

Package ‘clim.pact’

January 2, 2012

Version 2.3-10

Date November 22 2011

Title Climate analysis and empirical-statistical downscaling (ESD)
package for monthly and daily data.

Author Rasmus E. Benestad <rasmus.benestad@physics.org>

Maintainer Rasmus E. Benestad <rasmus.benestad@physics.org>

Depends ncdf, akima, R (>= 2.10.0)

Description The package contains R functions for retrieving data, making climate analysis and downscaling of monthly mean and daily mean global climate scenarios. (Windows-users may need to obtain the ‘ncdf’ package from URL ‘<http://www.stats.ox.ac.uk/pub/RWin/>’ to get clim.pact to work). The package is also described in the book ‘Empirical-Statistical Downscaling’ (<http://www.worldscibooks.com/environsci/6908.html>)

License GPL (>= 2)

URL <http://rcg.gvc.gu.se/edu/esd.pdf>

ZipData no

Repository CRAN

Date/Publication 2011-11-22 08:52:54

R topics documented:

ABC4ESD	3
addClim	4
addland	5
addland1	6
adjustEOF	6
anomaly.field	7
anomaly.station	8

avail.elem	9
bergen.dm	10
bergen.t2m	11
caldat	12
catFields	13
CCA	15
cdfcont	18
cdfextract	19
coherence	20
composite.field	21
COn0E65N	22
corEOF	23
corField	24
datestr2num	25
delta	26
dist2norm	26
distAB	27
DNMI.t2m	28
DS	29
ds2station	34
DSpdf.exp	35
EOF	36
eof.c	39
eof.dc	40
eof.dmc	42
eof.mc	44
eof.slp	45
getClimateExplorer	46
getdnmi	47
getecsn	48
getgiss	49
getnacd	51
getnordklim	52
grd.box.ts	53
helsinki.t2m	55
instring	56
julday	57
km2lat	58
km2lon	59
koebenhavn.t2m	60
lagStation	61
lower.case	61
map	62
mapEOF	63
mapField	64
meanField	65
mergeEOF	66
mergeStation	67

mixFields	68
mod	70
MVR	71
narp	73
newFig	74
num2str	74
objDS	75
oslo.dm	77
oslo.t2m	79
patternIndex	80
plotDS	81
plotEOF	82
plotField	83
plotStation	84
plumePlot	86
POP	87
r2cdf	88
retrieve.nc	90
reverse	92
rotate	93
satellite	94
SSA	95
station.obj	96
station.obj.dm	98
stationmap	100
stockholm.t2m	101
strip	102
tromsoe.t2m	103
upper.case	104
what.data	104

Index**105**

ABC4ESD

*ABC for Empirical-Statistical Downscaling.***Description**

Downloads the compendium 'Empirical-Statistical Downscaling' from <http://home.broadpark.no/~rbene>. Other literature on matters relating to clim.pact can be found on <http://home.broadpark.no/~rbene/ras-publ.html>

Usage

```
ABC4ESD(browser="firefox",url="http://rcg.gvc.gu.se/edu/esd.pdf")
```

Arguments

browser	Browser to use
url	URL for the ESD-compendium.

Value

a PDF-document.

Author(s)

R.E. Benestad

References

Benestad, R.E., Hanssen-Bauer, I. And Chen, D.(2008) Empirical-Statistical Downscaling, World Scientific Publishers, ISBN 978-981-281-912-3 <http://www.worldscibooks.com/environsci/6908.html>

Examples

```
## Not run: ABC4ESD()
```

addClim	<i>Add climatology</i>
---------	------------------------

Description

Adds the climatological values to anomalies. The inverse of [anomaly.station](#).

Usage

```
addClim.station(x,param=c("t2m","precip"))
preClim.station(x,dd=NULL,mm=NULL,yy=NULL,param=1)
```

Arguments

x	A station object or a vector.
mm	month
dd	day.
yy	year.
param	For daily objects: identifies the parameters (addClim.station) or determines which model to use (preClim.station).

Value

A station object (addClim.station) or a vector (preClim.station).

Author(s)

R.E. Benestad

Examples

```
data(oslo.dm, envir=environment())
oslo.dma <- anomaly.station(oslo.dm)
oslo.dmac <- addClim.station(oslo.dma)
plot(oslo.dm$yy+(oslo.dm$mm-1)/12+oslo.dm$dd/365.25, oslo.dm$t2m,
      type="l", lwd=3, col="grey", xlim=c(1980, 1985))
lines(oslo.dm$yy+(oslo.dm$mm-1)/12+oslo.dm$dd/365.25,
      oslo.dmac$t2m, lty=2)
lines(oslo.dm$yy+(oslo.dm$mm-1)/12+oslo.dm$dd/365.25,
      preClim.station(oslo.dma, oslo.dm$dd, oslo.dm$mm,
      oslo.dm$yy), col="red", lwd=2)
```

`addland`*Add land contours to map.*

Description

The function superimposes land contours on a map.

Usage

```
addland(col="grey50", lwd=1)
```

Arguments

<code>col</code>	colour
<code>lwd</code>	line width

Author(s)

R.E. Benestad

Examples

```
plot(c(-90, 90), c(0, 80), type="n")
addland()
grid()
```

addland1	<i>Coordinates of coast line.</i>
----------	-----------------------------------

Description

Coordinates of coast line: `lat.cont` and `lon.cont` give the latitude and longitude coordinates of the coasts (continents).

Usage

```
data(addland1)
data(addland2)
```

Format

Two vectors, `lat.cont` and `lon.cont`, of length 124622.

Source

Ferret web site

References

URL <http://ferret.wrc.noaa.gov/Ferret/>

Examples

```
library(clim.pact)
data(addland1)
ls()           # [1] "lat.cont" "lon.cont"
```

adjustEOF	<i>Adjust common EOFs</i>
-----------	---------------------------

Description

Forces same std and mean in the subsegments of common PCs.

Usage

```
adjustEOF(x)
```

Arguments

`x` a 'eof' object (see [EOF](#)).

Value

an 'eof' object

Author(s)

R.E. Benestad

Examples

```
data(eof.c)
eof <- adjustEOF(eof.c)
plotEOF(eof)
```

anomaly.field *Anomalies of a field object.*

Description

Estimates the anomalies of a field object. Also see [anomaly.field](#).

Usage

```
anomaly.field(x,period=NULL)
daily2monthly.field(field,min.days.month=20,method="colMeans",na.rm=TRUE)
```

Arguments

x	A field object.
period	Period to use as climatology NULL -> the entire series as reference clim.
field	A field object.
min.days.month	Minimum days per month with valid data allowed for estimating the monthly mean
method	Method for estimating the monthly value: e.g. "colMeans" or "colsums"
na.rm	Remove invalid data.

Value

A field object.

Author(s)

R.E. Benestad

Examples

```
## Not run:
slp<-retrieve.nc("ncep_slp.nc")
slp.a<-anomaly.field(slp)

## End(Not run)
```

anomaly.station	<i>Anomaly.station</i>
-----------------	------------------------

Description

Computes anomalies of a station series by subtracting the climatology. The climatology is estimated either by taking the average of the respective months over a given reference period or a least squares fit to the 6 leading harmonics, depending on the appropriateness. Also see [anomaly.field](#).

Usage

```
anomaly.station(obs,period=c(1961,1990),param=c("t2m","precip"))
daily2monthly.station(obs,param="t2m",min.days.month=20,
                      method="mean",na.rm=TRUE)
```

Arguments

obs	Monthly station series
period	Period to use as climatology NULL -> the entire series as reference clim.
param	For daily objects: identifies the names of the parameters.
min.days.month	Mininum days per month for estimating monthly mean.
method	Meethod for treating the daily values.
na.rm	na.rm

Value

Station series object.

Author(s)

R.E. Benestad

Examples

```
library(clim.pact)
data(oslo.t2m)
oslo.t2ma<-anomaly.station(oslo.t2m)
```

avail.elem	<i>Available elements</i>
------------	---------------------------

Description

The function `avail.preds` searches for available predictors and returns a list of file names. `avail.elem` returns a list of available elements and `avail.locs` a list of available locations. These functions are support functions for `getnacd` and `getnordklim`.

Usage

```
avail.ds(direc="output")
avail.eofs(direc="data")
avail.preds(direc="data")
avail.elem()
avail.locs(ele)
```

Arguments

<code>direc</code>	String containing the data directory.
<code>ele</code>	Integer code for element in the Nordklim dataset:
101	mean T(2m)
111	mean maximum T(2m)
112	highest maximum T(2m) [Th]
113	day of Th date Thd
121	mean minimum T(2m)
122	lowest minimum T(2m) [Tl]
123	day of Tl date Tld
401	mean SLP
601	monthly accum. precip.
602	maximum precip.
701	Number of days with snow cover (> 50% covered) days dsc
801	Mean cloud cover % N
911	mean snow depth

Value

<code>avail.preds</code>	vector of characters
<code>avail.locs</code>	a list with name, lons, lats, country, ident
<code>avail.elem</code>	a list with data.set, ele, name

Author(s)

R.E. Benestad

Examples

```

library(clim.pact)
avail.elem()$name
# [1] "mean T(2m)"
# [2] "mean maximum T(2m)"
# [3] "highest maximum T(2m)"
# [4] "day of Th date Thd"
# [5] "mean minimum T(2m)"
# [6] "lowest minimum T(2m)"
# [7] "day of Tl date Tld"
# [8] "mean SLP"
# [9] "monthly accum. precip."
#[10] "maximum precip."
#[11] "Number of days with snow cover (> 50% covered) days dsc"
#[12] "Mean cloud cover % N"
#[13] "mean snow depth"

# The following assumes that the subdirectory 'data' exists
## Not run:
avail.locs()$name[avail.locs()$country=="FIN"]
# [1] "HELSINKI"      "TURKU"          "TAMPERE"        "LAPPEENRANTA"
# [5] "JYVASKYLA"     "KUOPIO"         "KAJAANI"        "OULU"
# [9] "KUUSAMO"       "SODANKYLA"     "Maarianhamina" "Helsinki"
#[13] "Turku"         "Huitinen"      "Tampere"        "Hattula"
#[17] "Heinola"       "Virolahti"     "Lappeenranta"  "Lavia"
#[21] "Virrat"        "Orivesi"       "Jyvaeskylae"   "Vaasa"
#[25] "Ylistaro"     "Aehtaeri"      "Kuopio"         "Maaninka"
#[29] "Joensuu"      "Kestilae"      "Kajaani"        "Oulu"
#[33] "Yli-Ii"       "Pudasjaervi"  "Kuusamo"        "Sodankylae"

avail.preds()
# [1] "eof.dc.Rdata"
# [2] "eof.dmc.Rdata"
# [3] "eof.mc2.Rdata"
# [4] "eof.nn.dc.Rdata"
# [5] "eof.nn.dmc.Rdata"
# ...

## End(Not run)

```

bergen.dm

*Daily Bergen record.***Description**

A station record of daily mean temperature and daily precipitation from Bergen-Florida.

Usage

```
data(bergen.dm)
```

Format

a "daily.station.record"-class object.

t2m	a vector holding daily mean temperature.
precip	a vector holding daily precipitation.
dd	a vector holding day of month.
mm	a vector holding the month.
yy	a vector holding the year.
obs.name	the name of observation: eg c("Daily mean temperature", "Daily precipitation").
unit	the unite of observation: eg c("deg C", "mm/day").
ele	element code: eg c("tam", "rr").
station	local (national) station number.
lat	latitude.
lon	longitude.
alt	altitude.
location	name of location.
wmo.no	WMO number of station.
start	start of measurements.
yy0	first year of record.
country	name of country.
ref	reference to the data.

Source

The Norwegian Meteorological Institute, Climatology deivision.

References

The Norwegian Meteorological Institute, P.O. Box 43, 0313 Oslo, Norway (<http://www.met.no>).

Examples

```
data(bergen.dm)
```

```
bergen.t2m
```

Monthly mean temperature in Bergen.

Description

A station record of monthly mean temperature from Bergen-Florida.

Usage

```
data(bergen.t2m)
```

Format

a <- list of "monthly.station.record" class:

val	The monthly values (a 12-column matrix with one column for each year).
station	station number.
yy	The years of observation (vector).
lat,lon	Latitude and longitude of the location.
x.0E65N,y.0E65N	Distance in km from 0E, 65N.
location	Name of location .
wmo.no	WMO number.
start	Start of observatins from this location.
yy0	First year in current record.
ele	Code of theelement.
obs.name	Name of the element.
unit	Unit of the element.
country	The country in which the location is located.
quality	Code/description for data quality.
found	Flag: T - the data requested was found.
ref	Reference for the data set.

Source

The Nordklim data set: http://www.smhi.se/hfa_coord/nordklim/

References

Tuomenvirta et al. (2001), "Nordklim data set 1.0", DNMI KLIMA 08/01, pp. 26;
The Norwegian Meteorological Institute, P.O. Box 43, 0313 Oslo, Norway (<http://www.met.no>).

Examples

```
data(bergen.t2m)
```

caldat

Converts Julian days to month, day, and year

Description

The function computes month, day, and year from Julian days. The code is based on the algorithm from Press et al. (1989), "Numerical Recipes in Pascal", Cambridge, p. 13. See also `chron` and `date` for similar functions. This function was included to avoid the dependency to the `chron` and `date` packages. See also [julday](#).

Usage

```
caldat(julian)
```

Arguments

julian The Julian day from 1-1-1.

