Seventh Framework Programme
Theme 6

Project: 607193 UERRA

Full project title:
Uncertainties in **E**nsembles of **R**egional **R**e-**A**nalyses

Deliverable D1.12
Gridding resolution enhancement

WP no:	1
WP leader:	URV
Lead beneficiary for deliverable :	KNMI
Name of author/contributors:	Richard Cornes and Phil Jones
Nature:	Other
Dissemination level:	PU
Deliverable month:	24
Submission date: May 6, 2016	Version nr: 1

Report for Deliverable 1.12 (D1.12): Beta version of enhanced gridding software available to project partners

By Richard Cornes[1 and 2] and Phil Jones[2]

[1] KNMI, Netherlands
[2] Climatic Research Unit, School of Environmental Sciences, University of East Anglia, Norwich, UK

The original gridding software used for E-OBS was written in Fortran. To allow for the developments of the E-OBS under the UERRA project (some of which are described in Deliverable D1.10) and to improve the efficiency of the code, a new software package has been produced, written in the R programming language (R Development Core Team, 2008). In the first instance this software package replicates the methods used in the original Fortran code, but the software also includes many additional functions that achieve the objectives being conducted under UERRA of improving and developing the E-OBS dataset. The use of this programming language allows easier dissemination of the software to project partners and ultimately to the wider research community. The manual for the package "eobs" is included in the Appendix of this document and the software, currently in restricted beta form and only for use by project partners, is available on request to Richard Cornes at KNMI (cornes@knmi.nl).

The software package is designed to allow the gridding procedures used in E-OBS to make greater use of parallel computing, which greatly reduces the time taken to complete a new run of the E-OBS dataset. This allows more computationally intensive procedures to be incorporated into the gridding procedures (such as the regression kriging technique described in deliverable D1.10) without significantly lengthening the time taken to complete a gridding run.

The code used in the "eobs" R package is currently being run on the high-performance computing cluster (HPCF) at the European Centre for Medium-Range Weather Forecasts (ECMWF). The running of the scripts is largely automated through the use of the ECTRANS suite of programs. This marks a significant development in the gridding of the E-OBS dataset: in the first instance the runtime for the gridding jobs is reduced from a few weeks to a few days, but the use of this cluster will allow more intensive gridding procedures to be used in the future. The prime example of this is the generation of an ensemble of realizations, which will be the subject of Deliverable D1.14.

Links

ECMWF HPCF: http://www.ecmwf.int/en/computing/our-facilities/supercomputer)
ECTRANS: https://software.ecmwf.int/wiki/display/ECAC/Unattended+file+transfer+-+ectrans

References

R Development Core Team (2008). R: A language and environment for statistical computing. R Foundation for Statistical Computing, Vienna, Austria. ISBN 3-900051-07-0, URL http://www.R-project.org.

Appendix: Manual for the “eobs” R package

Package ‘eobs’

May 6, 2016

Title Functions to grid station-based climate data

Version 0.1

Author ``Richard Cornes <cornes@knmi.nl> [aut, cre]"

Maintainer Richard Cornes <cornes@knmi.nl>

Description A collection of functions for the gridding of station data using the approach used in the E-OBS dataset. The gridding is done using thin-plate splines for monthly data, with the daily data being gridded using kriging. The functions are optimized for use on a computing cluster, principally by allowing the gridding to be split by months, i.e. each month of data may be sent to a separate computing node for processing. However, the functions will also run without such computing infrastructure.

Depends R (>= 3.0.0), methods

Imports data.table (>= 1.9.2), foreach, reshape, raster (>= 2.4-20), gstat, fields (>= 8.2-1), LatticeKrig (>= 5.4-1), rgcvpack (>= 0.1-4), lubridate (>= 1.5.0), sp (>= 1.2-1), rts (>= 1.0-10)

License file LICENSE

LazyData true

Suggests geosphere, lmomco, fitdistrplus, knitr, yaml, doParallel, mgcv (>= 1.8-4), snow (>= 0.4-1)

NeedsCompilation yes

VignetteBuilder knitr

RoxygenNote 5.0.1

R topics documented:

`eobs-package` *Grid climate data using the E-OBS scheme*

Description

`eobs` provides a suite of tools to allow gridding of station-based climate data using the techniques used for the E-OBS dataset produced by the ECA&D group at KNMI.

