

Package: climatol (via r-universe)

August 20, 2024

Version 4.1.0

Date 2024-03-23

Title Climate Tools (Series Homogenization and Derived Products)

Author Jose A. Guijarro <jaguijarro21@gmail.com>

Maintainer Jose A. Guijarro <jaguijarro21@gmail.com>

Depends R (>= 3.6)

Imports grDevices, graphics, stats, utils

Suggests evd, fields, gstat, maps, mapdata, ncd4, raster, readxl, RODBC, sp

Description Functions for the quality control, homogenization and missing data filling of climatological series and to obtain climatological summaries and grids from the results. Also functions to display wind-roses, meteograms, Walter&Lieth diagrams, and more.

License GPL (>= 3)

URL <https://climatol.eu>

NeedsCompilation no

Date/Publication 2024-03-22 23:00:02 UTC

Repository <https://jaggh.r-universe.dev>

RemoteUrl <https://github.com/cran/climatol>

RemoteRef HEAD

RemoteSha 39970f3f6f2c676d9c775c474efefe5404fd5487

Contents

climatol-internal	2
climatol2reclimindex	2
csv2climatol	4
dahgrid	5
dahstat	7

daily2climatol	9
Datasets	11
datrestore	12
datsubset	13
db2dat	14
dd2m	16
dens2Dplot	18
diagwl	19
exampleFiles	20
fix.sunshine	21
homogen	22
IDFcurves	27
meteogram	29
MHisopleths	30
outrename	31
QCthresholds	32
rclimdex2climatol	33
runtd	35
sef2climatol	36
windrose	37
xls2csv	38

Index **40**

climatol-internal *Internal climatol functions*

Description

Internal climatol functions

Details

These functions are used internally and are not intended to be called directly by the user.

climatol2rclimdex *Convert DAILY data from climatol to RCLimDex input format*

Description

This function reads homogenized daily series of precipitation (RR) and extreme temperatures (TX, TN), adjusted from the last homogeneous sub-period, and writes them in files (one per station) with RCLimDex format.

Usage

```
climatol2rclimdex(varRR, varTX, varTN, yiRR, yfRR, yiTX=yiRR, yfTX=yfRR,
  yiTN=yiRR, yfTN=yfRR, header=TRUE, prefix='hoc1m', dir=NA, na='-99.9')
```

Arguments

varRR, varTX, varTN	Name of the variables in the climatol destination files. If some variable is not available, name it as "".
yiRR, yfRR	Initial and final years of the homogenized RR series.
yiTX, yfTX, yiTN, yfTN	Initial and final years of the TX and TN series. (The same as yiRR and yfRR by default.)
header	include a header in the files? (TRUE by default)
prefix	Prefix to prepend to station codes to name the output RCLimDex files.
dir	Destination directory of the output RCLimDex files. (If not set, they will be saved into the current R working directory).
na	Missing data code to use in the output files. ('-99.9' by default.)

Details

After homogenizing daily series with `climatol`, the user may be interested in applying the `RCLimDex` program to the homogenized series. This function automatizes the conversion of file formats between both programs.

Note that if there are some days with $TX < TN$ (can happen because of the independent homogenization of extreme temperatures), a trivial fix will be applied by just exchanging the problem values.

See Also

[homogen](#)

Examples

```
## Set a temporal working directory and generate input files:
wd <- tempdir()
wd0 <- setwd(wd)

## copy example daily RR, TX and TN homogenization results:
file.copy(exampleFiles('RR_1981-1995.rda'), '.')
file.copy(exampleFiles('TX_1981-1995.rda'), '.')
file.copy(exampleFiles('TN_1981-1995.rda'), '.')

## Now run the example:
climatol2rclimdex('RR', 'TX', 'TN', 1981, 1995)

## Return to user's working directory:
setwd(wd0)
```

```
## Input and output files can be found in directory:
print(wd)
```

csv2climatol *Convert data in a single CSV file to climatol input format*

Description

This function helps to prepare the climatol input files when the users have their data in a single CSV file, as the output of xls2csv().

Usage

```
csv2climatol(csvfile, datacol=6:8, stnfile=csvfile, stncol=1:5, varcli,
anyi=NA, anyf=NA, mindat=NA, sep=',', dec='.', na.strings='NA',
dateformat='%Y-%m-%d', cf=1, ndec=1, header=TRUE)
```

Arguments

csvfile	name of the CSV file containing the data.
datacol	column(s) holding station codes, dates and data.
stnfile	name of the CSV file containing station codes, names and coordinates (if these data are not in the csvfile).
stncol	columns holding longitudes, latitudes, elevations and station codes and names.
varcli	short name of the climatic variable under study.
anyi	first year to study.
anyf	last year to study.
mindat	minimum required number of data per station (by default, 60 monthly data or 365 daily data).
sep	data separator (',' by default: Comma Separated Values).
dec	decimal point ('.' by default).
na.strings	strings coding missing data ('NA' by default).
dateformat	format of dates (if not in separate columns. Default: '%Y-%m-%d')
cf	conversion factor to apply if data units need to be changed.
ndec	no. of decimals to round to.
header	TRUE by default, set to FALSE if csvfile has no header.

Details

If datacol holds 4 (or 5) values, dates are expected to appear as year, month (and days) in separate columns. Otherwise, dates will be provided as character strings (see parameter dateformat). Station codes, names and coordinates can go in a separate file stnfile. At least coordinates and station codes must be present in either csvfile or stnfile. Put a zero for any inexistent columns. Example when stnfile contains only, in this order, latitudes, longitudes and station names: stncol=c(2,1,0,3,0). Note that if a stnfile is provided, then sep, dec, na.strings and header defined for csvfile will also be applied to stnfile.

See Also

[xls2csv](#), [homogen](#)

Examples

```
## Set a temporal working directory:
wd <- tempdir()
wd0 <- setwd(wd)

## Create origin and destination directories and copy example input files:
dir.create('dir1'); dir.create('dir2')
file.copy(exampleFiles('p064.xlsx'),'dir1')
file.copy(exampleFiles('p082.xlsx'),'dir1')
file.copy(exampleFiles('p084.xlsx'),'dir1')

## Create input files for csv2climatol with the function xls2csv:
xls2csv('dir1','dir2','RR')

## Add bogus coordinates and elevations to the station file:
est=read.table('xls_RR_stations.csv',sep=',')
est=data.frame(1:3,21:23,101:103,est)
write.table(est,'xls_RR_stations.csv',sep=',',row.names=FALSE,col.names=FALSE)

## Now run the example of csv2climatol:
csv2climatol('xls_RR_data.csv', datacol=1:5, stnfile='xls_RR_stations.csv',
  varcli='RR',header=FALSE)

## Return to user's working directory:
setwd(wd0)

## Input and output files can be found in directory:
print(wd)
```

dahgrid

Interpolation of normalized homogeneous data on a predefined grid

Description

Homogenized data generated by [homogen](#) are normalized and interpolated on a grid provided by the user at every time step, and saved into a NetCDF file.

Usage

```
dahgrid(varcli, anyi, anyf, anyip=anyi, anyfp=anyf, grid, idp=2.0,
obsonly=TRUE, nmax=Inf)
```

Arguments

<code>varcli</code>	Short name of the studied climatic variable, as in the data file name.
<code>anyi</code>	Initial year of the homogenized data.
<code>anyf</code>	Final year of the homogenized data.
<code>anyip</code>	First year of the desired reference period. (The reference period defaults to the whole period of the data).
<code>anyfp</code>	Last year of the desired reference period.
<code>grid</code>	Grid on which interpolations must be performed.
<code>idp</code>	Power of the inverse distance weights (2 by default).
<code>obsonly</code>	Do not interpolate estimated missing data. (TRUE by default).
<code>nmax</code>	Maximum number of nearest stations to use (all by default).

Details

Homogenized data are read from the binary file ‘VRB_ANYI-ANYF.rda’ generated by [homogen](#). Only series reconstructed from their longest homogeneous sub-period are retained, and they are normalized by their means (and standard deviations, if `std=3`), computed for the selected reference period (or for the whole period of the data, by default).

Unless `obsonly` is set to `FALSE`, data that were missing in the observed series are deleted to avoid interpolation of already interpolated data.

Finally, the normalized homogeneous data are interpolated on the predefined grid for every time step using an inverse distance weight method, and the resulting grids are stored in a NetCDF file named ‘VRB_ANYIP-ANYFP.nc’, including grids of the reference means (and standard deviations, if applied).