Value

a list with: month, day, and year

Author(s)

R.E. Benestad

Examples

```
caldat(1)            # month=1, day=2, year=-4713
caldat(1721424)    #     1,     1,     1
caldat(2440588)    #     1,     1,     1970
caldat(2452887)    #     9,     4,     2003
```

catFields

catFields

Description

Concatinates fields two different gridded sets of observation. The two fields must be stored on the same spatial grid, and the routine performs a bilinear spatial interpolation to place the data on the same grid. Observations/data for representing values at n different locations at a given time (t) can be described in terms of a vector

$$\vec{x}(t) = [x_1, x_2, \dots, x_n].$$

The data set consists of a time series of vectors which can be represented by the means of matrices

$$X = [\vec{x}(t_1), \vec{x}(t_2), \dots, \vec{x}(t_n)].$$

Two different sets of observations can be represented by two matrices Y and Z with dimensions k x n and k x m respectively (k is the number of spatial points, whereas n and m indicate the number of observations in time). The information in these two data sets are combined combining the two matrices using rbind. The major difference between this routine and rbind is that this routine takes care of all the 'house keeping' in terms of grid, time and variable information.

cat.field can be used to process single fields by setting 'field.2=NULL'. This option allows for interpolation and extraction of sub-regions or sub-intervals, removing the mean values, and selecting a particular month or season.

The output from cat.fields can be further analysed in EOF. By using a concatenation of two fields of similar data, eg observed and simulated sea level pressure (SLP), it is possible to carry out a common EOF analysis. The application of DS to the EOFs of concatenated fields provides an analysis similar

to the common EOH method described in Benestad (2001), "A comparison between two empirical downscaling strategies", *Int. J. Climatology*, **vol 21**, 1645-1668, DOI 10.1002/joc.703.

fastRegrid is a quick way to re-grid field object, and uses [EOF](#) to decompose the field into a small number of spatial patterns (EOFs), which are then re-gridded, and subsequently transformed back to a field object by calling [EOF2field](#).

Usage

```
catFields(field.1, field.2=NULL, lat=NULL, lon=NULL, plot.interp=FALSE,
          interval.1=NULL, interval.2=NULL, mon=NULL, demean=TRUE, silent=FALSE,
          fastregrid=FALSE, neofs=20)
fastRegrid(field.1, lat=NULL, lon=NULL, mon=NULL, neofs=20,
           silent=TRUE, plot.interp=FALSE)
testfastRegrid()
```

Arguments

field.1	A 'field.object'.
field.2	A 'field.object'. A 'field.2=NULL' processes single fields.
lat	Latitudes to extract. If NULL, use the latitudes from the first field. Otherwise interpolate both fields to latitudes.
lon	Longitudes to extract. See 'lat'.
plot.interp	Flag: 'TRUE' plots the interpolation results - Used for checking interpolation.
interval.1	Extract the time interval for the 1st field.
interval.2	Extract the time interval for the 2nd field.
mon	Calendar month or season to extract. eg January or DJF.
demean	Flag: 'TRUE' subtracts the mean values. This flag should be set to 'FALSE' if for instance two time slices are concatenated and the object is to investigate the mean change between these periods (see examples in eof.dc or eof.dmc).
silent	FALSE: verbose output.
fastregrid	Option in catFields - use fastRegrid
neofs	Number of EOF-modes to retain, determining the degree of details to be retained.

Value

A 'field.object'.

Author(s)

R.E. Benestad

Examples

```
## Not run:
library(clim.pact)
x.1 <- retrieve.nc("/home/kareb/data/ncep/ncep_t2m.nc",
                  x.rng=c(-60,40),y.rng=c(50,75))
x.2 <- retrieve.nc("/home/kareb/data/ncep/ncep_slp.nc",
                  x.rng=c(-60,40),y.rng=c(50,75))
print(x.1$v.name)

print("Read GCM predictor data.")
X.1 <- retrieve.nc("data/mpi-gsdiot2m.nc",
                  x.rng=c(-60,40),y.rng=c(50,75))
X.2 <- retrieve.nc("data/mpi-gsdiot_slp.nc",
                  x.rng=c(-60,40),y.rng=c(50,75))
print(X.1$v.name)
print("Cat fields.")
xX.1 <- catFields(x.1,X.1,interval.1=c(1958,1998),interval.2=c(1958,2050))
xX.2 <- catFields(x.2,X.2,interval.1=c(1958,1998),interval.2=c(1958,2050))
xX <- mixFields(xX.1,xX.2,mon=1,
               interval=c(1900,2050))
print("EOF")
eof.c <- eof(xX.1,mon=1)
eof.mc <- eof(xX,mon=1)

testfastRegrid()

data(DNMI.slp)
slp <- fastRegrid(DNMI.slp,lat=seq(30,80,by=1),lon=seq(-90,60,by=1),mon=1)

## End(Not run)
```

Description

Applies a canonical correlation analysis (CCA) to two data sets. The CCA here can be carried out in two different ways: i) an [svd](#) based approach (after Bretherton et al. (1992), J. Clim. Vol 5, p. 541, also documented in Benestad (1998): "Evaluation of Seasonal Forecast Potential for Norwegian Land Temperatures and Precipitation using CCA", DNMI KLIMA Report 23/98 at http://met.no/english/r_and_d_activities/publications/1998.html) or ii) a covariance-eigenvalue approach (after Wilks, 1995, "Statistical methods in the Atmospheric Sciences", Academic Press, p. 401).

The analysis can also be applied to either EOFs or fields.

Note: the analysis has sometimes been somewhat unstable, returning inconsistent results. The recommendation is to use EOFs and SVD option.

The CCA analysis can be used to develop statistical models according to:

$$Y = \Psi X$$

Where Y is the predictand and X the predictor. `plotCCA` plots the CCA results, `testCCA` is for code verification, and `Psi` returns the matrix

$$\Psi$$

`stations2field` turns a group of station objects into a field by the means of a simple and crude interpolation/gridding. `check.repeat` is a quality-control function that eliminates repeated years in the station objects.

Usage

```
CCA(x1,x2,SVD=TRUE,plot=TRUE,main="CCA",sub="",
    test=FALSE,i.eofs=1:8,LINPACK=TRUE)
plotCCA(cca,icca=1)
testCCA(method="CCA",reconstr=FALSE,mode=1,test=TRUE,
        LINPACK=TRUE,SVD=TRUE,n.pc=4,synthetic=TRUE)
Psi(cca)
predictCCA(Psi,X)
stations2field(data.set=c("narp"),ele=101,obj.type="monthly.field.object",
              plot=TRUE,silent=FALSE,intrp.method="interp",
              interpolation.option="simple")
check.repeat(x)
```

Arguments

<code>cca</code>	A CCA object.
<code>icca</code>	Choice of which CCA-pattern to plot.
<code>x1</code>	A field or an eof object.
<code>x2</code>	A field or an eof object.
<code>SVD</code>	Flag: determine which approach to use: SVD or eigenfunction-based algorithm.
<code>plot</code>	Flag: plot the diagnostics.
<code>test</code>	Flag: test by reconstructing one series (leading EOF or a grid-box series).
<code>i.eofs</code>	Which EOFs to include (only when the input is given as eof objects).
<code>LINPACK</code>	'TRUE': svd; 'FALSE':La.svd
<code>main</code>	main title (see <code>link{plot}</code>).
<code>sub</code>	subtitle (see <code>link{plot}</code>).
<code>method</code>	Which method to test: CCA or MVR.
<code>reconstr</code>	For the test-reconstruction of fields.
<code>mode</code>	Test for a particular EOF pattern/mode - the other modes are randomized.
<code>n.pc</code>	Number of principal components to include.

synthetic	Construct artificial test data from a random number generator and cosine series.
Psi	CCA-based prediction model (a matrix): $Y = \text{Psi } X$
X	A field object used as predictor: $Y = \text{Psi } X$
data.set	Strings (eg, "narp", "nordklim", or "nacd") if data has been installed, or list of station objects (and nothing else).
ele	element code - see getnordklim .
obj.type	Object type, e.g "monthly.field.object"
silent	FALG - verbose or not.
intrp.method	Method fro gridding the data.
interpolation.option	Either "simple", "distance", or "test". In current version, it's only "simple" which seems to work.
x	station object.

Value

A CCA object: a list containing a.m, b.m, u.k, v.k, and r, describing the Canonical Correlation variates, patterns and correlations. a.m and b.m are the patterns and u.k and v.k the vectors (time evolution).

Author(s)

R.E. Benestad

Examples

```
## Not run:
data(DNMI.t2m)
data(DNMI.slp)
eof.1 <- EOF(DNMI.t2m,mon=1)
eof.2 <- EOF(DNMI.slp,mon=1)
cca <- CCA(eof.1,eof.2)
# Testing routine:
testCCA()
```

```
## End(Not run)
```

`cdfcont`*netCDF names and dimensions.*

Description

Obtains the variable names and dimension names from a netCDF file. This routine provides some of the information returned by the old netCDF-package function `names.netCDF`, however this function does not depend on the netCDF package. If it doesn't work, make sure to set the path to include the location of `ncdump` and `ncgen` (two netCDF executables, usually under a `netcdf` directory). Alternatively, set the path.

Usage

```
cdfcont(filename,path="",method=NULL)
```

Arguments

<code>filename</code>	name of netCDF file.
<code>path</code>	Path to the location where <code>ncdump</code> and <code>ncgen</code> are located.
<code>method</code>	The name of the function making calls to the system. In Linux, it is 'system', in Windows 'shell'. If set to NULL, the code tries to decide between these two by probing using Sys.info

Value

a list object containing two character vectors: `vars` and `dims` holding the names of the variables and dimensions. `dims` gives the dimensions for each of the variable, and may contain several names within the same string, separated by a comma, eg. "LONS","LATS",and "LATS, LONS" (`strsplit("TIME, LATS, LONS",",", "")` can be used to split up these variables into separate names). `'time.origin'` returns the `time_origin`, `'add.offset'` gives the `'add_offset'` attribute, and `'scale.factor'` gives the `scale_factor`.

Author(s)

R.E. Benestad

Examples

```
## Not run: ncinfo <- cdfcont("test.nc")
```

cdfextract	<i>Extract a subfield from a netCDF file.</i>
------------	---

Description

A slow routine that extracts a subfield. This routine is suitable for reading subsections of large data files that are too big for [retrieve.nc](#).

This version uses [cdfcont](#) to obtain vital meta data for handling the data in the netCDF file and constructing a 'field' object.

Usage

```
cdfextract(filename,varname,x.rng=NULL,y.rng=NULL,t.rng=NULL,
           greenwich=TRUE,x.nam="lon",y.nam="lat",t.nam="tim",
           plot=TRUE,l.scale=TRUE)
```

Arguments

filename	name of netCDF file.
varname	name of variable.
x.rng	X-range (in degrees East): c(min,max).
y.rng	Y-range (in degrees North): c(min,max).
t.rng	T-range (in units stored in 'tim'): c(min,max).
greenwich	TRUE: longitude runs from -180E to 180E.
x.nam	name of X-dimension.
y.nam	name of Y-dimension.
t.nam	name of T-dimension.
plot	TRUE for plotting.
l.scale	scale field by offset and scaling-factor.

Value

a field object. Also see [retrieve.nc](#). Saves the extracted data in a netCDF file called "cdfextract.nc" under current working directory (see [r2cdf](#)).

Author(s)

R.E. Benestad

Examples

```
## Not run:
slp <- cdfextract("data/nmc_slp.nc","slp",x.rng=c(-80,40),y.rng=c(20,75),
                 t.rng=c(times[is],times[is]+499),plot=FALSE)

## End(Not run)
```

 coherence

Coherence spectrum - cross-spectrum analysis

Description

Based on: http://en.wikipedia.org/wiki/Wiener-Khinchin_theorem; Press et al. (1989) 'Numerical Recipes in Pascal', Cambridge, section 12.8 'Maximum Entropy (All Poles) Method'; von Storch & Zwiers (1999) 'Statistical Analysis in climate Research', Cambridge, section 11.4, eq 11.67, p. 235;

A test with two identical series the original equation (eq 11.67) from von Storch & Zwiers (1999) gave uniform values: 1. The denominator was changed from $(\Gamma_{xx} * \Gamma_{yy})$ to $(\sqrt{\Gamma_{xx} * \Gamma_{yy}})$.

Usage

```
coherence(x,y,dt=1,M=NULL,plot=TRUE)
testcoherence(x=NULL,y=NULL)
```

Arguments

x	A vector (time series).
y	A vector (time series).
dt	time increment - for plotting.
M	Window length - default= half series length
plot	Flag: plot the diagnostics.

Value

A complex vector .

Author(s)

R.E. Benestad

Examples

```
## Not run:
data(DNMI.t2m)
data(DNMI.slp)
eof.1 <- EOF(DNMI.t2m,mon=1)
eof.2 <- EOF(DNMI.slp,mon=1)
cca <- CCA(eof.1,eof.2)
# Testing routine:
testCCA()

data(oslo.dm)
testcoherence(oslo.dm$t2m,oslo.dm$slp)

## End(Not run)
```

composite.field	<i>Composite maps</i>
-----------------	-----------------------

Description

Produce composites of maps based on station observations, weights, or years. For a station object, assign +/- 1 weights according to whether the values stored in val (monthly station objects) are greater than mean + sd or less than mean - sd.

Usage

```
composite.field(x,y,lsig.mask=TRUE,sig.lev=0.05,s=0.42,mon=NULL,
               lty=1,col="black",lwd=1,main=NULL,sub=NULL)
compositeField(x,y,lsig.mask=TRUE,sig.lev=0.05,s=0.42,mon=NULL,
               lty=1,col="black",lwd=1,main=NULL,sub=NULL)
```

Arguments

x	A field object.
y	A station object, a vector consisting of [-1,0,+1] or a vector consisting of the years (negative values are used for negative phase, eg. c(1991,1993,1998,-1961,-1963,-1994)).
lsig.mask	FLAG: mask the regions not statistically significant.
sig.lev	Level of significance.
s	Threshold for defining whether y is high or low = $s * sd(y)$.
mon	Month to analyse.
lty	Contour line type.
col	Contour line colour.
lwd	Contour line width.
main	Main title.
sub	subtitle.

Value

A map object.

Author(s)

R.E. Benestad

Examples

```
## Not run:
slp <- retrieve.nc("ncep_slp.nc")
data(oslo.t2m)
composite.field(slp,oslo.t2m)

## End(Not run)
```

COn0E65N

*Convert long-lat to km-km***Description**

The function returns the distance in km from 0E 65N given the longitude and latitude. See also [km2lon](#) and [km2lat](#).