Details

The original code used to produce E-OBS, and described by Haylock et al. (2008), was writen in Fortran. To allow extensions to the code and to make the code more efficient this package was written to reproduce the output of the Fortran code. Clearly there will be certain differences, but testing has revealed that these are relatively minor. The functions in this package make extensive use of the `raster` and `sp` packages, with the kriging functions being based on functions from the `gstat` package.

The gridding functions operate on time slices of the data, which are referred to as "periods". The value of the period depends on the time frame from which the daily anomalies are calculated. Typically the periods will be months, but they could be an time slice up to a year. This slicing allows parallelization of the gridding by simultaneously gridding the years/periods.

References

Haylock, M.R., N. Hofstra, A.M.G. Klein Tank, E.J. Klok, P.D. Jones, M. New. 2008: A European daily high-resolution gridded dataset of surface temperature and precipitation. J. Geophys. Res (Atmospheres), 113, D20119, doi:10.1029/2008JD10201

van den Besselaar, E.J.M., M.R. Haylock, G. van der Schrier and A.M.G. Klein Tank. 2011: A European Daily High-resolution Observational Gridded Data set of Sea Level Pressure. J. Geophys. Res., 116, D11110, doi:10.1029/2010JD015468

`clusterAverage` *Average stations within a specified distance.*

Description

This function uses the `zerodist` function to define clusters of stations, which are then averaged, i.e. the `distance` variable is used as the `zero` variable in that function. As a result the unit of `distance` depends on the coordinates specified in `obj`, which for longitude/latitude is KM. The default value of 1 matches the range used in the original E-OBS gridding code.

Usage

```
clusterAverage(obj, distance = 1, cluster_only = FALSE)
```

Arguments

`obj`	An object of class `SpatialPointsDataFrame`
`distance`	The distance over which averaging should take place.
`cluster_only`	If `TRUE` only the cluster identifier is returned

Value

A `SpatialPointsDataframe`

Author(s)

Richard Cornes

`combine.vg` *Average daily semi-variograms*

Description

Average daily semi-variograms

Usage

```
combine.vg(ifiles)
```

Arguments

`ifiles`	A list of .RDS files continaing the variograms

Details

This simple function averages each of the daily variograms produced by the function 'semi.VG'. This is the method that was originally used in the E-OBS dataset (see Haylock et al., 2008).

Value

The averaged variogram of class `data.table`

Author(s)

Richard Cornes

`dayGrid` *Grid Daily Anomalies using kriging*

Description

This function uses the `krige` function from the `gstat` package to interpolate the daily anomalies to a gridded format. The monthly gridded absolute values are then added back to the anomalies to form the final daily values.

Usage

```
dayGrid(obj, ...)

## S4 method for signature 'eobs'
dayGrid(obj, MonGrid, vg.dat, verbose = TRUE, nbmax = 75,
  maxdist = 500, cl = NULL)

## S4 method for signature 'eobs_rain'
dayGrid(obj, MonGrid, vg.dat, vg.prob, verbose = TRUE,
  nbmax = 75, maxdist = 450, ind_prob = 0.4, thresh = 0.5, cl = NULL)
```

Arguments

`obj`	An object of class `eobs`
`...`	Parameters passed to respective functions
`MonGrid`	An object of class `SpatialPointsDataFrame` containing the gridded monthly data
`vg.dat`	An object of class `gstatVariogram` containing the variogram of the daily anomalies. This will probably have been derived from a call to `semiVG`
`verbose`	If `TRUE` intermediate messages will be printed
`nbmax`	The maximum number of neighbours to be used in kriging. This will be passed to the `krige` function in the `gstat` package
`maxdist`	The maximum distance to be used in kriging. This will be passed to the `krige` function in the `gstat` package
`cl`	A cluster object
`vg.prob`	An object of class `gstatVariogram` containing the variogram of the daily probability of rainfall. This will probably have been derived from a call to `semiVG`
`ind_prob`	The threshold used in the final gridded data to define a wet-day (see Haylock et al. 2008)
`thresh`	The treshold used to define a wet-day (see Haylock et al. 2008)