The user must provide the grid as an object of class `SpatialPixel`, as in this example defining a grid from 40N,3E to 43N,7E with a resolution of 0.1 degrees:

```
grid <- expand.grid(x=seq(3,7,.1),y=seq(40,43,.1))
library(sp)
coordinates(grid) <- ~ x+y
```

The resolution of this grid need not be too high, but adjusted to the spatial density of the available series. However, a higher resolution will produce smoother maps when plotted.

The user may be more interested in obtaining grids of absolute values, rather than normalized. This can be achieved simply by undoing the normalization on the grids with the help of the provided grids of reference means and standard deviations. However, the resulting grids will only be the product of a geometrical interpolation, and will not reflect the influence of orography and other physiographic effects on the studied climatic variable. Therefore, it is more advisable to derive better reference grids of means (and standard deviations, if needed) by applying a geostatistical model to the reference means (provided in the file ‘VRB_ANYIP-ANYFP_means.csv’ with their corresponding coordinates).

This better quality climatic maps will probably have a higher resolution than that of the grids of the NetCDF file provided by this function. In that case, these normalized grids must be interpolated to the grid of the geostatistically derived climatic maps before undoing the normalization to obtain the final maps of absolute values at all or selected time-steps of the studied period.

See Also[homogen](#)**Examples**

```
## Set a temporal working directory and write input files:
wd <- tempdir()
wd0 <- setwd(wd)

## Copy an example file of homogenization results:
file.copy(exampleFiles('Temp_1991-2000.rda'),'.')

## Now run the example:
## (very coarse grid (3x2 points) to run in less than the 10 seconds CRAN limit)
grd <- expand.grid(x=seq(-2.8,-2.4,.2),y=seq(38.86,39.06,.2))
sp::coordinates(grd) <- ~ x+y
dahgrid('Temp',1991,2000,grid=grd)

## Return to user's working directory:
setwd(wd0)

## Input and output files can be found in directory:
print(wd)
```

dahstat

*Extract series or statistics of the homogenized data***Description**

Lists series, means, medians, standard deviations, quantiles or trends, for a specified period, from series homogenized by [homogen](#).

Usage

```
dahstat(varcli, anyi, anyf, anyip=anyi, anyfp=anyf, stat="me", ndc=NA, vala=2,
valm=vala, cod=NULL, prob=.5, all=FALSE, long=FALSE, relref=FALSE, pernyr=10,
estcol=c(1,2,4), sep=',', dec='.')
```

Arguments

varcli	Short name of the studied climatic variable, as in the data file name.
anyi	Initial year of the homogenized period.
anyf	Final year of the homogenized period.
anyip	First year of the period to analyze. (Defaults to anyi).
anyfp	Last year of the period to analyze. (Defaults to anyf).
stat	Statistical parameter to compute for the selected period:

	<p>"me": Means (default),</p> <p>"mdn" Medians,</p> <p>"max" Maxima,</p> <p>"min" Minima,</p> <p>"std" Standard deviations,</p> <p>"q" Quantiles (see the prob parameter),</p> <p>"tnd" OLS trends and their p-values,</p> <p>"series" Do not compute any statistic; only write homogenized series and flags into two CSV files.</p> <p>"mseries" As before, but output series in individual *.csv files. (Not applicable to daily series.)</p>
ndc	Number of decimal places to be saved in the output file (defaults to that used in the homogenization).
vala	<p>Annual values to compute from the sub-annual data:</p> <p>0: None,</p> <p>1: Sum,</p> <p>2: Mean (default),</p> <p>3: Maximum,</p> <p>4: Minimum.</p>
valm	<p>Monthly values to calculate from sub-monthly data (defaults to vala):</p> <p>1: Sum,</p> <p>2: Mean,</p> <p>3: Maximum,</p> <p>4: Minimum.</p>
cod	Vector of requested station codes (all by default).
prob	Probability for the computation of quantiles (0.5 by default, i.e., medians). You can set probabilities with more than 2 decimals, but the name of the output file will be identified with the rounded percentile.
all	If TRUE, all reconstructed series will be used. The default is FALSE, hence using only the series reconstructed from the last homogeneous subperiod.
long	If TRUE (the default is FALSE), only series reconstructed from the longest homogeneous subperiod will be used.
relref	If TRUE, statistics from reliable reference series will also be listed. (FALSE by default).
pernyr	Number of years on which to express trend units (10 by default).
estcol	Columns of the homogenized stations file to be included in the output file. (Defaults to c(1,2,4), the columns of station coordinates and codes).
sep	Field separator (',' by default).
dec	Decimal point ('.' by default).

Details

Homogenized data are read from the file 'VRB_ANYI-ANYF.rda' saved by [homogen](#), while this function saves the computed data for the specified period in 'VRB_ANYIP-ANYFP.STAT', where STAT is substituted by the `stat` requested statistic. An exception is when `stat="q"`, since then the extension of the output file will be `qPP`, where PP stands for the specified prob probability (in percent).

The output period ANYIP-ANYFP must of course be comprised within the period of the input data, ANYI-ANYF.

`stat='tnd'` computes trends by Ordinary Least Squares linear regression on time, listing them in a CSV file '`*_tnd.csv`' and their p-values in '`*_pval.csv`'

If `stat='series'` is chosen, two text files in CSV format will be produced for every station, one with the data and another with their flags: 0 for original, 1 for infilled and 2 for corrected data.

See Also

[homogen](#), [dahgrid](#).

Examples

```
## Set a temporal working directory:
wd <- tempdir()
wd0 <- setwd(wd)

## Copy an example file of homogenization results:
file.copy(exampleFiles('Temp_1991-2000.rda'), '.')

## Now run the examples:
dahstat('Temp', 1991, 2000)
dahstat('Temp', 1991, 2000, stat='q', prob=0.4)
dahstat('Temp', 1991, 2000, stat='tnd')
dahstat('Temp', 1991, 2000, stat='series')

## Return to user's working directory:
setwd(wd0)

## Input and output files can be found in directory:
print(wd)
```

Description

This function can be useful to prepare the `climatol` input files when the users have their daily data in per station individual files.

Usage

```
daily2climatol(stfile, stcol=1:6, datcol=1:4, varcli, anyi=NA, anyf=NA,
  mindat=365, sep=',', dec='.', na.strings='NA', dateformat='%Y-%m-%d',
  header=TRUE)
```

Arguments

stfile	File with file names and station coordinates, codes and names.
stcol	Columns in stfile holding data file names, longitudes, latitudes, elevations and station codes and names. (Defaults to 1:6. Use 0 for codes and/or names columns if they are missing, and numeric values will be assigned.)
datcol	Columns in data files holding year, month, day, value.
varcli	Short name of the studied climatic variable.
anyi	First year to study (defaults to the first year of available data).
anyf	Last year to study (defaults to the last year of available data).
mindat	Minimum required number of data per station. (Defaults to 365 daily data.)
sep	Field separator in all files, whether data or stations. (',' by default.)
dec	Decimal point. ('.' by default.)
na.strings	Strings coding missing data ('NA' by default).
dateformat	Format of dates if not in separate columns. ('%Y-%m-%d' by default.)
header	Logical value indicating whether input files have a header line or not. (TRUE by default.)

Details

Many users have their daily series in separate files (one per station). This function can be used to read these daily data files and write the input files needed by the homogen function of this climatol package.

When either station codes or names are missing in the stations file, its corresponding column must be set to 0. In this case, codes and/or names will be assigned with numeric values.

Field separator, decimal point and the presence of a header line must be consistent in all files (data files and stations file).

If your files follow the RClimDex convention, you can use the rclimdex2climatol function instead.

See Also

[rclimdex2climatol](#), [homogen](#)

Examples

```
## Set a temporal working directory and write example input files:
wd <- tempdir()
wd0 <- setwd(wd)
data(climatol_data)
```

```

df=cbind(File=c('p064.csv', 'p084.csv', 'p082.csv'), S1stations)
write.csv(df, 'stations.csv', row.names=FALSE, quote=FALSE)
write.csv(p064.df, 'p064.csv', row.names=FALSE, quote=FALSE)
write.csv(p084.df, 'p084.csv', row.names=FALSE, quote=FALSE)
write.csv(p082.df, 'p082.csv', row.names=FALSE, quote=FALSE)

## Now run the example:
daily2climatol(stfile='stations.csv', varcli='RR')

## Return to user's working directory:
setwd(wd0)

## Input and output files can be found in directory:
print(wd)

```

 Datasets

Data sets to run examples of the functions in the climatol package.

Description

This object contains several small datasets needed to run the examples of most functions.

Usage

```
data(climatol_data)
```

Details

This data set holds the following collection of data objects:

TX3st Maximum daily temperature of 3 stations during 1981-1995.

TN3st Minimum daily temperature of 3 stations during 1981-1995.

RR3st Daily precipitation of 3 stations during 1981-1995.