Usage

```
COn0E65N(lon, lat, lat.0=65,lon.0=0)
```

Arguments

lon	longitude
lat	latitude
lon.0	latitude for reference point
lat.0	latitude for reference point

Value

```
list(y=latitudes distance,x= longitudes distance)
```

Author(s)

R.E. Benestad

Examples

```
library(clim.pact)
data(oslo.t2m)
print(c(oslo.t2m$lon,oslo.t2m$lat))
#[1] 10.71667 59.95000
xy<-COn0E65N(oslo.t2m$lon,oslo.t2m$lat)
oslo.t2m$lon<-xy$x
oslo.t2m$lat<-xy$y
print(c(oslo.t2m$lon,oslo.t2m$lat))
#[1] 595.4086 -560.3004
lon<-km2lon(oslo.t2m$lon,oslo.t2m$lat,x.centre=0,y.centre=65)
lat<-km2lat(oslo.t2m$lon,oslo.t2m$lat,x.centre=0,y.centre=65)
```

```
print(c(lon,lat))
#[1] 10.71667 59.95000
```

corEOF *Field correlation*

Description

Produces maps of spatial correlation patterns from the EOF products

$$X = \mathbf{U}\mathbf{W}\mathbf{V}^T$$

according to following formula:

$$r(\mathbf{X}_{r,t}, \bar{y}) = \frac{\sum_k \mathbf{W}_k \mathbf{U}_{r,k} \mathbf{W}_k \sum_t [\mathbf{V}_{k,t}^T y'_t]}{\sqrt{\sum_k \mathbf{W}_k \mathbf{U}_{r,k} \mathbf{W}_k \sum_t (\mathbf{V}_{k,t}^T)^2 \times \sum_t (y'_t)^2}}$$

Reference: Environmental statistics for climate researchers <http://www.gfi.uib.no/~nilsg/kurs/notes/course.html>

NOTE: This routine is not finished yet - still contains some bugs, sometimes resulting in absolute correlation values greater than unity.

Usage

```
corEOF(x,y,lsig.mask=TRUE,sig.lev=0.05,neofs=20,
      lty=1,col="black",lwd=1)
```

Arguments

x	An EOF.
y	A station series.
lsig.mask	Mask out values that are not statistically significant.
sig.lev	Level of significance.
neofs	Number of modes to include.
lty	Contour line type.
col	Contour line colour.
lwd	Contour line width.

Value

A map object.

Author(s)

R.E. Benestad

Examples

```
data(oslo.t2m)
data(eof.slp)
corEOF(eof.slp,oslo.t2m)
```

corField

Field correlation

Description

Produces maps of spatial correlation patterns.

Usage

```
corField(x,y,lsig.mask=TRUE,sig.lev=0.05,mon=NULL,param="t2m",
         lty=1,col="black",lwd=1,main=NULL,z.levs=NULL,
         my.col=NULL,plot=TRUE)
```

Arguments

x	A field object.
y	A station series or a field object.
lsig.mask	Mask out values that are not statistically significant.
sig.lev	Level of significance.
mon	Month to analyse.
param	Used for daily station objects to decide which element to use.
lty	Contour line type.
col	Contour line colour.
lwd	Contour line width.
main	Title of plot, same as 'main' in plot().
z.levs	contour levels.
my.col	colour template (see rgb).
plot	TRUE for graphics output).

Value

A map object.

Author(s)

R.E. Benestad

Examples

```
## Not run:
slp <- retrieve.nc("ncep_slp.nc")
data(oslo.t2m)
corField(slp,oslo.t2m)

## End(Not run)
```

datestr2num	<i>datestr2num</i>
-------------	--------------------

Description

Converts a date string to number.

Usage

```
datestr2num(datestr,vec=TRUE)
```

Arguments

datestr	A date string (character). Types of strings may be "01-Jan-1980", "1-1-1980", "01-01-1980", "1980-1-1", "1980-Jan-1"
vec	TRUE returns a vector c(year, month, day); FALSE returns year + month/12 + day/31.

Value

A number or a vector

Author(s)

R.E. Benestad

Examples

```
datestr2num("01-Jan-1980")
datestr2num("1-1-1980")
datestr2num("01-01-1980")
datestr2num("1980-1-1")
datestr2num("1980-Jan-1")
```

delta	<i>Delta function</i>
-------	-----------------------

Description

The dirac delta function: delta (i,j) returns 1 if i==j, 0 otherwise.

Usage

```
delta(i, j)
```

Arguments

i	first index.
j	second index.

Value

0 or 1

Author(s)

R.E. Benestad

Examples

```
zero <- delta(1,2)
one <- delta(2,2)
```

dist2norm	<i>Transform a series to a normally distributed series.</i>
-----------	---

Description

Used for DS on non-Gaussian data. Tailored for the use with daily precipitation. dist2norm transforms (maps) data to normal distribution and norm2dist carries out the reverse operation. The mapping/transformation is done through one-to-one matching of the empirical distribution functions (e.d.f.) of the data and a Gaussian distribution.

Usage

```
dist2norm(x, plot=FALSE, exclude=NULL, sd=1, mean=0, force.zero=TRUE)
norm2dist(x, plot=FALSE, exclude=NULL, sd=1, mean=0, force.zero=TRUE)
```

Arguments

x	vector or a list-object returned by dist2norm.
plot	T -> plot for illustrating the mapping.
exclude	Values (e.g. 0) to exclude for the transform.
sd	The standard deviation of the transformed quantities.
mean	The mean value of the transformed quantities.
force.zero	Force the e.d.f. to start from 0.

Value

a list object of class 'dist2norm' containing a vector and the empirical transform functions.

Author(s)

R.E. Benestad

Examples

```
data(oslo.dm)
x <- dist2norm(oslo.dm$precip)
plot(x$xT)

y <- norm2dist(x,plot=TRUE)
```

distAB	<i>Distance between two points on Earth</i>
--------	---

Description

The function returns the distance between two points on Earth given by the lon-lat coordinates. The distance is computed using the formula: $d = a * \theta$, and $|x_1| * |x_2| * \cos(\theta) = \text{inner-product}(x_1, x_2)$.

Usage

```
distAB(lon, lat, lons, lats, a=6.378e06)
```

Arguments

lon	Longitude of reference point (degrees East).
lat	Latitude of reference point (degrees North).
lons	Longitude of points of interest (vector) (degrees East).
lats	Latitude of points of interest (vector) (degrees North).
a	Radius of the Earth.

Value

A real value: units=meters.

Author(s)

R.E. Benestad

Examples

```
distAB(10,60,5,58) # [1] 362802.3
distAB(0,0,180,0) # [1] 20037078
distAB(0,90,0,-90) # [1] 20037078
```

DNMI.t2m

Gridded monthly mean climate data

Description

Gridded monthly mean 2-meter temperature [T(2m)], sea level pressure (SLP), and sea surface temperature (SST) analysis for the North Atlantic region covering the period 1873 – 1998. The actual data is filtered through 20 EOFs (see [EOF2field](#)), and stored as eof products in order to save space. The lines listed in the example shows how these data have been generated (NB. old objects with the name eof, xxx, climxxx, and i may be overwritten).

Usage

```
data(DNMI.t2m)
data(DNMI.slp)
data(DNMI.sst)
```

Format

The data is a 'field' object (see [retrieve.nc](#)).

Source

The data is also available from <http://noserc.met.no>.

References

Benestad, R.E. and Melsom, A. (2002) Is there a link between the unusually wet autumns in southeastern Norway and SST anomalies?, *Climate Research* Vol 23, 67-79. \ Melsom E. and Benestad, R.E. : Is there a link between the unusually wet autumns in southeastern Norway and SST anomalies? PDF-version available on-line from <http://www.dams.dk/nmm2002/Melsom%20and%20Benestad.pdf> \ Benestad, R.E.: Analysis of gridded sea level pressure and 2-meter temperature for 1873-1998 based on UEA and NCEP re-analysis II (RegClim), DNMI KLIMA Report 03/00 PDF-version available on-line from <http://noserc.met.no/DS/klima0300.html>. For SST, see <http://noserc.met.no/DS/dnmissst.html>. The analysis of the SST was the same as the T(2m) and SLP.

Examples

```
## Not run:
library(clim.pact)
t2m <- retrieve.nc("~/data/analysis/DNMI_t2m.nc")
eof <- EOF(t2m)
save(file="clim.pact/data/eof_DNMI_t2m.Rdata",eof)
t2m.eof <- EOF2field(eof)

# Check if the EOF reconstruction reproduces the original field:
newFig()
plotField(t2m,lat=60,lon=10)
plotField(t2m.eof,lat=60,lon=10,add=TRUE,lty=2,col="red")
# Very similar time series

newFig()
mapField(t2m.eof)
mapField(t2m,add=TRUE,col="red",lty=2,lwd=1)
# Very similar spatial pattern/contours

sst <- retrieve.nc("~/data/analysis/DNMI_sst.nc")
eof <- EOF(sst)
save(file="clim.pact/data/eof_DNMI_sst.Rdata",eof)

slp <- retrieve.nc("~/data/analysis/DNMI_slp.nc")
eof <- EOF(slp)
save(file="clim.pact/data/eof_DNMI_slp.Rdata",eof)

## End(Not run)
```

Description

Identifies statistical relationships between large-scale spatial climate patterns and local climate variations for monthly and daily data series. Calibrates a linear regression model using step-wise screening and common EOFs (EOF) as basis functions. Evaluates the statistical relationship. Predicts local climate parameter from predictor fields. Works with ordinary EOFs, common EOFs (*catFields*) and mixed-common EOFs (*mixFields*). The rationale for using mixed-common EOFs is that the coupled structures described by the mixed-field EOFs may have a more physical meaning than EOFs of single fields [Benestad et al. (2002), "Empirically downscaled temperature scenarios for Svalbard", *Atm. Sci. Lett.*, doi.10.1006/asle.2002.0051].

The downscaling analysis returns a time series representing the local climate, patterns of large-scale anomalies associated with this, ANOVA, and analysis of residuals. Care must be taken when using this routine to infer local scenarios: check the R2 and p-values to check whether the calibration yielded an appropriate model. It is also important to examine the spatial structures of the large-scale anomalies associated with the variations in the local climate: do these patterns make physical sense? Experiment with both single and mixed fields. It is also a good

idea to check whether there are any structure in the residuals: if so, then a linear model for the relationship between the large and small-scale structures may not be appropriate. It is furthermore important to experiment with predictors covering different regions [ref: Benestad (2001), "A comparison between two empirical downscaling strategies", *Int. J. Climatology*, vol 21, Issue 13, pp.1645–1668. DOI 10.1002/joc.703]. There is a cautionary tale for how the results can be misleading if the predictor domain is not appropriate: domain for northern Europe used for sites in Greenland [ref: Benestad (2002), "Empirically downscaled temperature scenarios for northern Europe based on a multi-model ensemble", *Climate Research*, vol 21 (2), pp.105–125. <http://www.int-res.com/abstracts/cr/v21/n2/index.html>]

The function `ds()` is a generic routine which in principle works for when there is any real statistical relationship between the predictor and predictand. The predictand is therefore not limited to a climate variable, but may also be any quantity affected by the regional climate. *It is important to stress that the downscaling model must reflect a well-understood (physical) relationship.*

The trend-estimation uses regression to fit a 5th-order polynomial (in time) to fit the observed time series. The rate-of-change is estimated by taking the time-derivative of this equation. If

$$y = c_0 + c_1x + c_2x^2 + c_3x^3 + c_4x^4 + c_5x^5,$$

where x is the time, then the rate-of-change is:

$$y = c_1 + 2c_2x + 3c_3x^2 + 4c_4x^3 + 5c_5x^4.$$

[ref: Benestad (2002), What can present climate models tell us about climate change?, *Climatic Change*, accepted.]

The routine uses a step-wise regression (step) using the leading EOFs. The calibration is by default carried out on de-trended data [ref: Benestad (2001), "The cause of warming over Norway in the ECHAM4/OPYC3 GHG integration", *Int. J. Clim.*, 15 March, vol 21, p.371-387.].

The downscaled scenario is saved in a text file in the output directory (default: 'output').

The course notes from Environmental statistics for climate researchers <http://www.gfi.uib.no/~nilsg/kurs/notes/course.html> is a useful reference to statistical modelling and regression.

DS provides the basis for an 'objective' downscaling though `objDS`.