Value

An object of `SpatialPointsDataFrame` containing the daily gridded monthly data

Methods (by class)

- `eobs`: For class "eobs"
- `eobs_rain`: For class "eobs_rain"

Author(s)

Richard Cornes

See Also

`gridTPS`, `krige`

`daysinperiod` *Days in the Year/Period*

Description

Get a vector of days in the given Year/Period

Usage

```
daysinperiod(obj, Year, Period)
```

Arguments

`obj`	Object of class eobs
`Year`	The year in question
`Period`	The period in question

Value

Vector of class "POSIXct" "POSIXt"

Author(s)

Richard Cornes

`eobs-class` *An S4 class for* `eobs` *general variables*

Description

An S4 class for `eobs` general variables

Slots

`lon_rng` Longitude range vector
`lat_rng` Latitude range vector
`var` Name of the variable
`rain` Logical indicating if rain is being gridded
`base` Base period range
`yrs` Year range
`periods` The period definition
`grid` The high-resolution grid
`stn` A spatial dataframe of station data
`data` The current station data being processed
`stn.id` A vector of unique station references

`fields` *Gridding using the* `Tps` *function*

Description

As `rgcv` but using the thin-plate spline functions from the `fields` package

Usage

```
fields(obj, yvar = "VarSum", rhs = ~lon + lat + I(alt/1000), cost = 1,
  Z = NULL, rev.trans = TRUE, se.fit = FALSE, cl = NULL, ...)
```

Arguments

`obj`	Object of class `eobs` or one of its derivatives
`yvar`	The left-hand side of the equation as a character string
`rhs`	The right-hand side of the equation as a `formula`
`cost`	Cost function supplied to `fitTps` function
`Z`	The `Z` component passed to the `Tps` function
`rev.trans`	If `TRUE` the values are converted back from a transformed state. For example, in the default case the values are returned as square-roots of the gridded values.
`se.fit`	If `FALSE` standard error values are not calculated
`cl`	A cluster object
`...`	Further parameters passed to the spline fitting function function

Value

An object of class `SpatialPointDataFrame`

Author(s)

Richard Cornes

See Also

`rgcv`, `gridTPS`, `Tps`

Other TPS.functions: `gam.hybrid`, `gridTPS`, `lkrig`, `rgcv`, `tprs`

gam.hybrid *Gridding using Regression-kriging*

Description

Gridding using Regression-kriging

Usage

```
gam.hybrid(obj, tprs.run, gam.res, lzn.fit, maxdist = 500, nmax = 15,
  nmin = 4, nsim = 0, cl = NULL)
```

Arguments

obj	Object of class eobs or one of its derivatives
tprs.run	The list returned from a call to a tps function
gam.res	The residuals from the tps function
lzn.fit	The variogram fitted to the residuals
maxdist	Maximum search radius in the krige function
nmax	Maximum points in the krige function
nmin	Minimum points in the krige function
nsim	Number of simulations used in Maximum points in the krige function
cl	A cluster object

Value

An object of class SpatialPointDataFrame

Author(s)

Richard Cornes

See Also

gridTPS

Other TPS.functions: fields, gridTPS, lkrig, rgcv, tprs

get_meta *Internal S4 Generic for extracting a spline's meta-data*

Description

Internal S4 Generic for extracting a spline's meta-data

Usage

```
get_meta(obj, ...)

## S4 method for signature 'gam'
get_meta(obj)

## S4 method for signature 'bam'
get_meta(obj)

## S4 method for signature 'Krig'
get_meta(obj)

## S4 method for signature 'LKrig'
get_meta(obj)

## S4 method for signature 'LatticeKrig'
get_meta(obj)

## S4 method for signature 'Tps'
get_meta(obj)
```

Arguments

obj	Spline class obtained from a call to gridTPS
...	Other values passed to the meta function

Author(s)

Richard Cornes

gridTPS *Grid the [monthly] station data*

Description

This is a convenience function that applies one of the selected gridding procedures to the station data.