S1stations Stations coordinates, codes and names of the *3st data.

p064.df Data frame with RR, TX and TN data for station p064.

p084.df Data frame with RR, TX and TN data for station p084.

p082.df Data frame with RR, TX and TN data for station p082.

AWS_1year Hourly data from an Automatic Weather Station during one year.

AWS_1day 10 minutes data from an Automatic Weather Station during one day.

datcli Monthly climatic parameters to plot a Walter&Lieth diagram.

Temp.dat Monthly temperature of five stations during 1961-2005.

Temp.est Stations coordinates, codes and names of the Temp.dat data.

Tav Annual average temperature at Oslo (Norway) during 1901-2020.

prec10min Ten minutes precipitation data during 1991-2020.

Some examples need the use of files rather than these data objects. In that case they are provided in a special folder of the installation directory tree and are made accessible through the function [exampleFiles](#).

Source

RR3st, TX3st, TN3st, p064.df, p082.df and p084.df data were obtained from the historical run (1950-2005) of the Regional Atmospheric Climate Model version 2 of the Royal Netherlands Meteorological Institute (KNMI) in the frame of the INDECIS project <<https://indecis.eu>>.

Oslo annual average temperatures Tav were downloaded from the HCLIM database.

The other objects contain real data, but anonymized to avoid data policy restrictions.

References

Lundstad, Elin; Brugnara, Yuri; Broennimann, Stefan (2022): Global Early Instrumental Monthly Meteorological Multivariable Database (HCLIM). <https://doi.org/10.1594/PANGAEA.940724>

See Also

[exampleFiles](#)

Examples

```
data(climatol_data)
datcli
head(p064.df)
head(AWS_1year)
```

datrestore

Rename homogen's output files

Description

This function restores some deleted outliers into the dah matrix of the *.rda output file.

Usage

```
datrestore(varcli, anyi, anyf, QCout=FALSE)
```

Arguments

varcli	Short name of the studied climatic variable, as in the data file name.
anyi	Initial year of the study period.
anyf	Final year of the study period.
QCout	Set this parameter to TRUE to read the selected outliers to be restored from the *-QC*_out.csv output by a previous run of homogen(..., onlyQC=TRUE).

Details

When the user checks the list of outliers in the output file *_out.csv, true extreme values (or sequences of identical values) that have been deleted can be restored by changing their deleted field to negative. This accepted values will be restored in the dah matrix of homogenized series contained in the *.rda file output by the homogen function, but only in the series reconstructed from the last homogeneous subperiod.

See Also

[homogen](#)

Examples

```
## Set a temporal working directory, write input files and homogenize them:
wd <- tempdir()
wd0 <- setwd(wd)
data(climatol_data)
write.table(Temp.est, 'Temp_1961-2005.est', row.names=FALSE, col.names=FALSE)
write(Temp.dat, 'Temp_1961-2005.dat')
homogen('Temp', 1961, 2005) #obtain homogenization output files
out <- read.csv('Temp_1961-2005_out.csv') #read list of outliers
## Change the sign of a couple of deleted values to be restored:
out[2,6] <- -1; out[6,6] <- -9
write.csv(out, 'Temp_1961-2005_out.csv', row.names=FALSE)

## Now run the example:
datrestore('Temp', 1961, 2005) #restore the selected values

## Return to user's working directory:
setwd(wd0)

## Input and output files can be found in directory:
print(wd)
```

datsubset

Subset data by subperiod, code list or no. of years with data

Description

This function allows saving a subset of climatol input data into new input files by selecting a subperiod, a minimum number of years with data and/or a group of stations.

Usage

```
datsubset(varcli, anyi, anyf, anyis=anyi, anyfs=anyf, minny=NA, codes=NULL,
na.strings=NA, ini=NA)
```

Arguments

<code>varcli</code>	Short name of the studied climatic variable.
<code>anyi</code>	Initial year of the data present in the file.
<code>anyf</code>	Final year of the data present in the file.
<code>anyis</code>	First year of the output subperiod. (Defaults to <code>anyi</code>).
<code>anyfs</code>	Last year of the output subperiod. (Defaults to <code>anyf</code>).
<code>minny</code>	Minimum number of years with data to retain the series.
<code>codes</code>	Vector of chosen station codes. (Defaults to NULL, meaning all).
<code>na.strings</code>	Strings marking missing data (NA by default).
<code>ini</code>	Initial date (if not January 1st).

Details

Homogenization by `climatol` requires that no time step be totally void of data in all stations simultaneously. This function allows subsetting already existing `climatol` input files by selecting a subperiod and/or stations with a minimum number of years with data (may contain gaps).

Another possibility is to choose a group of stations, useful when the initial cluster analysis reveals areas with different climate regimes that should be homogenized independently.

Examples

```
## Set a temporal working directory and write input files:
wd <- tempdir()
wd0 <- setwd(wd)
data(climatol_data)
write.table(Temp.est, 'Temp_1961-2005.est', row.names=FALSE, col.names=FALSE)
write(Temp.dat, 'Temp_1961-2005.dat', ncolumns=12)

## Now run the examples:
datsubset('Temp', 1961, 2005, 1971, 2000, minny=20)
datsubset('Temp', 1971, 2000, codes=c('st02', 'st03'))

## Return to user's working directory:
setwd(wd0)

## Input and output files can be found in directory:
print(wd)
```

db2dat

*Get daily or monthly data from a database and build input files *.dat and *.est*

Description

This function facilitates the creation of the input files needed by this package by retrieving the data from a database through an RODBC connection.

Usage

```
db2dat(varcli, anyi, anyf, minny=5, daily=TRUE, ch, dformat='%Y-%m-%d',
vtable, vcode, vdate, vval, stable, scode, sname, sx, sy, sz)
```

Arguments

varcli	Short name of the studied climatic variable, as it will appear in all data file names.
anyi	Initial year of the data to be included in the file.
anyf	Final year of the data to be included in the file.
minny	Minimum number of years with data for a series to be included in the file.
daily	Logical flag indicating whether the data are daily (the default) or monthly (set daily=FALSE in this case).
ch	Already open ODBC connection to the climatic database.
dformat	Date format in the database.
vtable	Name of the table containing our climatic variable.
vcode	Name of the variable containing station codes in the vtable table.
vdate	Name of the variable containing dates in the vtable table.
vval	Name of the climatic variable in the vtable table.
stable	Name of the table containing station information (metadata).
scode	Name of the variable containing station codes in the table stable.
sname	Name of the variable containing station names in the stable table.
sx	Name of the variable containing longitudes (degrees with decimals!) in the stable table.
sy	Name of the variable containing latitudes (degrees with decimals!) in the stable table.
sz	Name of the variable containing elevations (meters) in the stable table.

Details

This function creates the two input files needed by the homogenization functions of this package, 'VRB_YEAR-YEAR.dat' (holding the data) and 'VRB_YEAR-YEAR.est' (holding station coordinates, codes and names).

The table in the accessed database must contain either daily or monthly data (set daily=FALSE in this case). Otherwise the number of data per series will not be match the expected value and the function will fail.

Moreover, every data item must be in a different record in the database, as in this example table of monthly data (different variables for the same time step are O.K.):

```
Station Date T.max T.min Rel.Hum Precip Wind.speed
S032 1991-01-01 12.1 -2.1 59 128.2 5.4
S032 1991-02-01 13.2 -2.5 62 78.4 6.2
...
```

But if the table in the database arranges all monthly values of one year (or all daily values of one month) in a single record, then this function cannot be applied. In this cases, try to use the database functionalities to output series into CSV files and apply other conversion functions as `csv2climatol`.

See Also

[homogen](#), [csv2climatol](#)

Examples

```
## Not run:
## First we must access our climatic database through RODBC, wich requires to
## have this package installed. System programs that allow ODBC connections to
## databases must also be installed and properly configured.

## For this example we will assume that our database is named "climate" and we
## access it with user "USER" and password "PASS". Then we open the connection
## with:
library(RODBC)
ch <- odbcConnect("climate",uid="USER",pwd="PASS")

## Now we want to use this function to gather all monthly relative humidity
## averages for the period 1961-2015, requiring a minimum of 10 years of data
## (not necessarily consecutive). We must use the corresponding names of tables
## and headers existing the the database, and putting the parameters in the
## required order we avoid the need to name them:
db2dat('HRel',1961,2015,10,FALSE,ch,'%Y-%m-%d','monthly_relhum',
'Station','Date','Value','stations','Station','Name','Longitude',
'Latitude','Elevation')

odbcClose(ch) #close the connection if you do not need it anymore

## Our data would now be ready to be homogenized with the homogen function:
homogen('HRel',1961,2015,vmin=0,vmax=100)

## End(Not run)
```

dd2m

Compute monthly data from daily (or subdaily) series

Description

Daily or sub-daily series are aggregated into total, mean, maximum, or minimum monthly values, and saved to files in `climatol` input format.