Changes - 02.02.2005: Improved the representation of the correct constant level: `# pre.gcm <- pre.gcm - gcm.mean + obs.mean2 # REB 26.08.2004: 'obs.mean' replaced with 'obs.mean2' pre.gcm <- pre.gcm - cal.mean + obs.mean2 # REB 02.02.2005: 'gcm.mean' replaced with 'cal.mean'`

Usage

```
DS(dat,preds,mon=NULL,direc="output",cal.id=NULL,
    ldetrnd=TRUE,i.eofs=seq(1,8,by=1),ex.tag="",
    method="lm",plot=TRUE,leps=FALSE,param="t2m",
    plot.res=FALSE,plot.rate=FALSE,xtr.args="",
    swsm="step",predm="predict",lsave=FALSE,rmac=TRUE,
    silent=FALSE,exit.on.screening.failure=FALSE)
```

Arguments

<code>dat</code>	A climate.station object (<code>station.obj</code> or <code>station.obj.dm</code>). [e.g. from <code>getnacd</code> , <code>getnordklim</code> or <code>station.obj</code>].
<code>preds</code>	The predictor EOF.
<code>mon</code>	month or season to downscale, this is automatically changes if predictor only contains a different month (this is normally a redundant feature).
<code>direc</code>	name of directory inwhich the output is dumped (e.g. figures, tables).
<code>cal.id</code>	ID tag used for calibration. By default use the first field (<code>catFields</code>) for calibration.
<code>ldetrnd</code>	F for no detrending; T for removing linear trends before model calibration.
<code>i.eofs</code>	select which EOFs to include in the setp-wise screening.
<code>ex.tag</code>	Extra labelling tag for file names for experiments.
<code>method</code>	Sets the method to use for regression. Method is set to "lm" by default, but "anm" allows the incorporation of an analog model, see <code>anm</code> . "anm.weight" weights the principal components according to the eigenvalues, whereas "anm" uses unweighted series.
<code>plot</code>	'TRUE' produces figures.
<code>leps</code>	'TRUE' produces EPS figures (files).
<code>param</code>	Name of parameter (for plot labels).
<code>plot.res</code>	'TRUE' shows statistics for residuals.
<code>plot.rate</code>	'TRUE' shows analysis of rate-of-change.
<code>xtr.args</code>	Extra/additional arguments in the formula. E.g. for <code>method=='glm'</code> , one can pass on <code>'xtr.args="family='gaussian'"</code> . Can be used to pass on the <code>family</code> argument to <code>glm</code> .
<code>swsm</code>	Step-wise screening method, default=='step'; 'none' skips stepwise sceening.
<code>predm</code>	Prediction method, default is "predict".
<code>lsave</code>	TRUE -> saves the result on disc.
<code>rmac</code>	TRUE -> subtracts (removes) the annual cycle in station data.
<code>silent</code>	TRUE -> no output to screen..
<code>exit.on.screening.failure</code>	If stepwise screening returns no variables, then return()

Value

A 'ds' object - a list of following elements:

<code>X.1 .. X.n</code>	1..nth predictor pattern for n fields (<code>mixFields</code>).
<code>lon.1 ..</code>	Longitude coordinate of spatial fields (a vector).
<code>lat.1 ..</code>	Latitude coordinate of spatial fields (a vector).
<code>n fld</code>	Number of fields (different types of predictors, <code>mixFields</code>).
<code>unit</code>	Unit of quantity in station series.
<code>pred.name</code>	Name of predictor.

lon.loc	Longitude of predictand location.
lat.loc	Latitude of predictand location
yy.gcm	Years corresponding to scenario (GCM).
mm.gcm	Months corresponding to scenario (GCM).
dd.gcm	Days corresponding to scenario (GCM).
yy.cal	Years corresponding to observation (Calibration).
mm.cal	Months corresponding to observation (Calibration).
dd.cal	Days corresponding to observation (Calibration).
yy.o	Years corresponding to station series (obs.).
mm.o	Months corresponding to station series (obs.).
dd.o	Days corresponding to station series (obs.).
rate.ds	Estimated linear rate of change of downscaled scenario.
rate.err	Error estimate for rate.ds.
gcm.trnd.p	P-value of linear trend in downscaled scenario.
fit.p	ANOVA p-value for fit between large-scale and small-scale variability(from regression analysis).
fit.r2	ANOVA R2 for fit between large-scale and small-scale variability (from regression analysis).
pre.gcm	The downscaled scenario (a vector).
pre.y	The downscaled results using the calibration data.
location	Nsme of location of predictor.
gcm.stat	ANOVA of linear trend fit to scnrario.
month	Month of study (0-> all months).
v.name	Name of downscaled element.
region	Region used for downscaling.
pre.fit	Linear fit to prediction (downscaled scenario) (a vector).
pre.p.fit	Polynomial fit to the downscaled scenario.
tr.est.p.fit	Rate of change derived from a fifth-order polynomial trend-fit to prediction (downscaled scenario) (a vector).
id.1, id.2	IDs labelling which data was used for calibration (id.1).

Author(s)

R.E. Benestad

Examples

```
library(clim.pact)
data("oslo.t2m")
data("eof.mc")
a<-DS(dat=oslo.t2m,preds=eof.mc,plot=FALSE)
## Not run:
# Example 1: for computing common EOFs and using these as a basis for DS.
slp.obs <- retrieve.nc("ncep_slp.nc",x.rng=c(-20,40),y.rng=c(50,70))
# Get gridded observations/analysis from NCEP
slp.gcm <- retrieve.nc("EH40PYC_B2_slp.nc") # Get results from climate models
slp <- catFields(slp.obs,slp.gcm) # combine the fields.
eof <- EOF(slp,mon=1)
obs <- getnordklim("Stockholm")
ds <- DS(preds=eof,obs)
```

```

# Example 2:
# A demonstration for the linear regression model: monthly values
library(clim.pact)

# Read the gridded netCDF data:

t2m<-retrieve.nc("DNMI_t2m.nc")

# Manipulate the data: assign one part as calibration and one part as
# independent data
nt<-length(t2m$tim)
t2m$id.t[1:floor(nt/2)]<-"calibrate"
t2m$id.t[ceiling(nt/2):nt]<-"independent"

# Compute EOFs
eof<-EOF(t2m,mon=1,neofs=3)
plotEOF(eof)

# Get the predictand: a local station series
obs<-getnordklim("Goeteborg",ele=101)

# Apply the downscaling
DS(preds=eof,obs)
plotStation(obs,mon=1,add=TRUE,col="darkgreen",lwd=1,lty=2)

# Example 3: A demonstration for the linear regression model: daily values
# These files are not distributed with the clim.pact package, but
# nevertheless demonstrate how the downscaling can be done with a few
# clim.pact functions.

library(clim.pact)
data(eof.dc)
list<-read.table("data/daily/station.list.good",header=TRUE)
print(list)

i<-as.numeric(readline("Which number? (1-37)"))
obs1<-read.table(paste("data/daily/",list$file.name[i],sep=""))

obs<-station.obj.dm(t2m=obs1$V5,precip=obs1$V6,yy=obs1$V4,mm=obs1$V3,dd=obs1$V2,
                    station=obs1$V1[1],location=as.character(list$location[i]),
                    lon=list$lon[i],lat=list$lat[i],alt=list$alt[i],
                    obs.name<-c("t2m","precipitation"))

plotStation(obs)
DS(preds=eof.dc,obs)

```

Example 4: A demonstration for the analog model: daily values

```
library(clim.pact)
library(anm)
source("clim.R")
data(eof.dc)
list<-read.table("data/daily/station.list.good",header=TRUE)
print(list)

i<-as.numeric(readline("Which number? (1-37)"))
obs1<-read.table(paste("data/daily/",list$file.name[i],sep=""))

obs<-station.obj.dm(t2m=obs1$V5,precip=obs1$V6,yy=obs1$V4,mm=obs1$V3,dd=obs1$V2,
                  station=obs1$V1[1],location=as.character(list$location[i]),
                  lon=list$lon[i],lat=list$lat[i],alt=list$alt[i],
                  obs.name<-c("t2m","precipitation"))

plotStation(obs)

print("Please be patient - this takes a while...")
ds-anm<-DS(preds=eof.dc,obs,method="anm.weight",swsm="none",
          predm="predict.anm",param="precip",
          lsave=FALSE,ldetrnd=FALSE)

## End(Not run)
```

ds2station *Convert ds to station object*

Description

Convert 'ds' object to 'station' object so that downscaled scenarios can be handled like station series.

Usage

```
ds2station(x,what="scenario")
```

Arguments

x	a 'ds' object (see DS).
what	either 'scenario' or 'calibration'

Value

a station object

Author(s)

R.E. Benestad

Examples

```
data(eof.c)
data(oslo.t2m)
ds <- DS(preds=eof.c, oslo.t2m,plot=FALSE)
oslo.dm <- ds2station(ds)
plotStation(oslo.dm)
```

DSpdf.exp

*Downscale eponential PDF.***Description**

Functions handy for downscaling PFDs.

DSpdf.exp() is used to downscale exponential distributions for daily rainfall according to:

Benestad, R.E. (2007) Climate Research, CR34:195-210, doi: 10.3354/cr00693 (http://www.int-res.com/articles/cr_oa/c034p195.pdf) Benestad, R.E., C. Achberger, & E. Fernandez (2005) 'Empirical-statistical downscaling of distribution functions for daily precipitation', met-no report, 12-2005, Climate, pp.43 (http://met.no/english/r_and_d_activities/publications/2005/12_2005/abstract_12_2005.html). The dataset exp.par is used for this downscaling.

CDFtransfer() is a function that facilitates local quantile transformation.

empiricalRanking() uses a formula for estimating the cumulative probability P corresponding to rank m References: Jenkinson, A.F., 1977, U.K. Met.Office Synoptic Clim. Branch Memo 58; Beard, L.R., 1943, Trans. Amer. Meteor. Soc. Civ. Eng., 108, 1110-1160; Chegodaev, N.N., 1953 (in Russian) State Rail Transport Publishing House; Folland, C. and Anderson, C. (2002), J. Clim. 15, 2954-2960, equation (1)

Usage

```
DSpdf.exp(obs=NULL,dT=0,dP=0,plot=TRUE,year=NULL,month=NULL,quadratic=TRUE)
data(exp.par)
CDFtransfer(Y,CDF.2,CDF.1=NULL,method="empiricalRanking",plot=FALSE,
            silent=FALSE,smooth=TRUE,xlab="x2",ylab="x1")
empiricalRanking(x)
```

Arguments

x	A field object.
plot	TRUE: plot.
obs	daily station object
dT	Projected change in annual mean T(2m), e.g. from objDS (unit deg C).
dP	Projected change in annual mean precipitation, e.g. from objDS (units: mm/day!).
year	Year to extract
month	Month to extract
Y	A data series or daily station object.

CDF.1	If provided, assume this CDF to describe the cumulative distribution of Y.
CDF.2	cumulative distribution predicted for the future.
method	Method for estimating empirical distribution function (EDF).
silent	TRUE: less clutter on the screen.
smooth	Uses a 9th-order polynomial fit to provide a smoother fit for the local quantile transformation.
quadratic	Flag to toggle between the linear or the quadratic regression model in Benestad (2007)
xlab	see plot
ylab	see plot

Value

A field object.

Author(s)

R.E. Benestad

Examples

```
## Not run:
data(exp.par)
data(oslo.dm)
a<-DSpdf.exp(oslo.dm,dT=3,dP=1)
F1<- list(x=a$x,P=a$Fx.obs)
F2<- list(x=a$x,P=a$Fx.chg)
y<-CDFtransfer(Y=oslo.dm$precip,CDF.2=F2,CDF.1=F1,plot=TRUE)
plot(oslo.dm$precip,y)

## End(Not run)
```

EOF

Empirical Orthogonal Functions (EOFs).

Description

Computes EOFs (a type of principal component analysis) for combinations of data sets, typically from the NCEP reanalysis and corresponding data from climate models. Preprocessing by [catFields](#) allows for common EOF analysis [ref: Benestad (2001), "A comparison between two empirical downscaling strategies", *Int. J. Climatology*, vol 21, Issue 13, pp.1645-1668. DOI 10.1002/joc.703]. and [mixFields](#) prepares for mixed-field EOF analysis [ref. Bretherton et al. (1992) "An Intercomparison of Methods for finding Coupled Patterns in Climate Data", *J. Climate*, vol 5, 541-560; Benestad et al. (2002), "Empirically downscaled temperature scenarios for Svalbard", *Atm. Sci. Lett.*, doi.10.1006/asle.2002.0051].

Uncertainty estimates are computed according to North et al. (1982), "Sampling Errors in the Estimation of Empirical Orthogonal Functions", *Mon. Weather Rev.*, **vol 110**, 699-706.

NB: This routine may be computer-intensive! The computation of the EOFs tends to take some time, especially on computers/PCs with little memory (less than 128Mb) and slow processors less than 800MHz.

See the course notes from Environmental statistics for climate researchers <http://www.gfi.uib.no/~nilsg/kurs/notes/course.html> for a discussion on EOF analysis.

ExtEOF() computes extended EOFs.

lagField() introduces a time lag in the field.

EOF2field() reconstructs a field from EOF products. It's a useful routine for filtering out small-scale structure noise if the number of EOFs is low - thus including only the few most important modes. The function is also useful in conjunction with MVR to reconstruct the field objects if the regression is performed on EOF objects. The EOF2field function is used when reading the gridded data DNMI.sst/DNMI.t2m/DNMI:slp, and is used to decompress data stored compressed as EOF-products (note: some detail is lost this way).

Usage

```
EOF(fields,l.wght=TRUE,lc180e=FALSE,direc="data/",
     lon=NULL,lat=NULL,l.stndrd=TRUE,las=1,
     mon=NULL,plot=TRUE,neofs=20,l.rm.ac=TRUE,lsave=FALSE,
     LINPACK=TRUE,silent=FALSE)
ExtEOF(fields,lag=1,mon=NULL,lon=NULL,lat=NULL)
lagField(fields,lag=1)
EOF2field(eof,anomalies=FALSE)
```

Arguments

fields	A field object (eg from retrieve.nc).
l.wght	'TRUE' applies a geographical weighting.
lc180e	'TRUE' centers the maps on date line (180 deg E).
direc	Directory for the output.
lon	longitudinal region of interest.
lat	latitudinal region of interest.
l.stndrd	Not yet used.
las	Used by filled.contour , see par .
mon	Month (1-12) [season (1-4) for daily data] to extract.
plot	'TRUE' plots the results.
neofs	Number of leading EOFs to retain.
l.rm.ac	'TRUE' removes the annual cycle.
lsave	'TRUE' dumps the results to file.
LINPACK	'TRUE': svd; 'FALSE':La.svd
silent	'TRUE': quiet mode

lag	Lag in months or days (using the time unit) introduced to the field. Used with mixFields to estimate esxtnded EOFs.
eof	An 'EOF' object from EOF .
anomalies	True if only anomalies are to be shown.