Usage

```
gridTPS(obj, method = c("rgcv", "fields", "lkrig", "tprs", "gam.hybrid"),
  verbose = TRUE, ...)
```

Arguments

`obj`	Object of class eobs or one of its derivates
`method`	The spline method to be used
`verbose`	If `TRUE` intermediate messages will be printed
`...`	Parameters passed to the respective spline functions

Details

The formula used for the spline model is built using the `yvar` and `rhs` components, and provides a great deal of flexibility in the model fitting. For example, using the default components the following model is fitted

$y^2 = f(lon, lat, alt)$

where alt is scaled by 1000. This is the recommended value for a thin-plate spline fitted to longitude and latitude values scaled in degrees, and altitude values initially in metres are scaled in kilometres.

If a valid cluster object is supplied then the prediction is split across the cluster nodes. This will give a signficiant speed-up of the function. The cluster object (`cl`) is that required by the `clusterSplit` function. For parallel processing to occur the cluster object needs to be registered such that the `foreach` function can operate in parallel.

The values are predicted at either the `GRID` locations of at the `stn.test` locations: one of these options must be selected. If `stn.test` are specified then these locations are removed from the locations used in the spline fitting. This function grids station data using a thin-plate regression spline method from the `mgcv` package.

Value

An object of class `SpatialPointDataFrame`

Author(s)

Richard Cornes

References

Haylock, M.R., N. Hofstra, A.M.G. Klein Tank, E.J. Klok, P.D. Jones, M. New. 2008: A European daily high-resolution gridded dataset of surface temperature and precipitation. J. Geophys. Res (Atmospheres), 113, D20119, doi:10.1029/2008JD10201

van den Besselaar, E.J.M., M.R. Haylock, G. van der Schrier and A.M.G. Klein Tank. 2011: A European Daily High-resolution Observational Gridded Data set of Sea Level Pressure. J. Geophys. Res., 116, D11110, doi:10.1029/2010JD015468

See Also

startEOBS

Other TPS.functions: fields, gam.hybrid, lkrig, rgcv, tprs

Examples

```
## Not run:
eobs <- startEOBS.yaml("RR.yaml")
grid <- gridTPS(eobs, "rgcv")

## End(Not run)
```

---

lkrig *Gridding using LatticeKrig*

Description

Produce a Thin-Plate Spline interpolation using the /link/codeLatticeKrig package.

Usage

```
lkrig(obj, yvar = "VarSum^2", rhs = ~lon + lat, cost = 1, Z = ~alt,
  rev.trans = TRUE, se.fit = FALSE, draws = 30, cl = NULL)
```

Arguments

obj	Object of class eobs or one of its derivatives
yvar	The left-hand side of the equation as a character string
rhs	The right-hand side of the equation as a formula
cost	Cost function supplied to fitTps function
Z	The Z component passed to the Tps function
rev.trans	If TRUE the values are converted back from a transformed state. For example, in the default case the values are returned as square-roots of the gridded values.
se.fit	If FALSE standard error values are not calculated
draws	The number of simulations to produce
cl	A cluster object

Value

An object of class SpatialPointDataFrame

Author(s)

Richard Cornes

See Also

gridTPS, LatticeKrig

Other TPS.functions: fields, gam.hybrid, gridTPS, rgcv, tprs

NLgriddata *Dutch Rainfall Gridding Data*

Description

In these data the NLRainfall data have been processed by prep_data and the results have been amalgamated into data.tables which contain the results for all stations. These data can then by processed using the gridding routines in this package

Usage

```
NLgriddata
```

Format

A list 3 components:

day A data.table containing the daily data

month A data.table containing the monthly data

climate A data.table containing the climatology data

NLhires *Dutch High-resolution topography*

Description

Elevation data for the Netherlands taken from the gtopo30 dataset. These data have been resampled to 0.1 degree resolution and the coordinates have been transformed. This is a section of the high-resolution data for Europe used in the original version of E- OBS

Usage

```
NLhires
```

Format

A *raster* object with 1 variables:

alt Altitude in metres

Source

https://lta.cr.usgs.gov/GTOPO30

`NLRainfall` *Daily Rainfall Totals*

Description

A dataset containing daily rainfall totals from 37 automatic weather stations (AWS) across the Netherlands, over the period 1950-99. Note that certain stations may contain data for the full period. In the original data values > 0.5mm are marked as -1. These trace values have been changed here to zero.