Usage

```
dd2m(varcli, anyi, anyf, ndec=1, valm=2, namax=30, x=NULL, na.strings="NA",
tz='utc')
```


Arguments

varcli	Short name of the studied climatic variable, as in the data file name.
anyi	Initial year of the data present in the file.
anyf	Final year of the data present in the file.
ndec	Number of decimal places to be saved in the output file.
valm	Monthly value to compute: 1: Sum, 2: Mean, 3: Maximum, 4: Minimum, 5: Standard deviation.
namax	Maximum percentage of missing data in any month to compute its monthly value. (30 by default)
x	Time vector. If not provided, it will be built as dates (or date-time for sub-daily data) beginning January 1st of the initial year. The user must provide if data are taken at irregular intervals or are not beginning the first of January.
na.strings	Missing data code in the original daily data. (NA by default.)
tz	Time zone ('utc' by default). Only relevant for subdaily data.

Details

Data are read from files 'VRB_YEAR-YEAR.dat' and 'VRB_YEAR-YEAR.est', and output monthly data will be saved to files with the same names but with the suffix -m appended to the name of the variable.

See Also

[homogen](#), [dahstat](#), [dahgrid](#)

Examples

```
## Set a temporal working directory and write input files:
wd <- tempdir()
wd0 <- setwd(wd)
data(climatol_data)
write.table(SIstations, 'RR_1981-1995.est', row.names=FALSE, col.names=FALSE)
write(as.matrix(RR3st[, 2:4]), 'RR_1981-1995.dat')

## Now run the example:
dd2m('RR', 1981, 1995, valm=1)

## Return to user's working directory:
setwd(wd0)

## Input and output files can be found in directory:
print(wd)
```

dens2Dplot	<i>Two dimensional density plot</i>
------------	-------------------------------------

Description

This function generates a scatter plot enhancing density with different colors.

Usage

```
dens2Dplot(x, y, nbins=100, pal=NULL, xlab='', ylab='', xlim=c(NA,NA),  
ylim=c(NA,NA), ...)
```

Arguments

x, y	Variables for the scatter plot.
nbins	Number of bins in X and Y coordinates of the scatter plot.
pal	Color palette
xlab, ylab	Labels for X and Y axis
xlim, ylim	Limits for X and Y axis
...	Other graphic parameters

Details

This function has been inspired by Elzizi's answer at <http://stackoverflow.com/questions/18089752/r-generate-2d-histogram-from-raw-data> The user can add a grid, title and other details to the scatter plot.

Examples

```
n=1000000; x=rnorm(n,15,4); y=x+rnorm(n,5,6)  
dens2Dplot(x,y,xlab='Variable X',ylab='Variable Y',las=1)  
  
## Let's add a grid and a title:  
grid(col=grey(.4))  
title('Example of dens2Dplot')
```

diagwl	<i>Walter & Lieth climatic diagram</i>
--------	--------------------------------------------

Description

Plot a Walter & Lieth climatic diagram of a station.

Usage

```
diagwl(dat, cols=1:6, format='%Y-%m-%d', yeari=NA, yearf=NA,
       stname='', alt=NA, per='', mlab='', shem=NULL, p3line=FALSE, ...)
```

Arguments

<code>dat</code>	Data frame with the required climatic data (see details).
<code>cols</code>	Columns containing dates and daily data of precipitation and extreme temperatures. Set to NULL if a monthly climate summary is provided.
<code>format</code>	Format of the dates if data are provided in 4 columns [<code>'%Y-%m-%d'</code>].
<code>yeari, yearf</code>	Initial and final years of the period to use. (Defaults to the period contained in <code>dat</code> , but if it contains a climate summary, then the parameter <code>per</code> should be supplied (see below).
<code>stname</code>	Name of the climatological station.
<code>alt</code>	Elevation (altitude) of the climatological station.
<code>per</code>	If data is a data frame with already calculated climate averages, the original period of the data.
<code>mlab</code>	Vector of 12 monthly labels for the X axis (see the details).
<code>shem</code>	Southern hemisphere? NULL by default, to be detected from warm season. Set to TRUE or FALSE to force southern or northern hemisphere.
<code>p3line</code>	Draw a supplementary precipitation line referenced to three times the temperature? (FALSE by default.)
<code>...</code>	Other optional graphic parameters.

Details

The data frame can contain daily data of precipitation and extreme temperatures or 12 columns with pre-calculated monthly climate parameters.

In the latter case, the monthly values from January to December must be in the 12 first columns (any additional trailing columns will be disregarded) and four rows, in the following order:

Row 1: Mean total precipitation

Row 2: Mean maximum daily temperature

Row 3: Mean minimum daily temperature

Row 4: Absolute minimum daily temperature

This last row is used only to determine the probable frost months (when absolute monthly minimums are equal or lower than 0 C).

Alternatively, if series of daily data of precipitation and extreme temperatures are provided, dates can be given in three separate columns (year, month, day) or in a single column with the specified format ('%Y-%m-%d' by default).

cols indicate in which columns are located the dates and climatic data. By default they are expected in columns 1 to 3 for year, month and day, and columns 4 to 6 for precipitation, maximum and minimum temperature respectively.)

mLab is the vector for the 12 monthly labels, but it may be set to just 'en' or 'es' to use the first letter of month names in English or Spanish respectively.

As described by Walter and Lieth, when monthly precipitation is greater than 100 mm, the scale is increased from 2 mm/C to 20 mm/C to avoid too high diagrams in very wet locations. This change is indicated by a black horizontal line, and the graph over it is filled in solid blue.

When the precipitation graph lies under the temperature graph ($P < 2T$) we have an arid period (filled in dotted red vertical lines). Otherwise the period is considered wet (filled in blue lines), unless p3line=TRUE, that draws a precipitation black line with a scale $P = 3T$; in this case the period in which $3T > P > 2T$ is considered semi-arid. (Parameter p3line was suggested by Bogdan Rosca.)

Daily maximum average temperature of the hottest month and daily minimum average temperature of the coldest month are frequently used in vegetation studies, and are labeled in black at the left margin of the diagram.

References

Walter H & Lieth H (1960): Klimadiagramm Weltatlas. G. Fischer, Jena.

Examples

```
data(climatol_data)

## from pre-calculated monthly climatic data:
diagwl(datcli,cols=NULL,est="My Airport",alt=100,per="1961-90",mlab="en")

## from daily series of precipitation and extreme temperatures:
diagwl(p064.df, stname="Cold Place", alt=100, mlab="en")

## idem limiting the period to calculate monthly values:
diagwl(p064.df, yearf=1990, stname="Cold Place", alt=100, mlab="en")
```

exampleFiles

Get the path to some example files

Description

This function provides the path to files needed to run examples of some functions of the climatol package.

Usage

```
exampleFiles(file=NULL)
```

Arguments

file Name of the needed file. If NULL, all example files will be listed.

Details

This function is an adaptation of `readxl_example`, of the `readxl` package.

Examples

```
exampleFiles()  
exampleFiles('Temp_1991-2000.rda')
```

fix.sunshine	<i>Check homogenized daily sunshine hours and prune any excess</i>
--------------	--------------------------------------------------------------------

Description

This function loads homogenization results of daily sunshine series and prunes any excess over maximum theoretical sunshine duration.

Usage

```
fix.sunshine(varcli, anyi, anyf)
```

Arguments

varcli Short name of the homogenized climatic variable.
anyi First year of the homogenized series.
anyf Last year of the homogenized series.

Details

Any modified value is listed to the console and written to `fix.sunshine.txt`

The original `*.rda` file is saved as `*.rda.bak` and a new `*.rda` file is written with the fixed sunshine values.

See Also

[homogen](#)

Examples

```
## Set a temporal working directory:
wd <- tempdir()
wd0 <- setwd(wd)

## copy example daily sunshine homogenization results:
file.copy(exampleFiles('SS_1991-2000.rda'),'.')

## Now run the example:
fix.sunshine('SS',1991,2000)

## Return to user's working directory:
setwd(wd0)

## Input and output files can be found in the directory:
print(wd)
```

homogen

Automatic homogenization of climatological series

Description

Automatic homogenization of climatological series, including missing data filling and detection and correction of outliers and shifts in the mean of the series.