Value

File containing an 'eof.c' object:

EOF	EOF patterns.
W	Eigenvalues.
dW	uncertainty in Eigenvalues (after North et al., 1982, eq 24: $dW \approx W \sqrt{2/N}$).
PC	Principal components of common PCA.
n.fld	Number of different predictors (see mixFields).
tot.var	Sum of all W squared.
id.t	Time labels for the fields (see catFields) - used in DS .
id.x	Spatial labels for the fields (see mixFields) - used in plotEOF .
id.lon	Spatial labels for the fields (see mixFields) - used in plotEOF .
id.lat	Spatial labels for the fields (see mixFields) - used in plotEOF .
region	Describes the region analysed.
tim	Time information (usually redundant).
lon	Longitudes associated with EOF patterns.
lat	Latitudes associated with EOF patterns.
var.eof	Fractional variances associated with EOF patterns.
yy	years.
mm	months.
dd	days.
v.name	Name of element.
c.mon	Month-season information.
f.name	File name of original data.
clim	mean values (climatology)
attributes	Attributes

The data is also saved as files.

Author(s)

R.E. Benestad

Examples

```
# Computes a set of mixed-common EOFs (overnight work..). This takes a while...
## Not run:
library(clim.pact)
x.1 <- retrieve.nc("/home/kareb/data/ncep/ncep_t2m.nc",
                  x.rng=c(-60,40),y.rng=c(50,75))
x.2 <- retrieve.nc("/home/kareb/data/ncep/ncep_slp.nc",
                  x.rng=c(-60,40),y.rng=c(50,75))
```

```

print(x.1$v.name)

print("Read GCM predictor data.")
X.1 <- retrieve.nc("data/mpi-gsdiot2m.nc",
                  x.rng=c(-60,40),y.rng=c(50,75))
X.2 <- retrieve.nc("data/mpi-gsdiot_slp.nc",
                  x.rng=c(-60,40),y.rng=c(50,75))
print(X.1$v.name)
print("Cat fields.")
xX.1 <- cat.fields(x.1,X.1,interval.1=c(1958,1998),interval.2=c(1958,2050))
xX.2 <- cat.fields(x.2,X.2,interval.1=c(1958,1998),interval.2=c(1958,2050))
xX <- mix.fields(xX.1,xX.2,mon=1,
                interval=c(1900,2050))
print("EOF")
eof.c <- EOF(xX.1,mon=1)
eof.mc <- EOF(xX,mon=1)

print("Reconstruct the field from EOFs can filter out small-scale
structure noise:")
EOF2field(xX.1) -> xX.1.filtered

load("data/ceof.Rdata") # loads the object 'eof'
field <- EOF2field(eof)

## End(Not run)

```

eof.c

Monthly common EOF.

Description

Common EOFs for January mean 2-meter temperature (T(2m)).

Usage

```
data(eof.c)
```

Format

EOF	EOF patterns.
W	Eigen values.
PC	Principal components of common PCA.
n.fld	Number of different predictors (see mixFields).
tot.var	Sum of all W squared.
id.t	Time labels for the fields (see catFields) - used in DS .
id.x	Spatial labels for the fields (see mixFields) - used in plotEOF .
id.lon	Spatial labels for the fields (see mixFields) - used in plotEOF .
id.lat	Spatial labels for the fields (see mixFields) - used in plotEOF .
region	Describes the region analysed.

tim	Time information (usually redundant).
lon	Longitudes associated with EOF patterns.
lat	Latitudes associated with EOF patterns.
var.eof	Fractional variances associated with EOF patterns.
yy	years.
mm	months.
dd	days.
v.name	Name of element.
c.mon	Month-season information.
f.name	File name of original data.

Source

The common EOF was produced using [EOF](#), with the combined January 2-meter air temperature data field from National Center for Environmental Prediction (NCEP; USA) reanalysis (Kalnay et al., (1996) "The NCEP/NCAR 40-Year Reanalysis Project", *Bul. Am. Met. Soc.*, **vol 77**, no 3, 437-471; e.g. see URL: <http://www.cdc.noaa.gov/index.html>) and the ECHAM4-GSDIO scenario (Max-Planck Institute for Meteorology, Hamburg, Germany; URL: <http://www.mpimet.mpg.de/>). The region is 60W - 40E, 50N - 75N.

References

Reference to methodology: R.E. Benestad (2001), "A comparison between two empirical down-scaling strategies", *Int. J. Climatology*, **vol 210**, pp.1645-1668. [DOI 10.1002/joc.703].

Examples

```
library(clim.pact)
data(eof.c)
```

eof.dc

Daily common EOF.

Description

Common EOFs for daily December-February 2-meter temperature (T(2m)).

Usage

```
data(eof.dc)
```

Format

EOF	EOF patterns.
W	Eigen values.
PC	Principal components of common PCA.
n.fld	Number of different predictors (see mixFields).
tot.var	Sum of all W squared.
id.t	Time labels for the fields (see catFields) - used in DS .
id.x	Spatial labels for the fields (see mixFields) - used in plotEOF .
id.lon	Spatial labels for the fields (see mixFields) - used in plotEOF .
id.lat	Spatial labels for the fields (see mixFields) - used in plotEOF .
region	Describes the region analysed.
tim	Time information (usually redundant).
lon	Longitudes associated with EOF patterns.
lat	Latitudes associated with EOF patterns.
var.eof	Fractional variances associated with EOF patterns.
yy	years.
mm	months.
dd	days.
v.name	Name of element.
c.mon	Month-season information.
f.name	File name of original data.

Source

The common EOF was produced using [EOF](#), with the combined December-February (DJF) 2-meter air temperature data field from European Centre for Medium-Range Weather Forecasts (ECMWF; UK) reanalysis (see URL: <http://www.ecmwf.int/>) and HIRHAM dynamically downscaled scenarios from the ECHAM4-GSDIO scenario (Max-Planck Institute for Meteorology, Hamburg, Germany; URL: <http://www.mpimet.mpg.de/>). The region is 5E - 25E, 58N - 65N.

References

Reference to methodology: R.E. Benestad (2001), "A comparison between two empirical down-scaling strategies", *Int. J. Climatology*, **vol 210**, pp.1645-1668. [DOI 10.1002/joc.703].

Examples

```
#The EOFs were produced using the following code:
library(clim.pact)
## Not run:
x.1.dm<-retrieve.nc("/data1/era15/ERA-15_t2m.nc",x.rng=c(5,25),y.rng=c(58,65))
X.1.dm<-retrieve.nc("/data1/hirham/T2M_198001-199912.nc",x.rng=c(5,25),
                    y.rng=c(58,65))
Y.1.dm<-retrieve.nc("/data1/hirham/T2M_203001-204912.nc",x.rng=c(5,25),
                    y.rng=c(58,65))
```

```

Y.1.dm$yy <- Y.1.dm$yy + 50
# It is important that demean=FALSE when concatenating the two time slices
# from the model simulations, if a study of climate change is the objective.
xX.1.dm <- catFields(X.1.dm,Y.1.dm,demean=FALSE)
xX.1.dm <- catFields(x.1.dm,xX.1.dm)
eof.dc <- eof(xX.1.dm,mon=1)

## End(Not run)
# To read the data:
data(eof.dc)

```

eof.dmc

Daily common EOF.

Description

Common EOFs for daily December-February 2-meter temperature (T(2m)) and sea level pressure (SLP).

Usage

```
data(eof.dmc)
```

Format

EOF	EOF patterns.
W	Eigen values.
PC	Principal components of common PCA.
n.fld	Number of different predictors (see mixFields).
tot.var	Sum of all W squared.
id.t	Time labels for the fields (see catFields) - used in DS .
id.x	Spatial labels for the fields (see mixFields) - used in plotEOF .
id.lon	Spatial labels for the fields (see mixFields) - used in plotEOF .
id.lat	Spatial labels for the fields (see mixFields) - used in plotEOF .
region	Describes the region analysed.
tim	Time information (usually redundant).
lon	Longitudes associated with EOF patterns.
lat	Latitudes associated with EOF patterns.
var.eof	Fractional variances associated with EOF patterns.
yy	years.
mm	months.
dd	days.
v.name	Name of element.
c.mon	Month-season information.
f.name	File name of original data.

Source

The common EOF was produced using EOF, with the combined December-February (DJF) 2-meter air temperature data field from European Centre for Medium-Range Weather Forecasts (ECMWF; UK) reanalysis (see URL: <http://www.ecmwf.int/>) and HIRHAM dynamically downscaled scenarios from the ECHAM4-GSDIO scenario (Max-Planck Institute for Meteorology, Hamburg, Germany; URL: <http://www.mpimet.mpg.de/>). The region is 5E - 25E, 58N - 65N.

References

Reference to methodology: R.E. Benestad (2001), "A comparison between two empirical down-scaling strategies", *Int. J. Climatology*, **vol 210**, pp.1645-1668. [DOI 10.1002/joc.703].

Examples

```
library(clim.pact)
## Not run:
x.1.dm<-retrieve.nc("/data1/era15/ERA-15_t2m.nc",x.rng=c(5,25),y.rng=c(58,65))
X.1.dm<-retrieve.nc("/data1/hirham/T2M_198001-199912.nc",x.rng=c(5,25),
                    y.rng=c(58,65))
Y.1.dm<-retrieve.nc("/data1/hirham/T2M_203001-204912.nc",x.rng=c(5,25),
                    y.rng=c(58,65))
Y.1.dm$yy <- Y.1.dm$yy + 50
# It is important that demean=FALSE when concatenating the two time slices
# from the model simulations, if a study of climate change is the objective.
xX.1.dm <- catFields(X.1.dm,Y.1.dm,demean=FALSE)
xX.1.dm <- catFields(x.1.dm,xX.1.dm)
x.2.dm<-retrieve.nc("/data1/era15/ERA-15_slp.nc",x.rng=c(5,25),y.rng=c(58,65))
X.2.dm<-retrieve.nc("/data1/hirham/PSL_198001-199912.nc",x.rng=c(5,25),
                    y.rng=c(58,65))
Y.2.dm<-retrieve.nc("/data1/hirham/PSL_203001-204912.nc",x.rng=c(5,25),
                    y.rng=c(58,65))
Y.2.dm$yy <- Y.2.dm$yy + 50
# It is important that demean=FALSE when concatenating the two time slices
# from the model simulations, if a study of climate change is the objective.
xX.2.dm <- catFields(X.2.dm,Y.2.dm,demean=FALSE)
xX.2.dm <- catFields(x.2.dm,xX.2.dm)

xX.dm <- mix.fields(xX.1.dm,xX.2.dm,mon=1)
eof.dmc <- eof(xX.dm,mon=1)

## End(Not run)
# To read the data:
data(eof.dmc)
```

eof.mc

*Monthly mixed-common EOF.***Description**

Monthly mixed-common EOFs for January mean 2-meter temperature (T(2m)) and sea-level-Pressure (SLP).

Usage

```
data(eof.mc)
```

Format

EOF	EOF patterns.
W	Eigen values.
PC	Principal components of common PCA.
n.fld	Number of different predictors (see mixFields).
tot.var	Sum of all W squared.
id.t	Time labels for the fields (see catFields) - used in DS .
id.x	Spatial labels for the fields (see mixFields) - used in plotEOF .
id.lon	Spatial labels for the fields (see mixFields) - used in plotEOF .
id.lat	Spatial labels for the fields (see mixFields) - used in plotEOF .
region	Describes the region analysed.
tim	Time information (usually redundant).
lon	Longitudes associated with EOF patterns.
lat	Latitudes associated with EOF patterns.
var.eof	Fractional variances associated with EOF patterns.
yy	years.
mm	months.
dd	days.
v.name	Name of element.
c.mon	Month-season information.
f.name	File name of original data.

Source

The common EOF was produced using [EOF](#), with the combined January 2-meter air temperature and sea level pressure data field from National Center for Environmental Prediction (NCEP; USA) reanalysis (Kalnay et al., (1996) "The NCEP/NCAR 40-Year Reanalysis Project", *Bul. Am. Met. Soc.*, vol 77, no 3, 437-471; e.g. see URL: <http://www.cdc.noaa.gov/index.html>) and the ECHAM4-GSDIO scenario (Max-Planck Institute for Meteorology, Hamburg, Germany; URL: <http://www.mpimet.mpg.de/>). The region is 60W - 40E, 50N - 75N.

References

Reference to methodology: R.E. Benestad (2001), "A comparison between two empirical down-scaling strategies", *Int. J. Climatology*, vol 210, pp.1645-1668. [DOI 10.1002/joc.703].

Examples

```
library(clim.pact)
data(eof.mc)
```

eof.slp

EOF of NCEP reanalysis SLP.

Description

EOF of NCEP reanalysis January mean sea level pressure

Usage

```
data(eof.slp)
```

Format

EOF	EOF patterns.
W	Eigen values.
PC	Principal components of common PCA.
n.fld	Number of different predictors (see mixFields).
tot.var	Sum of all W squared.
id.t	Time labels for the fields (see catFields) - used in DS .
id.x	Spatial labels for the fields (see mixFields) - used in plotEOF .
id.lon	Spatial labels for the fields (see mixFields) - used in plotEOF .
id.lat	Spatial labels for the fields (see mixFields) - used in plotEOF .
region	Describes the region analysed.
tim	Time information (usually redundant).
lon	Longitudes associated with EOF patterns.
lat	Latitudes associated with EOF patterns.
var.eof	Fractional variances associated with EOF patterns.
yy	years.
mm	months.
dd	days.
v.name	Name of element.
c.mon	Month-season information.
f.name	File name of original data.

Source

The common EOF was produced using [EOF](#), with the combined January 2-meter air temperature data field from National Center for Environmental Prediction (NCEP; USA) reanalysis (Kalnay et al., (1996) "The NCEP/NCAR 40-Year Reanalysis Project", *Bul. Am. Met. Soc.*, **vol 77**, no 3, 437-471; e.g. see URL: <http://www.cdc.noaa.gov/index.html>).

References

Reference to methodology: R.E. Benestad (2001), "A comparison between two empirical down-scaling strategies", *Int. J. Climatology*, **vol 210**, pp.1645-1668. [DOI 10.1002/joc.703].

Examples

```
library(clim.pact)
data(eof.slp)
```

```
getClimateExplorer      Retrieve station record through the KNMI ClimateExplorer.
```

Description

Retrieve data through: URL <http://climexp.knmi.nl/>. Also see [getdnmi](#), [getnordklim](#), and [getnaccd](#).