Usage

```
NLRainfall
```

Format

A data frame with 344754 rows and 3 variables:

STATION Unique station ID number
YYYYMMDD Date, in the format year, month, day
VALUE Daily Rainfall Total in mm

Source

http://www.knmi.nl/climatology/daily_data/selection.cgi

`NLStations` *Dutch AWS Station Meta-data*

Description

A dataset containing meta-data from the 37 automatic weather stations across the Netherlands

Usage

```
NLStations
```

Format

A data frame with 37 rows and 6 variables:

ID Unique station ID number
LAT Station Latitude in decimal degrees North
LON Station Longitude in decimal degrees East
ALT Station Altitude in metres
STATION_NAME Name of the station
COUNTRY_NAME Country of the station - always NL

Source

http://www.knmi.nl/climatology/daily_data/selection.cgi

---

prep_data *Data Preparation S4 Generic*

Description

Data Preparation S4 Generic

Usage

```
prep_data(obj, ...)

## S4 method for signature 'eobs'
prep_data(obj, dat, id, nthresh = 0.2, baseThresh = 0,
  climateThresh = baseThresh, verbose = TRUE, ...)

## S4 method for signature 'eobs_rain'
prep_data(obj, dat, id, nthresh = 0.2, baseThresh = 0,
  climateThresh = baseThresh, verbose = TRUE, trace = 0.05,
  gammaFunc = c("GammaFunc.lmom", "GammaFunc.censor", "GammaFunc.SCI"), ...)

## S4 method for signature 'eobs_dtr'
prep_data(obj, tmax, tmin, id, nthresh = 0.2,
  baseThresh = 0, climateThresh = baseThresh, verbose = TRUE, ...)

## S4 method for signature 'eobs_tmn'
prep_data(obj, tmax, tmin, id, nthresh = 0.2,
  baseThresh = 0, climateThresh = baseThresh, verbose = TRUE, ...)
```

Arguments

obj	Object of class eobs
...	Parameters passed to respective functions
dat	data.frame with the following columns: year,month,day,var
id	The unique station id
nthresh	Threshold of missing values tolerated. Defaults to 0.2, i.e. <20% values tolerated
baseThresh	The proportion of missing values permitted in the base-period calculations. Defaults to zero, i.e. must be complete
climateThresh	The proportion of missing values permitted in the climatology calculations. Defaults to the same value as baseThresh
verbose	If true messages are printed to screen

`trace`	Values below which rainfall is treated as missing
`gammaFunc`	The gamma function to use. Either `GammaFunc.censor` or `GammaFunc.lmom`
`tmax`	Maximum daily temperature data.frame
`tmin`	Minimum daily temperature data.frame

Value

List containing data.tables under the titles: day, month and param, respectively daily and monthly processed data and parameters by period.

Methods (by class)

- `eobs`: class "eobs"
- `eobs_rain`: class "eobs_rain"
- `eobs_dtr`: class "eobs_dtr"
- `eobs_tmn`: class "eobs_tg"

Author(s)

Richard Cornes

Examples

```
library(data.table)
data(NLRainfall)
data(NLStations)

## Initiate the eobs class
NL <- startEOBS(c(3.27,7.43),c(50.9,53.8),c(1981,2010),c(1971,2014),1:12,var="rr")
stns <- NLStations$ID

## Select one station
id <- 210
raw <- NLRainfall[NLRainfall$STATION==id,]
yrs <- as.numeric(substr(raw$YYYYMMDD,1,4))
mons <- as.numeric(substr(raw$YYYYMMDD,5,6))
days <- as.numeric(substr(raw$YYYYMMDD,7,8))
dat <- data.table(year=yrs, month=mons, day=days, var=raw$VALUE)

## Calculate the statistics
stats <- prep_data(NL, dat, id)
```

`readECAD.full` *Read ECA&D data file*

Description

Read and process the contents of an ECA&D full data file

Usage

```
readECAD.full(dat, QC = TRUE, verbose = TRUE)
```

Arguments

`dat`	Data.frame in ECAD format
`QC`	If TRUE values classified by ECA&D QC as suspect are removed. Applies to any QC code >0
`verbose`	If TRUE prints debugging messages