Usage

```
homogen(varcli, anyi, anyf, test='snht', nref=NULL, std=NA, swa=NA,
ndec=1, niqd=c(4,1), dz.max=.01, dz.min=-dz.max, cumc=NA, wd=NULL, inht=25,
sts=5, maxdif=NA, maxite=999, force=FALSE, wz=.001, mindat=NA, onlyQC=FALSE,
annual=c('mean','sum','total'), x=NULL, ini=NA, na.strings="NA", vmin=NA,
vmax=NA, hc.method='ward.D2', nclust=300, cutlev=NA, grdcol=grey(.4),
mapcol=grey(.4), expl=FALSE, metad=FALSE, sufbrk='m', tinc=NA, tz='utc',
rlemin=NA, rlemax=NA, cex=1.1, uni=NA, raway=TRUE, graphics=TRUE, verb=TRUE,
logf=TRUE, snht1=NA, snht2=NA, gp=NA)
```

Arguments

varcli	Short name of the studied climatic variable.
anyi	Initial year of the data.
anyf	Final year of the data.
test	Inhomogeneity test to apply: 'snht' (the default) or 'cuct' (Cucconi test, experimental).
nref	Maximum number of references for data estimation [defaults to 10 in the detection stages, and to 4 in the final series adjustments].
std	Type of normalization:

	<p>1: deviations from the mean, 2: rates to the mean (only for means greater than 1), 3: standardization (subtract the mean and divide by the sample standard deviation).</p>
swa	Size of the step forward to be applied to the overlapping window application of the detection test [365 terms (one year) for daily data, and 60 otherwise].
ndec	Number of decimal digits to round the homogenized data [1].
niqd	Number of interquartile distances to delete big outliers [4] and too long runs of identical values [1]. [Defaults to $c(4, 1)$]
dz.max	Threshold of outlier tolerance, in standard deviations if greater than one, or as a percentage of data to reject otherwise [0.01].
dz.min	Lower threshold of outlier tolerance if different from dz.max.
cumc	Code of accumulated missing data.
wd	Distance (in km) at which reference data will weight half of those located at zero distance [$c(0, 0, 100)$].
inht	Thresholds for the change in the mean detection tests [25].
sts	Series tail size (no. of terms) not tested for inhomogeneities [5].
maxdif	Maximum data difference from previous iteration [$ndec/2$].
maxite	Maximum number of iterations to compute means of the series [999].
force	Force direct homogenization of daily or sub-daily series [FALSE].
wz	Scale parameter of the vertical coordinate Z [0.001].
mindat	Minimum number of data for a split fragment to become a new series [swa/2 for daily series or 12 terms otherwise].
onlyQC	Set to TRUE if only initial Quality Controls are requested [FALSE]
annual	Running annual value to graph in the PDF output. One of 'mean' (the default), 'sum' or 'total' (equivalent to 'sum').
x	Time vector. Only needed if data are taken at irregular intervals.
ini	Initial date, with format 'YYYY-MM-DD', if series does not begin on January first (as recommended).
na.strings	Character strings to be treated as missing data ['NA'].
vmin	Minimum possible value (lower limit) of the studied variable.
vmax	Maximum possible value (upper limit) of the studied variable.
hc.method	Hierarchical clustering method ['ward.D2'].
nclust	Maximum number of series for the cluster analysis [300].
cutlev	Level to cut the dendrogram to define clusters [NA].
grdcol	Color of the graphic background grids [$grey(0.04)$].
mapcol	Color of coastlines and borders in the stations map [$grey(0.04)$].
expl	Perform an exploratory analysis? [FALSE].
metad	Use the breakpoints file as metadata? [FALSE].

sufbrk	Suffix to add to varcli to form the name of the provided metadata file ['m'].
tinc	Time increment between data [NA].
tz	Time zone ['utc']. Only relevant for subdaily data.
rlemin	Data run lengths will exclude values <= rlemin in quality control [NA].
rlemax	Data run lengths will exclude values >= rlemax in quality control [NA].
cex	Character expansion factor for graphic labels and titles [1.1].
uni	Units to use in some axis labels ["].
raway	Increase internal distances to reanalysis series to give more weight to observed series [TRUE].
graphics	Output graphics to a PDF file [TRUE].
verb	Verbosity [TRUE].
logf	Save console messages to a log file? [TRUE].
snht1, snht2	Obsolete (use inht instead), but kept for backwards compatibility.
gp	Obsolete (use graphics=FALSE for gp=0, onlyQC=TRUE for gp=1 or annual="total" for gp=4), but kept for backwards compatibility.

Details

Input data must be provided in two text files, one with the data (with extension `dat`) and another with the station coordinates (with extension `est`). Both have as base name, 'VRB_YEAR-YEAR', composed by the short name of the climatological variable, and the initial and final years of the data, as set in the first three parameters of the call, `varcli`, `anyi` and `anyf`.

Data are stored in a free blank separated format (any number of data items per line is allowed), in chronological order, station by station (all data from station 1 go first, then all data from station 2, and so on). As dates are not stored in this file, all data must be present in the file, using a code for any missing data in the records (NA by default, but any other code can be used, provided that they are specified in the parameter `na.strings`).

The stations file, with extension `est`, is also a blank separated text file where each line identifies a single station, with structure 'X Y Z CODE NAME'. Coordinates X and Y are expected in geographical degrees (longitude and latitude, in this order and in decimal form). Otherwise they will be assumed to be in km, or in m if the mean of either X and Y is greater than 10000; elevation Z must be supplied in m; and the identification CODE and the full NAME of the station must be quoted if they contains blanks). Fully reliable series may be marked by putting an asterisk (*) at the beginning of their CODE to skip their outlier and break-point analysis. This is not recommended with observed series, but can be useful when using reanalysis series as references in data sparse regions.

This function will stop with an error condition if any time step becomes void of data in all stations at the same time. One or more series with data in the void time steps must be added to successfully run `homogen` again. If no other series are available in the area, reanalysis series of the closer grid-points can be used, adding their coordinates to the `*.est` file and prepending an asterisk (*) to the codes assigned to the series as mentioned above.

`dz.max` (and `dz.min` if different from `dz.max`) can be a vector of two values, one for suspect data and the other for probable errors. Only the latter will be deleted, but all will be listed in the `*_out.csv` output file. By default, the more extreme 0.01% in each tail of the distribution will be considered errors, and values exceeding 0.1% will be suspect data. Inspection of the anomalies

histogram near the end of the PDF output file will help in tuning these parameters by setting number of standard deviations to be used as rejection thresholds.

`inhht` has a default value of 25, which is a suitable conservative value for monthly values of temperature, but not so much for precipitation or for daily series. Therefore it is advisable to adjust it empirically with the help of the histograms available by the end of the graphic output. Anyway, inhomogeneities in daily or subdaily series should be detected on their monthly aggregates, which can be easily obtained by means of the function `dd2m`. Two values can be given to this parameter (one for each of the two detection stages), as in e.g. `inhht=c(30, 25)`. When only one value is provided, it will be used for both stages. If any or both values are zeros, the corresponding homogenization stage will be skipped.

The default value `wz=0.001` gives to the vertical coordinate (in m) the same weight as the horizontal coordinates (internally managed in km). Other values can be set to overweight elevation differences (`wz>0.001`) or to calculate only horizontal distances (`wz=0`).

`vmin` and `vmax` are unset by default, but if the variable is found to have a skewed probability distribution with a minimum value of zero, `vmin` will be set to zero. The same will happen if the user sets `std=2`.

`sufbrk` is only relevant when `metad=TRUE`. Its default value 'm' is meant to read the file of break-points detected at the monthly scale, but if the data were originally monthly, `sufbrk=''` should be set.

`tinc`, unset by default, can be defined for subdaily data, as in e.g.: `tinc='3 hours'`, especially if first and/or last years are incomplete. Units can be 'hours', 'mins' or 'secs'.

The default `cex=1.1` increase by a 10% the size of labels in the graphic output. Note that if station names are long, they will not fit in titles when increasing this parameter too much.

The graphic output file (in PDF format) begins with a first quality control of the series, providing box-plots for every series showing (1) the range of their values, (2) their increments between successive terms and (3) the length of segments with constant data. Too big outliers are deleted at this stage because they would compromise the quality of the homogenization and missing data filling. During the rest of the process outlier detection and deletion is based on spatial differences between neighboring normalized data. (Deleted data which were not errors but due to local phenomena can be restored to the homogenized series with the help of the `datrestore` function.)

