpatialPointsMeteorology-class with the meteorological data to be written.
dir	Output directory for meteorology data.
metadatafile	The name of the file that will store the meta data describing all written files.

Author(s)

Miquel De Cáceres Ainsa, Centre Tecnologic Forestal de Catalunya

Nicolas Martin, INRA-Avignon

See Also

readmeteorologypoint, SpatialPointsMeteorology-class

Index