Value

data.table with columns date, var and id

Author(s)

Richard Cornes

`regrid` *Regridding using box averaging S4 Generic*

Description

Regridding using box averaging S4 Generic

Usage

```
regrid(obj, ogrid, ...)

## S4 method for signature 'RasterLayer'
regrid(obj, ogrid, ...)

## S4 method for signature 'RasterBrickTS'
regrid(obj, ogrid, ...)
```

Arguments

obj	An object of class *raster* containing data to be regridded
ogrid	An object of class *raster* containing the output grid
...	Not Currnently used

Methods (by class)

- RasterLayer: For class "RasterLayer"
- RasterBrickTS: For class "RasterBrickTS"

Author(s)

Richard Cornes

---

regrid_rot *Regridding using the "chain" method S4 Generic*

Description

Regridding using the "chain" method S4 Generic

Usage

```
regrid_rot(obj, ogrid, ...)

## S4 method for signature 'RasterLayer'
regrid_rot(obj, ogrid, ...)

## S4 method for signature 'RasterBrickTS'
regrid_rot(obj, ogrid, ...)
```

Arguments

obj	An object of class *raster* containing data to be regridded
ogrid	An object of class *raster* containing the output grid
...	Not Currnently used

Methods (by class)

- RasterLayer: For class "RasterLayer"
- RasterBrickTS: For class "RasterBrickTS"

Author(s)

Richard Cornes

`reverse_gamma` *Reverse Gamma Transform*

Description

Reverse the Gamma Transform used in gridding rainfall. This is a simple wrapper for `qgamma`

Usage

```
reverse_gamma(prob, shape, scale)
```

Arguments

`prob`	A vector of probabilities
`shape`	A vector of shape paramaters
`scale`	A vector of scale parameters

Value

Vector

Author(s)

Richard Cornes

`rgcv` *Gridding using the* `fitTps` *function*

Description

Produce Thin-Plate Spline interpolation using the `fitTps` function from the `rgcvpack` package. This produces a result very close to the original Fortran e-obs gridding code.

Usage

```
rgcv(obj, yvar = "VarSum^2", rhs = ~lon + lat + I(alt/1000), cost = 1,
  rev.trans = TRUE, cl = NULL, ...)
```

Arguments

`obj`	Object of class `eobs` or one of its derivatives
`yvar`	The left-hand side of the equation as a character string
`rhs`	The right-hand side of the equation as a `formula`
`cost`	Cost function supplied to `fitTps` function
`rev.trans`	If `TRUE` the values are converted back from a transformed state. For example, in the default case the values are returned as square-roots of the gridded values.
`cl`	A cluster object
`...`	Further parameters passed to the spline fitting function function

Value

An object of class `SpatialPointDataFrame`

Author(s)

Richard Cornes

See Also

`gridTPS`, `predict.Tps`, `fitTps`, `clusterSplit`, `foreach`

Other TPS.functions: `fields`, `gam.hybrid`, `gridTPS`, `lkrig`, `tprs`

`semiVG` *Variogram S4 Generic*

Description

Using the gstat 'variogram' function, produce the semi-varigram for each day of data and write the parameters to file. The intention is then that these daily semi-variograms are then compiled with the function `combine.vg`. This is the method that was originally used in the E-OBS dataset (see Haylock et al., 2008).

Usage

```
semiVG(obj, ...)

## S4 method for signature 'eobs'
semiVG(obj, form = anom ~ 1, Nlags = 20, cutoff = 500)

## S4 method for signature 'eobs_dtr'
semiVG(obj, form = anom ~ 1, Nlags = 20,
  cutoff = 500)

## S4 method for signature 'eobs_tmn'
```


```
semiVG(obj, form = anom ~ 1, Nlags = 20,
  cutoff = 500)

## S4 method for signature 'eobs_rain'
semiVG(obj, form = anom ~ 1, Nlags = 20,
  thresh = 0.5, cutoff = 450)
```