The following pages offer: (a) a summary of the data availability and frequency distribution; (b) a correlogram of the first differences of the series, (c) a dendrogram based on these correlations and a map with the station locations (marked with numbers if less than 100, and with symbols otherwise); (d) graphics of normalized spatial anomalies showing the detected breaks, the minimum distance to a reference data and the number of references used; (e) a histogram of maximum `inhht` values found in overlapping window analysis; (d) and (e) are repeated for the analysis on the whole series; (f) histograms of number of splits per station and per year; (g) graphics of final anomalies of the series; (h) graphics of the reconstructed series and applied corrections; (i) a histogram of the normalized anomalies of all data (useful to set rejection thresholds for the outliers); (j) final histograms of `inhht` values; and (k) a plot of quality/singularity of the stations (a bad score may be due to a bad quality of the series, but also to a singular siting with a peculiar micro-climate).

Note that every time that a significant shift in the mean of the series is detected, it will be split into two (potentially) homogeneous sub-periods, and hence the final number of homogenized series will be increased, as complete homogeneous series will be reconstructed from all of them. When several homogeneous series have been yielded for the same location, the user can choose to use that reconstructed from the last sub-period (the usual behavior of other homogenization packages),

which is perfect for climate monitoring of newly incoming data. However, statistics derived from all of them can be useful for climatic mapping, when no a priori knowledge can indicate which of the sub-periods will be more representative at the spatial scale of the map).

The processing time can range from seconds (a few monthly series) to many hours (hundreds of daily series). If you must process a huge amount of data, you should consider splitting your study region into smaller areas to be homogenized independently.

Value

This function does not return any value, its results being saved to files with the same base name as the input files, and extensions:

- ***.txt**: A text file that logs all the processing output,
- ***_out.csv**: List of corrected outliers,
- ***_brk.csv**: List of corrected breaks,
- ***.pdf**: PDF file with a collection of diagnostic graphics,
- ***.rda**: Homogenization results in R binary format, used by the `dahstat` and other post-processing functions, but can be loaded by the user for further data manipulation with the function `load`. This file contains the following objects:
 - dat** matrix of the original series,
 - dah** matrix of the homogenized series,
 - nd** number of data (time steps) in every series,
 - ndec** number of decimals in the data,
 - uni** data units,
 - est.c** data frame with columns:
 - X** longitude,
 - Y** latitude,
 - Z** elevation,
 - Code** station code,
 - Name** station name,
 - pod** percentage of original data,
 - snht** (or `cuct` when `test='cuct'`): Remaining inhomogeneity test values in the homogenized series. Can be greater than the set `inh` threshold because of a lower number of reference stations,
 - rmse** estimated root mean squared errors of the homogenized series
 - ct** Cluster Analysis series groups,
 - nei** number of input series,
 - ne** number of series after the homogenization,
 - nm** number of "months" (data items) in a year (0=daily data),
 - std** type of normalization applied to the data,
 - x** vector of the time dimension,
 - ini** initial date of the period under study.

References

Guijarro JA (2014): Quality Control and Homogenization of Climatological Series. In Eslamian S (Ed.), Handbook of Engineering Hydrology, Vol. 1: Fundamentals and Applications. Francis and Taylor, CRC Group, USA, ISBN 9781466552357, 636 pp.

Azarin-Molina C, Guijarro JA, McVicar TR, Trewin BC, Frost AJ, Chen D (2019): An approach to homogenize daily peak wind gusts: An application to the Australian series. Int. J. Climatol., 39:2260-2277. doi: 10.1002/joc.5949

Dumitrescu A, Cheval S, Guijarro JA (2019): Homogenization of a combined hourly air temperature dataset over Romania. Int. J. Climatol., 40:2599-2608, DOI: 10.1002/joc.6353

Visit <<https://climatol.eu/>> for updates of code and documentation (user's guide, links to videos, etc).

See Also

[dahstat](#), [dahgrid](#), [outrename](#), [datrestore](#), [dd2m](#)

Examples

```
## Set a temporal working directory and write input files:
wd <- tempdir()
wd0 <- setwd(wd)
data(climatol_data)
write.table(Temp.est, 'Temp_1961-2005.est', row.names=FALSE, col.names=FALSE)
write(Temp.dat, 'Temp_1961-2005.dat')

## Now run the example:
homogen('Temp', 1961, 2005)

## Return to user's working directory:
setwd(wd0)

## Input and output files can be found in directory:
print(wd)
```

IDFcurves

Obtain Intensity-Duration-Frequency curves

Description

Intensity-Duration-Frequency curves are obtained from a sub-hourly time series of precipitation by adjusting Generalized Extreme Value distribution to annual maxima of different time intervals.

Usage

```
IDFcurves(prdat, stname, clmn=1:2, tz='utc', na.code=NA,
prunits='mm', mindpy=0.8, gumbel=TRUE, timeaggr=c(10,20,30,60,120,180,360,720),
retper=c(5,10,20,30,50,75,100),...)
```

Arguments

<code>prdat</code>	Data frame with Time (as POSIXct) and sub-hourly precipitation data.
<code>stname</code>	Station name.
<code>clmn</code>	Columns where Time and precipitation data are located in <code>prdat</code> .
<code>tz</code>	Time zone ['utc' by default].
<code>na.code</code>	Numeric missing data code.
<code>prunits</code>	Precipitation units [mm].
<code>mindpy</code>	Minimum available data proportion to process data in any year.
<code>gumbel</code>	Adjust a Gumbel distribution? [TRUE].
<code>timeaggr</code>	Time intervals (in minutes) on which to aggregate precipitation.
<code>retper</code>	Return periods (in years) for extreme precipitation estimation.
<code>...</code>	Additional graphic parameters.

Details

The precipitation time series must be provided as a data frame with POSIXct times in the first column and precipitation in the second. However, these data can be in other columns of a wider data frame if the columns containing these variables are defined in the parameter `clmn`.

When setting `gumbel=FALSE` a Generalized Extreme Value distribution will be adjusted instead of the particular case of a Gumbel distribution.

Value

A table of maximum precipitation accumulations (totals, not mm/h as in the graphic) is returned invisibly.

Examples

```
## Not run:
data(climatol_data)
tab <- IDFcurves(prec10min,'My airport',cex.axis=1.2,cex.lab=1.2) #IDF plot

## See the maximum precipitation accumulations in the different time intervals:
tab

## End(Not run)
```

meteogram

*Daily meteogram of eight meteorological variables***Description**

This function plots a meteogram from hourly or sub-hourly data of eight meteorological variables available in a data frame spanning one day.

Usage

```
meteogram(df, code='', name='', cols=1:9, tz='utc', hlab='Hours',
datefm='%Y-%m-%d', vlab=c('Wind direction (deg)', 'Wind speed (m/s)', NA, NA,
'Temperature (C)', 'Rel. humidity (%)', 'Precip. (mm)', 'Pressure (hPa)'),
vcol=c(hsv(.1, 1, .9), hsv(.1, 1, .9), 2, 2, 2, hsv(.4, 1, .7), 4, 'brown'),
llim=c(0, 0, NA, NA, 0, 0, 0, NA), ulim=c(360, 20, NA, NA, 20, 100, 4, NA))
```

Arguments

df	Data frame with (around) one day of data.
code	Code of the station.
name	Name of the station.
cols	Column order of the expected variables (see details).
tz	Time zone of the supplied time vector ('utc' by default).
hlab	Label for hours ('Hours' by default).
datefm	Date format for the title of the meteogram (the default is '%Y-%m-%d', the ISO 8601 date format).
vlab	Variable labels.
vcol	Colors for every variable.
llim	Lower graphic limits (if fixed).
ulim	Upper graphic limits (if fixed).

Details

This function expects a data frame containing observation time and eight meteorological variables in this column order:

1. Time of the observation (as POSIXct)
2. 10 minutes average wind direction in degrees
3. 10 minutes average wind speed in m/s
4. 3 sec. maximum gust direction in degrees
5. 3 sec. maximum gust speed in m/s
6. Air temperature in degrees Celsius

7. Relative humidity in %
8. Precipitation in mm
9. Barometric pressure in hPa

However, if the data frame has these variables in a different order, it can be specified with the parameter `cols`.

See [strftime](#) for ways to specify date formats.

See Also

[strftime](#)

Examples

```
data(climatol_data)
meteogram(AWS_1day, 'S123', 'My airport')
```

MHisopleths

Isopleths on a months-hours diagram

Description

This function takes hourly or subhourly data (spanning at least one year) and plots isopleths of the chosen variable in a colored two-dimensional (months, hours) diagram.

Usage

```
MHisopleths(dat, vrb, fun='mean', xlab='Months', ylab='Hours', cex=1.2,
  col4RP=c('cyan','yellow','red'), title='')
```

Arguments

<code>dat</code>	dataframe containing the data in columns with date/time of class POSIX in the first column.
<code>vrb</code>	name of the column containing the chosen data.
<code>fun</code>	function to aggregate subhourly data into hourly.
<code>xlab, ylab</code>	labels for the X and Y axis.
<code>cex</code>	character expansion parameter for the size of labels.
<code>col4RP</code>	vector of colors for the <code>colorRampPalette</code> function.
<code>title</code>	main title.