Usage

```
getClimateExplorer(location,URL="http://climexp.knmi.nl/data/",ele.c="t")
```

Arguments

location	ClimateExplorer station number.
ele.c	name of element [e.g. avail.elem(), or 't', 'p', 'l'].
URL	URL for the data report.

Value

a <- list of "monthly.station.record" class:

val	The monthly values (a 12-column matrix with one column for each year).
station	station number.
yy	The years of observation (vector).
lat,lon	Latitude and longitude of the location.
x.0E65N,y.0E65N	Distance in km from 0E, 65N.
location	Name of location .
wmo.no	WMO number.
start	Start of observatins from this location.
yy0	First year in current record.

ele	Code of the element.
obs.name	Name of the element.
unit	Unit of the element.
country	The country in which the location is located.
quality	Code/description for data quality.
found	Flag: T - the data requested was found.
ref	Reference for the data set.

Author(s)

R.E. Benestad

Examples

```
## Not run:
greenwich <- getClimateExplorer(3779.1)
plotStation(greenwich, what="t")
all.station.list <- getClimateExplorer()

## End(Not run)
```

getdnmi

Retrieve station record from DNMI database filed.

Description

Retrieve station record from DNMI (The Norwegian Meteorological Institute, met.no) database filed. URL <http://www.met.no>. Also see [getnacd](#) and [getnordklim](#).

Usage

```
getdnmi(location, ele.c='101', silent = FALSE, direc="data/")
```

Arguments

location	name of climate station location.
ele.c	name of element [e.g. avail.elem(), or 't2m', 'rr', 'slp'].
silent	TRUE: no printing
direc	Defunct argument.

Value

a list of "monthly.station.record" class:

val	The monthly values (a 12-column matrix with one column for each year).
station	station number.
yy	The years of observation (vector).
lat,lon	Latitude and longitude of the location.
x.0E65N,y.0E65N	Distance in km from 0E, 65N.
location	Name of location .
wmo.no	WMO number.
start	Start of observatins from this location.
yy0	First year in current record.
ele	Code of theelement.
obs.name	Name of the element.
unit	Unit of the element.
country	The country in which the location is located.
quality	Code/description for data quality.
found	Flag: T - the data requested was found.
ref	Reference for the data set.

Author(s)

R.E. Benestad

Examples

```
## Not run:
oslo.t2m.dnmi <- getdnmi("oslo")
ferder.t2m.dnmi <- getdnmi("ferder")

## End(Not run)
```

getecsn

Retrieve daily station record from ECSN set.

Description

Reads the data from Nordklim available at URL: <http://eca.knmi.nl/dailydata/predefinedseries.php>. Also see <http://www.eumetnet.eu.org/contecsn.html> and [getgiss](#).

THESE DATA CAN BE USED FREELY PROVIDED THAT THE FOLLOWING SOURCE IS ACKNOWLEDGED: Klein Tank, A.M.G. and Coauthors (2002) 'Daily dataset of 20th-century surface air temperature and precipitation series for the European Climate Assessment.' *Int. J. of Climatol.*, **22**, 1441-1453. Data and metadata available at <http://eca.knmi.nl>

Usage

```
getecsn(location="prompt",param=c("TG","RR"),dataset="blended",
        data.path="data.eca",country=NULL,update = FALSE)
```

Arguments

location	name of climate station location.
param	name of element [e.g. avail.elem(), or 't2m', 'rr', 'slp'].
dataset	"blended" (GCOS + EUMETNET), "gcos", or ."eumetnet"
data.path	name of directory in which the data are stored.
country	Country to search for stations.
update	For downloading new data.

Value

a <- list of "daily.station.record" class.

Author(s)

R.E. Benestad

Examples

```
## Not run:
obs <- getecsn("helsinki")
plotStation(obs)

## End(Not run)
```

getgiss

Retrieve station record from the GISS or NARP data set from URL.

Description

Reads the data from GISS available at URL: <http://www.giss.nasa.gov/data/update/gistemp/> or NARP data from http://projects.met.no/~narp/data_index.html (<http://thule.oulu.fi/narp/>). Also see [getnordklim](#), [getdnmi](#) and [getnacd](#).

Use eg options(timeout=120) if problems with retrieving the GISS data. Some times, the code doesn't manage to read the data, and returns with the message: "cannot open: HTTP status was '404 Not Found'". Try again, since slow connections cause this problem. One trick that often helps is to use the function to get the name of the station (stations <- getgiss()), then use a browser (Mozilla) to open the data file from http://www.giss.nasa.gov/data/update/gistemp/station_data/, and then call getgiss again with the station details. It is easier to access the NARP data.

Usage

```
getgiss(stnr=NULL,location=NULL,lon=NULL,lat=NULL,stations=NULL,
        silent=FALSE)
getnarp(stnr=NULL,location=NULL,lon=NULL,lat=NULL,stations=NULL,
        silent=TRUE,ele=101)
```

Arguments

stnr	Station number.
location	Name of location.
lon	longitude. If both lon and lat are specified and the other arguments are not, then find the closest station.
lat	latitude.
stations	a list object with details of GISS stations (Prescribing this saves time).
silent	For verbose use.
ele	Element code: see http://projects.met.no/~narp/data_index.html .

Value

a <- list of "monthly.station.record" class:

val	The monthly values (a 12-column matrix with one column for each year).
station	station number.
yy	The years of observation (vector).
lat,lon	Latitude and longitude of the location.
x.0E65N,y.0E65N	Distance in km from 0E, 65N.
location	Name of location .
wmo.no	WMO number.
start	Start of observations from this location.
yy0	First year in current record.
ele	Code of the element.
obs.name	Name of the element.
unit	Unit of the element.
country	The country in which the location is located.
quality	Code/description for data quality.
found	Flag: T - the data requested was found.
ref	Reference for the data set.

or a list of station details if the requested station was not found.

Author(s)

R.E. Benestad

Examples

```
## Not run:
#GISS:
obs.oxford <- getgiss(location="Oxford") # Takes longer time
stations <- getgiss()
obs.oxford <- getgiss(location="Oxford",stations=stations) #Quicker
plotStation(obs.oxford)
obs.broome <- getgiss(stnr="501942030004",stations=stations)
obs.120E.40S <- getgiss(lon=120,lat=-40,stations=stations)

#NARP:
> obs <- getnarp(4360)
#[1] "Retrieving the data from URL http://projects.met.no/~narp/narp"
#[1] "Please be patient"
#[1] "Found Tasilaq stnr= 4360 lon= -37.63 lat= 65.6 country= G"
> obs <- getnarp(lon=-37,lat=65)
#[1] "Retrieving the data from URL http://projects.met.no/~narp/narp"
#[1] "Please be patient"
#[1] "Find the nearest station to -37E and 65N."
#[1] "Found Tasilaq stnr= 4360 lon= -37.63 lat= 65.6 country= G"
> obs <- getnarp(lon=10,lat=60)
#[1] "Retrieving the data from URL http://projects.met.no/~narp/narp"
#[1] "Please be patient"
#[1] "Find the nearest station to 10E and 60N."
#[1] "Found Oslo stnr= 18700 lon= 10.72 lat= 59.95 country= N"

## End(Not run)
```

getnacd

Retrieve station record from the NACD set.

Description

Retrieve station record from the North Atlantic Climate Data (NACD) set: URL <http://www.dmi.dk/>. Also see [getdnmi](#) and [getnordklim](#).

Usage

```
getnacd(location="prompt",ele.c='101',ascii=FALSE,silent=FALSE,direc="data")
data(meta.nacd)
```

Arguments

location	name of climate station location or NACD station number.
ele.c	name of element [e.g. <code>avail.elem()</code> , or <code>'t2m'</code> , <code>'rr'</code> , <code>'slp'</code>].
ascii	Flag. T -> force ascii read, otherwise look for R-formatted version (faster).
silent	Flag. F -> print error messages.
direc	name of directory in which the data are stored.

Value

a <- list of "monthly.station.record" class:

val	The monthly values (a 12-column matrix with one column for each year).
station	station number.
yy	The years of observation (vector).
lat,lon	Latitude and longitude of the location.
x.0E65N,y.0E65N	Distance in km from 0E, 65N.
location	Name of location .
wmo.no	WMO number.
start	Start of observations from this location.
yy0	First year in current record.
ele	Code of the element.
obs.name	Name of the element.
unit	Unit of the element.
country	The country in which the location is located.
quality	Code/description for data quality.
found	Flag: T - the data requested was found.
ref	Reference for the data set.

Author(s)

R.E. Benestad

Examples

```
## Not run:
helsinki.rr <- getnacd("helsinki",ele=601)
obs.t2m <- getnacd()
data(meta.nacd,envir = env)

## End(Not run)
```

getnordklim

Retrieve station record from the Nordklima set or HadObs.

Description

getnordklim reads the data from Nordklim available at URL: http://www.smhi.se/hfa_coord/nordklim/. Also see [getdnmi](#) and [getnacd](#).

Usage

```
getnordklim(location=NULL,ele.c='101',ascii=FALSE,silent=FALSE,direc="data")
getHadObs(what="CET")
```

Arguments

location	name of climate station location. NULL returns the list of all names.
ele.c	name of element [e.g. avail.elem(), or 't2m', 'rr', 'slp'].
ascii	Flag. T -> force ascii read, otherwise look for R-formatted version (faster).
silent	Flag. F -> print error messages.
direc	name of directory in which the data are stored.
what	"CET": reads the central England Temperature; "EWP": reads Monthly England and Wales precipitation

Value

a <- list of "monthly.station.record" class:

val	The monthly values (a 12-column matrix with one column for each year).
station	station number.
yy	The years of observation (vector).
lat,lon	Latitude and longitude of the location.
x.0E65N,y.0E65N	Distance in km from 0E, 65N.
location	Name of location .
wmo.no	WMO number.
start	Start of observations from this location.
yy0	First year in current record.
ele	Code of the element.
obs.name	Name of the element.
unit	Unit of the element.
country	The country in which the location is located.
quality	Code/description for data quality.
found	Flag: T - the data requested was found.
ref	Reference for the data set.

Author(s)

R.E. Benestad

Examples

```
## Not run:
helsinki.rr <- getnordklim("helsinki",ele=601)

## End(Not run)
```

Description

Plots time series from field objects, eg from [retrieve.nc](#). Uses a bilinear interpolation scheme (`link{interp}`) from the `akima` package.

Usage

```
grd.box.ts(x,lon,lat,lev=NULL,what="abs",greenwich=TRUE,mon=NULL,
           col="grey10",lwd=1,lty=1,pch=".",add=FALSE,
           filter=NULL,type="l",main=NULL,sub=NULL,xlab=NULL,ylab=NULL,
           xlim=NULL,ylim=NULL)
```

Arguments

<code>x</code>	A field object.
<code>lon</code>	Longitude to plot.
<code>lat</code>	Latitude to plot.
<code>lev</code>	Vertical level to plot.
<code>what</code>	What to draw: "ano"-> anomalies, "cli"-> climatological values, "abs" -> absolute values.
<code>greenwich</code>	Maps centre on the Greenwich meridian.
<code>mon</code>	Month to extract
<code>col</code>	Colour.
<code>lwd</code>	Line width
<code>lty</code>	Line style.
<code>pch</code>	Plot character.
<code>add</code>	'TRUE' adds curve to old plot.
<code>filter</code>	If not NULL, this is a vector specifying window weighting argument of same name in <code>link{filter}</code> .
<code>type</code>	same as in <code>plot()</code> , only works for single time series.
<code>main</code>	Preset main title (see plot).
<code>sub</code>	subtitle
<code>xlab</code>	x label
<code>ylab</code>	y label
<code>xlim</code>	see plot
<code>ylim</code>	see plot

Value

A station object with interpolated values (see [station.obj](#) and [station.obj.dm](#)) that can be plotted using eg [plotStation](#).

Author(s)

R.E. Benestad

Examples

```
## Not run:
slp <- retrieve.nc("ncep_slp.nc")
grd.box.ts(slp,0,60,what="ano",mon=1)

## End(Not run)
```

helsinki.t2m	<i>Monthly mean temperature in Helsinki.</i>
--------------	--

Description

A station record of monthly mean temperature from Helsinki.

Usage

```
data(helsinki.t2m)
```

Format

a <- list of "monthly.station.record" class:

val	The monthly values (a 12-column matrix with one column for each year).
station	station number.
yy	The years of observation (vector).
lat,lon	Latitude and longitude of the location.
x.0E65N,y.0E65N	Distance in km from 0E, 65N.
location	Name of location .
wmo.no	WMO number.
start	Start of observations from this location.
yy0	First year in current record.
ele	Code of the element.
obs.name	Name of the element.
unit	Unit of the element.
country	The country in which the location is located.
quality	Code/description for data quality.
found	Flag: T - the data requested was found.
ref	Reference for the data set.

Source

The Nordklim data set: http://www.smhi.se/hfa_coord/nordklim/

References

Tuomenvirta et al. (2001), "Nordklim data set 1.0", DNMI KLIMA 08/01, pp. 26;
The Norwegian Meteorological Institute, P.O. Box 43, 0313 Oslo, Norway (<http://www.met.no>).

Examples

```
data(helsinki.t2m)
```

`instring`

instring

Description

Finds the position of a character in a string (character vector). Similar to `regexpr()`, but a test with `regexpr()` failed with some characters. `instring()` returns all position with a character match, whereas `regexpr()` only returns the first position for a pattern match.

New version: in addition to finding position of one character, it also finds the beginning position of a given test pattern.

Usage

```
instring(c, target, case.match=TRUE)
```

Arguments

<code>c</code>	Character to look for.
<code>target</code>	string to search.
<code>case.match</code>	FALSE -> not case sensitive.

Value

vector of integers.