Arguments

obj	Object of class eobs
...	Parameters passed to respective functions
form	The formula that is passed to gstat::variogram
Nlags	Produces the width parameter in variogram such that width=cutoff/Nlags
cutoff	Seperation range
thresh	Value indicating a trace of rainfall

Value

NULL, but has the side effect of writing to file

Methods (by class)

- eobs: For class "eobs"
- eobs_dtr: For class "eobs_dtr"
- eobs_tmn: For class "eobs_tg"
- eobs_rain: For class "eobs_rain"

Author(s)

Richard Cornes

Examples

```
## Not run:
library(sp)
library(data.table)
library(raster)

data(NLgriddata)
data(NLStations)
data(NLhires)

get_VG <- function(idata){

   ## Add the station meta (including coordinates)
   idata <- meta[idata]

   ## Construct a SpatialPointsDataFrame
```


```
    coordinates(idata) <- ~LON+LAT
    proj4string(idata) <- CRS("+proj=longlat +ellps=WGS84")

    set_data(NL) <- idata

    semiVG(NL)
}

## Set the eobs class
NL <- startEOBS(c(3.27,7.43),c(50.9,53.8),c(1981,2010),c(1971,2014),1:12,var="rr")
hires <- rasterToPoints(NLhires,spatial=TRUE)
set_grid(NL) <- hires

## Get the station meta-data
meta <- data.table(NLStations)
meta[,c("COUNTRY_NAME","STATION_NAME"):=NULL]
setkey(meta,ID)

## Run the variogram over each day of the year 2000
day_data <- NLgriddata$day
setkey(day_data,id)
DT <- day_data[year==2000,get_VG(.SD),by=c("year","month","day")]

## Average the variograms
vg <- DT[,lapply(.SD,mean,na.rm=TRUE),
.SDcols=c("np","dist","gamma","dir.hor","dir.ver"),
by=c("variable","N")]

class(vg) <- "gstatVariogram"

## End(Not run)
```

`set_data<-` *set_data*

Description

This is a replacement function that takes a Spatial Points Dataframe consisting of the station data and assigns it to the `eobs` S4 object. If `lat_rng` and `lon_rng` have been specified in the `eobs` class then the station data will first be clipped to that extent. Following that the data are reprojected onto the same grid as the the high-resolution grid. For this reason `set_grid<-` must be called before this function. This function also averages stations within a radius that is set by the variable `distance` when calling `startEOBS`.

A replacement function to set the high-resolution grid to the `eobs` S4 class

Usage

```
set_data(obj) <- value
```


```
## S4 replacement method for signature 'eobs'
set_data(obj) <- value

set_grid(obj) <- value

## S4 replacement method for signature 'eobs'
set_grid(obj) <- value
```

Arguments

obj	Object of class 'eobs' or one of its derivatives
value	A spatial points dataframe that consists of the station data
value	A spatial points dataframe that consists of the high-resolution grid that will be interpolated to.

Note

The high-resolution interpolation grid must first be set with `set_grid<-`.

Author(s)

Richard Cornes

Richard Cornes

See Also

`startEOBS`, `SpatialPoints`.

`startEOBS`, `SpatialPoints`.

`startEOBS` *Initiate the eobs project*

Description

Initiate the 'eobs' S4 class

Usage

```
startEOBS(lon_rng, lat_rng, base, yrs, periods, distance = 1, var = c("rr",
  "tx", "tn", "dtr", "tg", "tmn", "pp"), ...)
```

Arguments

`lon_rng`	A vector of length two specifying the longitude extent
`lat_rng`	A vector of length two specifying the latitude extent
`base`	A vector of length two specifying the base period
`yrs`	A vector specifying the unique years to be gridded. Must be consecutive
`periods`	A vector of length 12 specifying the period configuration
`distance`	The distance over which over which station values are averaged
`var`	The name of the variable being gridded
`...`	Not currently used

Value

S4 object of class eobs or a derivative

Author(s)

Richard Cornes

Examples

```
## An E-OBS gridding class for rainfall across the Netherlands
NL <- startEOBS(c(3.27,7.43),c(50.9,53.8),c(1961,1990),1950:1999,1:12,var="rr")
```

`startEOBS.yaml` *eobs class from a yaml file*

Description

Construct an eobs class from a yaml file

Usage

```
startEOBS.yaml(yfile, ...)
```

Arguments

`yfile`	The path to the yaml file
`...`	Other options required by the `yaml.load_file` function

Details

This is a simple wrapper for `startEOBS`, which enables an eobs class to be constructed from a suitable yaml file.