Details

The user can choose any column of data present in `dat`. (Depending on the variable the default colors may not be the most appropriate.)

Examples

```
data(climatol_data)
MHisopleths(AWS_1year,'Temp',title='Mean temperature (C) -- My airport, 2002')
MHisopleths(AWS_1year,'WSpd',title='Wind speed (m/s) -- My airport, 2002')
```

outrename	<i>Rename homogen's output files</i>
-----------	--------------------------------------

Description

This function inserts a suffix to the output file names of homogen, to prevent them from being overwritten by any further run.

Usage

```
outrename(varcli, anyi, anyf, suffix, restore=FALSE)
```

Arguments

varcli	Short name of the studied climatic variable, as in the data file name.
anyi	Initial year of the study period.
anyf	Final year of the study period.
suffix	Suffix to be inserted (or removed) in the output file names.
restore	Set this parameter to TRUE to remove the suffix previously inserted by this function. (FALSE by default).

Details

The suffix is appended to the varcli after a hyphen. The purpose of this function is to allow a new application of homogen to the same data with different parameters without overwriting the previous results.

See Also

[homogen](#)

Examples

```
## Set a temporal working directory, write input files and homogenize them:
wd <- tempdir()
wd0 <- setwd(wd)
data(climatol_data)
write.table(Temp.est,'Temp_1961-2005.est',row.names=FALSE,col.names=FALSE)
write(Temp.dat,'Temp_1961-2005.dat',ncolumns=12)
datsubset('Temp',1961,2005,1991) #subset data to shorten example run time
homogen('Temp',1991,2005) #obtain homogenization output files
```

```
## Now run the example:
outrename('Temp',1991,2005,'bak') #rename them to avoid being overwritten

## Return to user's working directory:
setwd(wd0)

## Input and output files can be found in directory:
print(wd)
```

QCthresholds

Obtain monthly thresholds for Quality Control alerts

Description

This function calculate monthly quantiles of daily or subdaily series that can be used as thresholds for Quality Control alerts.

Usage

```
QCthresholds(dat, ndec=1, probs=c(0.,.001,.01,.99,.999,1.), minval=NA,
maxval=NA, homog=TRUE, verb=TRUE)
```

Arguments

dat	Either the name of a *.rda file of climatol homogenization results or a data.frame of daily (or subdaily) data in columns, dates or date/times (of class Date or POSIXct) in the first column and station codes in the header
ndec	number of decimals of output values [1] (defaults shown between brackets)
probs	probabilities of the quantiles to be computed [0., .001, .01, .99, .999, 1.]
minval	minimum value to compute runs of constant values [NA].
maxval	maximum value to compute runs of constant values [NA].
homog	use homogenized data if a *.rda file is used as input [TRUE].
verb	list all calculated values? [TRUE].

Details

minval and maxval allow to exclude frequent values that would result in the report of long runs of identical data. Examples: set minval=0.1 in daily precipitation to avoid long runs of zeros or set maxval=97 in relative humidity to avoid long runs of near saturation values in episodes of persistent fog.

Calculated thresholds are shown in the text output and are also saved in a binary R file named QCthresholds.Rdat, which contains the matrices thr1, thr2 and thr3. Load this file and write the thresholds in the required format for importation into a Climate Data Management System.

See Also

[homogen](#)

Examples

```
## Set a temporal working directory and write input files:
wd <- tempdir()
wd0 <- setwd(wd)
data(climatol_data)

## Now run the examples:
QCthresholds(RR3st,minval=0.1) #daily precipitation of three stations
QCthresholds(TX3st) #daily maximum temperatures of three stations
load('QCthresholds.Rdat') #load last calculated thresholds
thr1[,1,] #thresholds with 0% probability to find lower values
thr1[,3,] #monthly thresholds of the third station
thr2 #thresholds of absolute increments between consecutive data
thr3 #thresholds for equal data run lengths

## Return to user's working directory:
setwd(wd0)

## Input and output files can be found in directory:
print(wd)
```

rclimdex2climatol *Convert RCLimDex daily data files to climatol input format*

Description

This function can be useful to prepare the climatol input files when the user have their daily data in RCLimDex format.

Usage

```
rclimdex2climatol(stfile, stcol=1:5, kvar, chrcod=c(6,10), sep=' ', anyi=NA,
anyf=NA, mis=-99.9, mindat=365, header=TRUE)
```

Arguments

stfile	Name of the file with the list of data file names and station coordinates, codes and names.
stcol	Columns in stfile holding data file names, longitudes, latitudes, elevations and station codes and names. (Defaults to 1:5.)
kvar	RCLimDex variable to extract: 1 (RR), 2 (TX), 3 (TN).
chrcod	Initial and final characters of data file names to be used as station codes. c(6,10) by default.
sep	Field separator in stfile and data files (space or tab by default).
anyi	First year to study. (Defaults to the first year of available data.)
anyf	Last year to study. (Defaults to the last year of available data.)

mis	Missing data code. (Defaults to -99.9.)
mindat	Minimum required number of data per station. (Defaults to 365 daily data.)
header	Do files have a header line? TRUE by default).

Details

Users of the RClimDex program can convert their daily data files to the climatol format. All files listed in `stfile` will be read, and the selected variable (precipitation, maximum or minimum temperature) will be stored in a unique `*.dat` file, with its companion `*.est` station file. Therefore, if you want to convert all three variables, you must run this function three times.

Coordinates must be given in degrees with decimals, using the minus sign for southern latitudes and western longitudes.

See Also

[homogen](#), [climatol2rclimdex](#)

Examples

```
## Set a temporal working directory:
wd <- tempdir()
wd0 <- setwd(wd)

## Prepare a few files in RClimDex format:
data(climatol_data)
gY=c(46,46,46); mY=c(06,15,14); sY=c(42,25,53)
gX=c(14,15,14); mX=c(03,09,50); sX=c(05,06,05)
df=data.frame(File=c('p064.txt', 'p084.txt', 'p082.txt'),
  LatDeg=gY, LatMin=mY, LatSec=sY, LonDeg=gX, LonMin=mX, LonSec=sX,
  elev=SIstations[,3], name=SIstations[,5])
write.table(df, 'stations.txt', sep='\t', row.names=FALSE)
write.table(p064.df, 'p064.txt', sep='\t', row.names=FALSE, quote=FALSE)
write.table(p084.df, 'p084.txt', sep='\t', row.names=FALSE, quote=FALSE)
write.table(p082.df, 'p082.txt', sep='\t', row.names=FALSE, quote=FALSE)

## Now run the example:
rclimdex2climatol('stations.txt', 3, chrcod=c(1,4))

## Return to user's working directory:
setwd(wd0)

## Input and output files can be found in directory:
print(wd)
```

runtnd	<i>Running trends on time windows of different lengths</i>
--------	------------------------------------------------------------

Description

This function plots running trends on time windows of different lengths in a colored grid with axis 'Last year' and 'Window length'.

Usage

```
runtnd(d, anyi, minyr=10, units='Units', pernyr=10, stname=NA, k=NULL,
       palneg=c('blue', 'white'), palpos=c('white', 'red'), ...)
```

Arguments

d	Series of annual values (without missing data).
anyi	Initial year of the series.
units	Units label for the legend.
minyr	Minimum no. of years to compute trends (10 by default).
pernyr	Factor for trend units (per 10 years by default).
stname	Station name (for the title).
k	Vector of breaks for the trend scale colors (automatically set by default).
palneg	Color gradation for negative trends [c('blue', 'white')].
palpos	Color gradation for positive trends [c('white', 'red')].
...	Additional graphic parameters.

Details

The input must be a complete (no missing data) series of annual values.

If `minyr` is negative, running trends calculated on `-minyr` years will be plotted, with increasing line widths when significance reaches 0.10 and 0.05 levels. Otherwise, a colored graphic of running trends calculated on different window widths will be displayed, masking low significance values with white dots.

Value

A data frame or a list with `tnd` (trends) and `pvl` (p-values) is returned invisibly when `minyr` is negative or positive, respectively.

Examples

```
data(climatol_data)
runtnd(Tav, 1901, -30, units='C', stname='Oslo', cex.axis=1.2, cex.lab=1.2)
runtnd(Tav[31:120], 1931, 30, units='C', stname='Oslo')
```

sef2climatol *Convert SEF data files to climatol input files.*

Description

This function reads all SEF files contained in a directory and writes their data in *.dat and *.est climatol input files.