Author(s)

R.E. Benestad

Examples

```
instring("e", "efile.dat")
# 1 5
regexpr("e", "efile.dat")
#[1] 1
#attr(,"match.length")
#[1] 1
# Case when regexpr() doesn't give the desired result:
regexpr(".", "file.name")
```

```
#[1] 1
#attr(,"match.length")
#[1] 1
instring(".", "file.name")
#[1] 5
instring("where", "guess where the word where is")
#
```

julday

Converts from month, day, and year to Julian days

Description

The function computes Julian days from month, day, and year. The code is based on the algorithm from Press et al. (1989), "Numerical Recipes in Pascal", Cambridge, p. 10. See also `chron` and `date` for similar functions. This function was included to avoid the dependency to the `chron` and `date` packages. See also [caldat](#).

Bug correction: 04.02.2005: got rid of 'Warning messages: 1: number of items to replace is not a multiple of replacement length'

Usage

```
julday(mm, id, iyyy)
```

Arguments

mm	month.
id	day.
iyyy	year.

Value

real

Author(s)

R.E. Benestad

Examples

```
julday(1,1,1)           # 1721424
julday(1,1,1970)       # 2440588
julday(9,4,2003)       # 2452887
julday(9,4,2003)-julday(1,1,1970) # 12299
julday(9,4,2003)-julday(1,1,2003) # 246
julday(1,1,2003)-julday(1,1,2002) # 365
julday(1,1,2001)-julday(1,1,2000) # 366
```

km2lat *Convert long-lat to km-km*

Description

The function computes the latitude from given the distance from a reference point. See also [km2lon](#) and [CO0E65N](#).

Usage

```
km2lat(x, y, x.centre=0, y.centre=65)
```

Arguments

x	not used.
y	distance from reference latitude in meridional direction.
x.centre	reference longitude.
y.centre	reference latitude.

Value

real

Author(s)

R.E. Benestad

Examples

```
library(clim.pact)
data(oslo.t2m)
print(c(oslo.t2m$lon,oslo.t2m$lat))
#[1] 10.71667 59.95000
xy<-CO0E65N(oslo.t2m$lon,oslo.t2m$lat)
oslo.t2m$lon<-xy$x
oslo.t2m$lat<-xy$y
print(c(oslo.t2m$lon,oslo.t2m$lat))
#[1] 595.4086 -560.3004
lon<-km2lon(oslo.t2m$lon,oslo.t2m$lat,x.centre=0,y.centre=65)
lat<-km2lat(oslo.t2m$lon,oslo.t2m$lat,x.centre=0,y.centre=65)
print(c(lon,lat))
#[1] 10.71667 59.95000
```

km2lon	<i>Convert long-lat to km-km</i>
--------	----------------------------------

Description

The function computes the longitude from given the distance from a reference point. See also [km2lat](#) and [CO0E65N](#).

Usage

```
km2lon(x, y, x.centre=0, y.centre=65)
```

Arguments

x	not used.
y	distance from reference latitude in meridional direction.
x.centre	reference longitude.
y.centre	reference latitude.

Value

real

Author(s)

R.E. Benestad

Examples

```
library(clim.pact)
data(oslo.t2m)
print(c(oslo.t2m$lon,oslo.t2m$lat))
#[1] 10.71667 59.95000
xy<-CO0E65N(oslo.t2m$lon,oslo.t2m$lat)
oslo.t2m$lon<-xy$x
oslo.t2m$lat<-xy$y
print(c(oslo.t2m$lon,oslo.t2m$lat))
#[1] 595.4086 -560.3004
lon<-km2lon(oslo.t2m$lon,oslo.t2m$lat,x.centre=0,y.centre=65)
lat<-km2lat(oslo.t2m$lon,oslo.t2m$lat,x.centre=0,y.centre=65)
print(c(lon,lat))
#[1] 10.71667 59.95000
```

 koebenhavn.t2m

Monthly mean temperature in Copenhagen.

Description

A station record of monthly mean temperature Copenhagen.

Usage

```
data(koebenhavn.t2m)
```

Format

list of "monthly.station.record" class:

val	The monthly values (a 12-column matrix with one column for each year).
station	station number.
yy	The years of observation (vector).
lat,lon	Latitude and longitude of the location.
x.0E65N,y.0E65N	Distance in km from 0E, 65N.
location	Name of location .
wmo.no	WMO number.
start	Start of observations from this location.
yy0	First year in current record.
ele	Code of the element.
obs.name	Name of the element.
unit	Unit of the element.
country	The country in which the location is located.
quality	Code/description for data quality.
found	Flag: T - the data requested was found.
ref	Reference for the data set.

Source

The Nordklim data set: http://www.smhi.se/hfa_coord/nordklim/

References

Tuomenvirta et al. (2001), "Nordklim data set 1.0", DNMI KLIMA 08/01, pp. 26; The Norwegian Meteorological Institute, P.O. Box 43, 0313 Oslo, Norway (<http://www.met.no>).

Examples

```
data(koebenhavn.t2m)
```

lagStation	<i>Introduce a lag in station values</i>
------------	--

Description

Shifts the station values in time: introduce a lag

Usage

```
lagStation(x, lag=0)
```

Arguments

x	A station object (see station.obj).
lag	lag

Value

A station object

Author(s)

R.E. Benestad

Examples

```
data(oslo.t2m)
oslo.t2m.1 <- lagStation(oslo.t2m,1)
plotStation(oslo.t2m,mon=1,what="t")
plotStation(oslo.t2m.1,add=TRUE,mon=1,what="t",col="darkblue")
```

lower.case	<i>convert to lower case</i>
------------	------------------------------

Description

Converts characters to lower case.

Usage

```
lower.case(u.case)
```

Arguments

u.case	Strings or arrays of strings.
--------	-------------------------------

Value

converted strings or arrays of strings.

Author(s)

R.E. Benestad

Examples

```
print(upper.case(c("qwerty e", "asdf rT"))) # "QWERTY" "ASDF"
print(lower.case(c("QWERTY", "ASDF"))) # "qwErty" "asdf"
print(strip(c("Hello there!", "Oslo", " ", "NA "))) # "Hello" "Oslo" " " "NA"
```

map

Produce a map

Description

Produces maps.

Usage

```
map(x, y=NULL, col="black", lwd=1, lty=1, sym=TRUE,
    plot=TRUE, inv.col=FALSE, add=FALSE, las = 1,
    levels=NULL, main=NULL, sub=NULL, xlim=NULL,
    ylim=NULL, newFig=TRUE)
```

Arguments

x	A map object.
y	A map object. If given, map.map plots the difference: x - y
col	Colour of contours.
lwd	Contour line width.
lty	Contour line style.
sym	Symmetry: if True, use zlimits c(-lmaxl,+lmaxl).
plot	TRUE gives graphics
inv.col	Inverse color scheme (e.g. 'TRUE' gives red for drier and blue for wetter conditions).
add	Add to plot: add contours only.
las	see par
levels	see contour
main	see plot
sub	see plot
xlim	see plot
ylim	see plot
newFig	True, call newFig for new figure.

Value

A map object

Author(s)

R.E. Benestad

Examples

```
data(DNMI.slp)
slpmap <- mapField(DNMI.slp,what="abs")
map(slpmap,sym=FALSE)
```

mapEOF

Map EOF

Description

Draws maps of the spatial structures described by the EOF ([EOF](#) patterns. Is similar to [plotEOF](#), but only plots the spatial information. Useful for comparing the spatial patterns in different EOFs.

Usage

```
mapEOF(x, i.eof=1, nlevs=9, add=FALSE,
       col=c("red", "blue", "darkgreen", "steelblue"), lwd=2, lty=1)
```

Arguments

x	An EOF object.
i.eof	The EOF to plot.
nlevs	Number of contour levels.
add	Add a map on pre-existing map - see contour.
col	Colour.
lwd	Line width.
lty	Line type.

Value

A map object (currently, only the last field in a mixed.field object).

Author(s)

R.E. Benestad

Examples

```
library(clim.pact)
data(eof.slp)
mapEOF(eof.slp)
mapEOF(eof.slp,i.eof=2,col="blue",add=TRUE)
```

mapField

*MapField***Description**

Draws maps of fields in a field object, eg read using [retrieve.nc](#).

Usage

```
mapField(x,l=NULL,greenwich=TRUE,plot=TRUE,
         what="ano",method="nice",val.rng=NULL,
         col="black",col.coast="grey",lwd=2,lty=1,
         add=FALSE,las = 1,levels=NULL,xlim=NULL,
         ylim=NULL,newFig=TRUE)
```

Arguments

x	A field object.
l	The field to map. Default: the last field in the record.
what	What to draw: "ano"-> anomalies, "cli"-> climatological values, "abs" -> absolute values.
method	"nice" -> filled.contour, otherwise use image.
val.rng	Valid range: interval used for colour scale.
greenwich	'TRUE' centres on Greenwich meridian.
plot	'TRUE' produces graphic.
col	Contour line colour for levels.
col.coast	Contour line colour for coast lines.
lwd	Line width.
lty	Line type.
add	Adds map to old figure.
las	See par
levels	See contour
xlim	see plot
ylim	see plot
newFig	TRUE call newFig for new figure.

Value

A [map](#) object.

Author(s)

R.E. Benestad

Examples

```
library(clim.pact)
## Not run:
skt<-retrieve.nc("skt.mon.mean.nc",
                 x.rng=c(-90,50),y.rng=c(0,75))
bitmap("ncep.skt.jpg",type="jpeg")
mapField(skt)
dev.off()

## End(Not run)
```

meanField

Mean field

Description

Computes a map showing the mean field values.

Usage

```
meanField(x,lon.rng=NULL,lat.rng=NULL,t.rng=NULL,mon=NULL)
```

Arguments

x	A field object.
lon.rng	Extract the longitude interval.
lat.rng	Extract the latitude interval.
t.rng	Extract the time interval. Can be numerical (year) or character (date, e.g. "01-Jan-1998")
mon	Month (1-12) or season (1-12) to extract if !is.null.

Value

A map object

Author(s)

R.E. Benestad

Examples

```
## Not run:
slp <- retrieve.nc("ncep_slp.nc", x.rng=c(5,12), y.rng=c(58,63))
mslp <- meanField(slp)

## End(Not run)
```

mergeEOF

*Merge EOFs***Description**

Merges two EOF objects by making the first one resembling the second (a bit similar to [mergeStation](#)). Can be used for reconstructing fields.

Usage

```
mergeEOF(eof1, eof2, plot=TRUE, silent=FALSE, method="lm",
         match="time", cut.off=8, adjust=TRUE)
```

Arguments

eof1	The EOF object (see EOF) for the earliest data (assuming this data set contains most errors).
eof2	The EOF object (see EOF) for the most recent data (assuming this data set contains less errors)
plot	if 'TRUE' then plot diagnostics
silent	if 'TRUE' then do not print out diagnostics.
method	"lm" or "matrix projection" (G. Strang (1988), "Linear algebra and its applications", Hartcourt Brace & Company, 3rd ed.(p.147))
match	"time" or "space"
cut.off	How many EOFs to include
adjust	TRUE: apply adjustEOF to results.

Value

An 'eof' object.

Author(s)

R.E. Benestad

Examples

```
## Not run:
data(DNMI.slp)
NCEP.slp<-retrieve.nc("~/data/ncep/slp.mon.mean.nc",
                      x.rng=c(-90,50),y.rng=c(0,75))

#-----
# Need to fix some details of the NCEP.slp object (only for
# proper 'housekeeping')

NCEP.slp$dd[] <- 15
attr(NCEP.slp$tim,unit') <- "month"
class(NCEP.slp) <- c("field","monthly.field.object")
#-----

eof1 <- EOF(DNMI.slp,mon=1)
eof2 <- EOF(NCEP.slp,mon=1)
eof <- mergeEOF(eof1,eof2)

## End(Not run)
```

mergeStation

Merge climate station series.

Description

Merges two series from different sources, eg from NACD ([getnacd](#)) and DNMI ([getdnmi](#)). The code is useful for updating long climate series with new observations from a different database. The routine cokpares data for overlapping times and prints out diagnostics about the two data sets.

Usage

```
mergeStation(x.1,x.2,plot=FALSE,print=TRUE,rescale=TRUE)
```

Arguments

x.1	1st series.
x.2	2nd series.
plot	'TRUE' plots the overlap.
print	'TRUE' prints diagnostics.
rescale	'TRUE' uses lm to determine amplitude of second sequence otherwise the amplitude of second sequence is not changed.

Value

A climate station series object.

Author(s)

R.E. Benestad

Examples

```
## Not run:
oslo.1 <- getnacd("OSLO-BLINDERN")
oslo.2 <- getdnmi("oslo")
print(range(oslo.1$yy))
#[1] 1890 1990
print(range(oslo.2$yy))
#[1] 1937 2002
oslo <- mergeStation(oslo.1,oslo.2)
#[1] "Time intervals:"
#[1] 1890 1990
#[1] 1937 2002
#[1] 1937.042 1990.958
#[1] "RMSE: 0.04"
#
#Call:
#lm(formula = y ~ 1 + x, data = ovrlp)
#
#Residuals:
#   Min       1Q   Median       3Q      Max
#-7.24005 -0.03271  0.01161  0.06006  7.61593
#Coefficients:
#              Estimate Std. Error t value Pr(>|t|)
#(Intercept) 0.029044   0.047482   0.612   0.541
#x            0.993886   0.004866  204.231 <2e-16 ***
#---
#Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
#
#Residual standard error: 0.9738 on 644 degrees of freedom
#Multiple R-Squared: 0.9848, Adjusted R-squared: 0.9848
#F-statistic: 4.171e+04 on 1 and 644 DF,  p-value: < 2.2e-16

print(range(oslo$yy))
#[1] 1890 2002

## End(Not run)
```

mixFields

*mixFields***Description**

Mix fields by combining two different gridded sets of observation. Observations/data for representing values at n different locations at a given time (t) can be described in terms of a vector

$$\vec{x}(t) = [x_1, x_2, \dots, x_n].$$

Two different sets of observations can be represented by two vectors y and z of lengths n and m respectively. In `mix.fields`, the information in these two data sets are combined combining the two vectors:

$$\vec{x}(t) = [\vec{y}(t), \vec{z}(t)] = [y_1, y_2, \dots, y_n, z_1, z_2, \dots, z_m].$$

The length of the final vector of the mixed field is the sum of the lengths of the two respective vectors. The two data sets do not have to be on the same grid.

reference: Bretherton et al. (1992) "An Intercomparison of Methods for finding Coupled Patterns in Climate Data", *J. Climate*, **vol 5**, 541-560.