Value

An object of class 'eobs' or one of its derivatives

Author(s)

Richard Cornes

See Also

`yaml.load_file`

`station_test` *station_test*

Usage

```
station_test(obj)

## S4 method for signature 'eobs'
station_test(obj)
```

Arguments

`obj`	Object of class 'eobs' or one of its derivatives

Methods (by class)

- `eobs`: For class eobs

Author(s)

Richard Cornes

`test.stns` *Select appropriate test stations*

Description

Select appropriate test stations

Usage

```
test.stns(eobs, cands, N = 100, homogFile = NULL, HomogSel = c("USEFUL",
  "DOUBTFUL", "SUSPECT"), thresh = 0.95, verbose = TRUE,
  save.knots = TRUE, ...)
```

Arguments

`eobs`	An object of class `eobs`
`cands`	A data.frame (or data.table) that contains the latitudes and longitudes of the candidate stations. These must be named columns in the form "lat" and "lon"
`N`	Number of stations to select. This is passed as the `nd` parameter in the function `cover.design` from the `fields` package
`homogFile`	Path to the file containing the homogeneity information
`HomogSel`	The homogeneity criterion
`thresh`	Completeness threshold
`verbose`	If `TRUE` print messages to screen
`save.knots`	If `TRUE` the `data.table` containing the selection of stations is written to an `RDS` file using `saveRDS`. The location of this file is as specified in the `knotFile` slot of the `eobs` class.
`...`	Further options passed to the `fields` function `cover.design`. The most important being `P`, `Q`, `nn`, `num.nn` and

Details

Choose N ECA stations that are distributed evenly across the domain. Other options allow the staitons to be selected based on a completeness criterion and/or which meet selected homogeneity characteristcs.

Value

`data.table` object with coordinates of the selected stations

Author(s)

Richard Cornes

Examples

```
## Not run:
eobs <- readRDS("eobs.RDS")
neweobs <- test.stns(eobs, "PREC_19512013__homogeneity.txt")
saveRDS(neweobs, file="neweobs.RDS")

## End(Not run)
```

`toDeci` *Convert a latitude/longitude string to decimal a value*

Description

The input string must be in the format degrees:minutes:seconds

Usage

```
toDeci(x)
```

Arguments

`x`	character string to be converted

Value

Coordinates in decimal degrees [numeric] to 2 decimal places. Since the input is in degrees, minutes, seconds, then this is a realistic precision, and is equivalent to a precision of 1.1132 km N/S & E/W at the equator, 1.0247 km E/W at 23N/S and 434.96 m E/W at 67N/S. See http://en.wikipedia.org/wiki/Decimal_degrees.

Author(s)

Richard Cornes

Examples

```
lat <- "52:20:27"
lat.dec <- toDeci(lat)
lon <- "01:25:00"
lon.dec <- toDeci(lon)
```

`tprs` *Gridding using MGCV*

Description

Gridding using MGCV

Usage

```
tprs(obj, yvar = "VarSum^2", rhs = ~s(lon, lat) + s(I(alt/1000)),
  cost = 1, rev.trans = TRUE, family = gaussian(), se.fit = FALSE,
  cl = NULL)
```

Arguments

`obj`	Object of class `eobs` or one of its derivatives
`yvar`	The left-hand side of the equation as a character string
`rhs`	The right-hand side of the equation as a `formula`
`cost`	Cost function supplied to `fitTps` function
`rev.trans`	If `TRUE` the values are converted back from a transformed state. For example, in the default case the values are returned as square-roots of the gridded values.
`family`	As specified for the `gam` function
`se.fit`	If `FALSE` standard error values are not calculated
`cl`	A cluster object

Value

An object of class `SpatialPointDataFrame`

Author(s)

Richard Cornes

See Also

`gridTPS`, `gam`

Other TPS.functions: `fields`, `gam.hybrid`, `gridTPS`, `lkrig`, `rgcv`

Index