Usage

```
sef2climatol(dr, Vbl, varcli=Vbl, ndec=1, na.strings="NA", mindat=NA)
```

Arguments

dr	directory containing the SEF files
Vbl	name of the variable in the SEF files
varcli	name of the variable in the climatol destination files
ndec	number of decimals to save
na.strings	missing data codes (specified as quoted strings)
mindat	minimum required number of data per station

Details

SEF (Station Exchange Format) is the Copernicus Climate Change Service format for Data Rescue projects. Visit <https://datarescue.climate.copernicus.eu/node/80>

Missing elevations will be assigned the value 99

Some files may contain a single quotation mark in the metadata field, causing not reading the end of line until a pairing quoting is found in the following line, hence skipping half of the data. Parameter quote='\"' has been set in the reading command as a workaround.

All data are dumped into a temporary file named SEFdata.csv, which is used by the function csv2climatol to write the input files for climatol.

See Also

[csv2climatol](#), [homogen](#)

Examples

```
## Set a temporal working directory and write input files:
wd <- tempdir()
wd0 <- setwd(wd)

## Create a directory and copy all SEF files to be processed:
dir.create('dir1')
file.copy(exampleFiles('GHCN_v4_Bhamo.tsv'), 'dir1')
file.copy(exampleFiles('GHCN_v4_Diamond_Island.tsv'), 'dir1')
```

```
## Now run the function:
sef2climatol('dir1','ta')

## Return to user's working directory:
setwd(wd0)

## Input and output files can be found in directory:
print(wd)
```

windrose

Wind-rose plot

Description

This function plots a wind-rose from a data frame with columns DateTime, Wind direction and Wind speed.

Usage

```
windrose(dat, cols=1:3, code='', name='', uni='m/s', ndir=16, spdcut=NULL,
maxnsc=8, fnum=4, fint=5, flab=2, ang=-3*pi/16, margin=c(0,0,4,0),
pal=c('cyan','yellow','orange','red','brown'), ...)
```

Arguments

dat	Data frame with columns DateTime, Wind direction and Wind speed.
cols	Columns containing DateTime, Wind direction and Wind speed [1:3].
code	Station code.
name	Station name.
uni	Speed units for the legend header ['m/s'].
ndir	Number of classes of wind direction [16].
spdcut	Speed values to set the wind speed classes. If not provided, classes will be automatically calculated.
maxnsc	Maximum number of wind speed classes [8].
fnum	Number of reference circles to plot [4].
fint	Frequency interval (in %) between reference circles [5].
flab	Parameter indicating which circles must be labelled: 1: Label outer circle only, 2: Label all circles (the default), Other value: Do not label any circle.
ang	Angle along which circles will be labeled, in radians [-3*pi/16].
margin	Margins vector for the plot (to be passed to par) [c(0,0,4,0)].
pal	Color gradation to fill the frequency polygons.
...	Other graphic parameters.

Details

After reading the data, a frequency table is calculated in 16 wind directions and a variable number of wind speed classes, which can be set by the user. Calm observations (wind speed equal to zero) are distributed proportionally into the first wind speed class. The wind direction data must be provided in degrees.

This table, which covers all available pairs of wind direction and speed present in the data frame, is the basis of the wind-rose plot.

Value

The table of wind frequencies by direction and speed classes is returned invisibly.

Examples

```
data(climatol_data) #load example data
windtable <- windrose(AWS_1year, 1:3, 'st123', 'My airport') #plot windrose
print(windtable) #display the table of calculated wind frequencies
```

 xls2csv

*Join all data in *.xls or *.xlsx files into a single CSV file*

Description

This function reads all *.xls or *.xlsx files contained in a directory and dumps their data into a single CSV file.

Usage

```
xls2csv(tmpdir, archdir, var, datcols=1:4, codesep='-', dec='.', sep=',')
```

Arguments

tmpdir	temporal directory containing the files to read.
archdir	directory where to archive files after processing.
var	destination name of the variable.
datcols	data columns to be written to the output file.
codesep	character string separating the code from the rest of the file name ('-' by default).
dec	character to use as decimal point in the output file ('.' by default).
sep	character separating data in the output file (',' by default).

Details

File names must begin with their station code, which may optionally be followed by a hyphen ('-') or other code separator character (specified with the parameter `codesep`) and the name of the station or other characters.

File contents must have one header line at the top. If they contain more, supplementary header lines should have at least one empty cell in the columns of date and data to be read.

After their data have been dumped into the output `xls_*_data.csv` file, original files are moved to the `archdir` directory.

Note that data are appended to the output CSV files every time you run this function putting new files in the `tmpdir` directory.

Code and station names (if included in the file names) are appended to `xls_*_stations.csv`.

`climatol` input files can then be obtained from both output `xls_*.csv` files with the `csv2climatol` function.

See Also

[csv2climatol](#), [homogen](#)

Examples

```
## Set a temporal working directory:
wd <- tempdir()
wd0 <- setwd(wd)

## Create origin and destination directories and copy example input files:
dir.create('dir1'); dir.create('dir2')
file.copy(exampleFiles('p064.xlsx'), 'dir1')
file.copy(exampleFiles('p082.xlsx'), 'dir1')
file.copy(exampleFiles('p084.xlsx'), 'dir1')

## Now run the example:
xls2csv('dir1', 'dir2', 'TN', datcols=c(1:3,6))

## Return to user's working directory:
setwd(wd0)

## Input and output files can be found in the directory:
print(wd)
```

Index

- * **cluster**
 - homogen, [22](#)
 - * **datagen**
 - dahgrid, [5](#)
 - dahstat, [7](#)
 - dd2m, [16](#)
 - homogen, [22](#)
 - IDFcurves, [27](#)
 - QCthresholds, [32](#)
 - windrose, [37](#)
 - * **datasets**
 - Datasets, [11](#)
 - * **graphs**
 - homogen, [22](#)
 - * **hplot**
 - dens2Dplot, [18](#)
 - diagwl, [19](#)
 - homogen, [22](#)
 - IDFcurves, [27](#)
 - meteogram, [29](#)
 - MHisopleths, [30](#)
 - runtn, [35](#)
 - windrose, [37](#)
 - * **manip**
 - climatol2rclimindex, [2](#)
 - csv2climatol, [4](#)
 - daily2climatol, [9](#)
 - datsubset, [13](#)
 - db2dat, [14](#)
 - dd2m, [16](#)
 - fix.sunshine, [21](#)
 - rclimindex2climatol, [33](#)
 - sef2climatol, [36](#)
 - xls2csv, [38](#)
 - * **misc**
 - climatol-internal, [2](#)
 - exampleFiles, [20](#)
 - * **ts**
 - homogen, [22](#)
 - * **utilities**
 - dahstat, [7](#)
 - datrestore, [12](#)
 - fix.sunshine, [21](#)
 - outrename, [31](#)
- AWS_1day (Datasets), [11](#)
AWS_1year (Datasets), [11](#)
- cerrar (climatol-internal), [2](#)
climatol-internal, [2](#)
climatol.version (climatol-internal), [2](#)
climatol2rclimindex, [2](#), [34](#)
climatol_data (Datasets), [11](#)
csv2climatol, [4](#), [16](#), [36](#), [39](#)
cuct (climatol-internal), [2](#)
- dahgrid, [5](#), [9](#), [17](#), [27](#)
dahstat, [7](#), [17](#), [27](#)
daily2climatol, [9](#)
Datasets, [11](#)
datcli (Datasets), [11](#)
datrestore, [12](#), [27](#)
datsubset, [13](#)
db2dat, [14](#)
dd2m, [16](#), [27](#)
dens2Dplot, [18](#)
diagwl, [19](#)
- exampleFiles, [11](#), [12](#), [20](#)
- fix.sunshine, [21](#)
- homogen, [3](#), [5–7](#), [9](#), [10](#), [13](#), [16](#), [17](#), [21](#), [22](#), [31](#),
[32](#), [34](#), [36](#), [39](#)
- IDFcurves, [27](#)
- meteogram, [29](#)
MHisopleths, [30](#)
- outrename, [27](#), [31](#)

p064.df (Datasets), 11
p082.df (Datasets), 11
p084.df (Datasets), 11
prec10min (Datasets), 11

QCthresholds, 32

rclimindex2climatol, 10, 33
read.dat (climatol-internal), 2
RR3st (Datasets), 11
runtn, 35

sef2climatol, 36
SIstations (Datasets), 11
snht (climatol-internal), 2
strftime, 30

Tav (Datasets), 11
Temp.dat (Datasets), 11
Temp.est (Datasets), 11
TN3st (Datasets), 11
TX3st (Datasets), 11

unsuffix (climatol-internal), 2

windrose, 37
wtest (climatol-internal), 2

xls2csv, 5, 38