The output from `mixFields` can be used in [EOF](#) to compute mixed-common EOFs which subsequently can be used as predictors in [DS](#) in order to downscale climate scenarios (Benestad et al. (2002), "Empirically downscaled temperature scenarios for Svalbard", *Atm. Sci. Lett.*, doi:10.1006/asle.2002.0050).

The functions `combineFields` and `combineEOFs` bundle together fields of different parameters, and allows downscaling of multiple variables, but without performing a mixed-EOF analysis. Thus, allows ESD with a multiple regression analysis involving both e.g. SLP and T(2m) as input variables as in traditional multiple regression:

$$\hat{y}(t) = \alpha_0 + \alpha_1 x(t) + \alpha_2 z(t) + \dots$$

Usage

```
mixFields(field.1, field.2, mon=NULL, interval=NULL)
combineFields(field.1, field.2)
combineEOFs(eof.1, eof.2)
```

Arguments

<code>field.1</code>	A 'field.object'.
<code>field.2</code>	A 'field.object'.
<code>eof.1</code>	An 'EOF.object'.
<code>eof.2</code>	An 'EOF.object'.
<code>mon</code>	Calendar month to extract.
<code>interval</code>	Time interval to extract.

Value

A 'field.object'.

Author(s)

R.E. Benestad

Examples

```
## Not run:
library(clim.pact)
x.1 <- retrieve.nc("/home/kareb/data/ncep/ncep_t2m.nc",
                  x.rng=c(-60,40),y.rng=c(50,75))
x.2 <- retrieve.nc("/home/kareb/data/ncep/ncep_slp.nc",
                  x.rng=c(-60,40),y.rng=c(50,75))
print(x.1$v.name)

print("Read GCM predictor data.")
X.1 <- retrieve.nc("data/mpi-gsdiot2m.nc",
                  x.rng=c(-60,40),y.rng=c(50,75))
X.2 <- retrieve.nc("data/mpi-gsdiot_slp.nc",
                  x.rng=c(-60,40),y.rng=c(50,75))
print(X.1$v.name)
print("Cat fields.")
xX.1 <- catFields(x.1,X.1,interval.1=c(1958,1998),interval.2=c(1958,2050))
xX.2 <- catFields(x.2,X.2,interval.1=c(1958,1998),interval.2=c(1958,2050))
xX <- mixFields(xX.1,xX.2,mon=1,
                interval=c(1900,2050))
print("EOF")
eof.c <- eof(xX.1,mon=1)
eof.mc <- eof(xX,mon=1)

## End(Not run)
```

mod

Modulus of a division.

Description

Returns the modulus of a division: returns the remainder of the expression x/y .

Usage

```
mod(x,y)
```

Arguments

x	nominator (integer).
y	denominator (integer).

Value

integer value from 0 .. (y-1)

Author(s)

R.E. Benestad

Examples

```
mod(101,10) # 1
mod(4,12)   # 4
mod(123,12) # 3
```

MVR

*Multivariate regression analysis***Description**

Applies a multivariate regression (MVR) analysis to two data sets. The MVR here is based on the projection to obtain a least squares approximation and uses the formula of Strang (1988) "Linear Algebra and its applications", Harcourt Brace and Company, p. 156. The method is also documented in Benestad (1999) "MVR applied to Statistical Downscaling for prediction of Monthly Mean Land Surface Temperatures: Model Documentation", DNMI KLIMA Report 02/99 at http://met.no/english/r_and_d_activities/publications/1999.html.

For the expression

$$Ax = b$$

, then the projection of b onto the columns space of A through

$$p = A(A^T A)^{-1} A^T b$$

.

Here x is called the predictor, b is the prediction of the equation, and the data - called the predictand - used for calibration is y .

The predictor and predictand arguments can be both EOF-objects or field objects. The former requires much less memory.

Also see [CCA](#), [EOF](#) and [EOF2field](#).

Usage

```
MVR(x,y,plot=TRUE,main="Multivariate regression",sub="",test=FALSE,
    i.eofs=1:8,LINPACK=TRUE, SVD=TRUE)
```

Arguments

<code>x</code>	A field or an eof object. The predictor
<code>y</code>	A field or an eof object. The predictand
<code>plot</code>	Flag: plot the diagnostics.
<code>test</code>	Flag: test by reconstructing one series (leading EOF or a grid-box series).
<code>i.eofs</code>	Which EOFs to include (only when the input is given as eof objects).

LINPACK	'TRUE': svd; 'FALSE':La.svd
main	main title (see <code>link{plot}</code>).
sub	subtitle (see <code>link{plot}</code>).
SVD	Flag: determine which approach to use: SVD or eigenfunction-based algorithm.

Value

A MVR object that is similar to a field or EOF object (inherits the object type, with an additional "MVR" label) with the projection (dat), but with additional fields such as the weights (psi, which is a map object) [dat represents p and psi represents \hat{x} in Strang (1988)].

Author(s)

R.E. Benestad

Examples

```
## Not run:
data(DNMI.t2m)
data(DNMI.slp)

# MVR with fields
DNMI.t2m.jan <- catFields(DNMI.t2m,mon=1)
DNMI.slp.jan <- catFields(DNMI.slp,mon=1)
mvr.field <- MVR(DNMI.t2m.jan,DNMI.slp.jan)

# MVR with EOFs
eof.1 <- EOF(DNMI.t2m,mon=1)
eof.2 <- EOF(DNMI.slp,mon=1)
mvr.eof <- MVR(eof.1,eof.2)

DNMI.t2m.jan <- catFields(DNMI.t2m,mon=1)
DNMI.slp.jan <- catFields(DNMI.slp,mon=1)
mvr.field.1 <- MVR(DNMI.t2m.jan,DNMI.slp.jan,LINPACK=FALSE)
mvr.field.2 <- catFields(MVR(DNMI.t2m,DNMI.slp),mon=1)
mvr.field.3 <- MVR(DNMI.t2m.jan,DNMI.slp.jan)
EOF(mvr.field.3)-> eof3
EOF(mvr.field.2)-> eof2
EOF(mvr.field.1)-> eof1
plotEOF(eof1)
plotEOF(eof2)
plotEOF(eof3)

## End(Not run)
```

narp

Station data from Nordic Arctic Research Program

Description

Nordic Arctic Research Program (NARP) Long-term variations in atmospheric circulation and climate in the Arctic. These data have been freely available from a server at met.no, but due to re-organisation, they are no longer on the web. Thanks to Ole Einar Tveito, I got a copy of the data, which are now included in clim.pact. These data are accessed through [getnarp](#).

Usage

```
data(narp)
  data(narp.meta)
  data(nacd.meta)
  data(nordklim.meta)
```

Format

narp is a matrix with 16 columns and narp.meta is a list object. The 16th column in narp indicates the type of element: code '101', Mean monthly air-temperature; '111', Mean maximum monthly air-temperature; '112', Absolute maximum monthly air-temperature; '121', Mean minimum monthly air-temperature; '122', Absolute minimum air-temperature; '401', Mean monthly sea level pressure; '601', Mean monthly precipitation sum; '602', Highest monthly 1-day precipitation; '701', Mean monthly days with snow cover > 50 percent; '801', Mean monthly cloud cover.

Source

Eirik Forland and Ole Einar Tveito, The Norwegian Meteorological Institute; Trausti Jonsson, Icelandic Met Office; Claus Kern-Hansen, Data & Climate Division, Danish Meteorological Institute

References

Ellen Vaarby Laursen (2003) 'DMI monthly climate data, 1873-2002, contribution to Nordic Arctic Research Programme (NARP)', DANISH METEOROLOGICAL INSTITUTE, TECHNICAL REPORT 03-25 (<http://www.dmi.dk/dmi/tr03-25.pdf>); http://thule.oulu.fi/narp/Projects/a_natural/Forland.htm

Examples

```
data(narp.meta)
data(nacd.meta)
```

newFig *Create a new Window*

Description

Creates a new window by calling the appropriate device, eg `x11()`, `windows()`, `pdf()`, `postscript()`, etc.

Usage

```
newFig()
```

Author(s)

R.E. Benestad

Examples

```
newFig()
```

num2str *Convert numbers to string and format*

Description

Convert numbers to string and format. Similar to FORTRAN 'F8.2' format statement.

Usage

```
num2str(x,dec=2,f.width=NULL,d.point=".")
```

Arguments

x	Real numbers.
dec	number of decimal points.
f.width	width of field.
d.point	character marking the decimal point.

Value

string.

Author(s)

R.E. Benestad

Examples

```
print(num2str(c(1,23.4282,-3.14),dec=3))
#[1] "1.000" "23.428" "-3.140"
```

objDS

Objective downscaling of monthly means

Description

'Objective' downscaling based on [DS](#). The function selects region according to a correlation analysis, setting the borders where the correlation with the station series become zero. A fit to a truncated Fourier expansion is used to describe profiles of correlation coefficients in zonal and meridional directions from where the station is located.

Version 2.1-5: smallest spatial domain for objDS is set to +/- 10deg N and E of the station location in order to avoid problems with dimensions if one only contains one data point (the domain becomes a line instead of a rectangle).

RMSE estimates the root-mean-square-error between two field objects.

Usage

```
objDS(field.obs,field.gcm,station,plot=TRUE,positive=NULL,
      mon=NULL,direc="dsgraphicsoutput",cal.id=NULL,
      ldetrnd=TRUE,i.eofs=seq(1,8,by=1),ex.tag="",
      method="1m",leps=FALSE,param="t2m",failure.action=NULL,
      plot.res=FALSE,plot.rate=FALSE,xtr.args="",opt.dom=TRUE,
      swsm="step",predm="predict",lsave=FALSE,rmac=TRUE,
      silent=FALSE,qualitycontrol=TRUE,LINPACK=TRUE,wOBS=0.25,
      fastregrid=FALSE,neofs=20)

plotDSobj(result,outdir="dsgraphicsoutput",figs=c(1,2,3,4),main="")
RMSE(field.obs,field.gcm)
```

Arguments

field.obs	The gridded observation predictor retrieve.nc .
field.gcm	The climate simulation predictor retrieve.nc .
station	A climate.station object (station.obj or station.obj.dm). [e.g. from <code>getnacd</code> , <code>getnordklim</code> or <code>station.obj</code>].
plot	'TRUE' produces figures.
positive	'TRUE': only consider the region where correlations are positive (important for temperature predictors).
mon	month or season to downscale, loops though the 12 calendar months if NULL.
direc	name of directory in which the output is dumped (e.g. figures, tables).

cal.id	ID tag used for calibration. By default use the first field (<code>catFields</code>) for calibration.
ldetrnd	F for no detrending; T for removing linear trends before model calibration.
i.eofs	select which EOFs to include in the setp-wise screening.
ex.tag	Extra labelling tag for file names for experiments.
method	Sets the method to use for regression. Method is set to "lm" by default, but "anm" allows the incorporation of an analog model, see anm . "anm.weight" weights the principal components according to the eigenvalues, whereas "anm" uses unweighted series.
leps	'TRUE' produces EPS figures (files).
param	Name of parameter (for plot labels).
plot.res	'TRUE' shows statistics for residuals.
plot.rate	'TRUE' shows analysis of rate-of-change.
xtr.args	Extra/additional arguments in the formula.
swsm	Step-wise screening method, default=='step'; 'none' skips stepwise sceening.
predm	Prediction method, default is "predict"
lsave	TRUE -> saves the result on disc.
rmac	TRUE -> subtracts (removes) the annual cycle in station data.
silent	TRUE -> no output to screen.
qualitycontrol	TRUE: perform a quality control consisting of comparing the smoothness of the monthly trend estimates throughout the year and repeat the downscaling with a smaller domain if adjacent trend estimates vary significantly (diff greater than 3*variance of rates).
LINPACK	'TRUE': svd; 'FALSE':La.svd
wOBS	Used for weighting down GCM results in the common EOF analysis. The GCM results are re-scaled after the analysis.wOBS=NULL skips this scaling/re-scaling.
opt.dom	FALSE - do not search for 'optimum' domain.
failure.action	If stepwise screening in DS fails to select any variables, then call the function specified by a character string, unless set to NULL.
result	A 'objDS' object from objDS .
outdir	Directory for storing EPS-files with figures
figs	Figs to plot: 1 - time series, 2 - time series of residuals, 3 - distribution of residuals, 4 - rates of change.
main	title of plot
fastregrid	Option in <code>catFields</code> - use fastRegrid
neofs	Number of EOF-modes to retain, determining the degree of details to be retained.

Value

An 'objDS' object - a list of objects:

```

station  a 'station' object (see getnacd)
  Jan    a 'ds' (see DS) object
  Feb    a 'ds' object
  Mar    a 'ds' object
  Apr    a 'ds' object
  May    a 'ds' object
  Jun    a 'ds' object
  Jul    a 'ds' object
  Aug    a 'ds' object
  Sep    a 'ds' object
  Oct    a 'ds' object
  Nov    a 'ds' object
  Dec    a 'ds' object

```

Author(s)

R.E. Benestad

Examples

```

## Not run:
library(clim.pact)
oslo<-getnordklim("Oslo-Blindern")
t2m.obs <- retrieve.nc("air.mon.mean.nc.nc")
           # Get gridded observations/analysis from NCEP
t2m.gcm <- retrieve.nc("pcmdi.ipcc4.mpi_echam5.20c3m.run1.daily.tas_A2_1961-1980.nc",
                      v.nam="tas",x.rng=obs$lon+c(-40,40),y.rng=obs$lat+c(-20,15)
                      # Get results from climate models
attr(field.gcm$tim,"unit")
attr(field.gcm$tim,"unit") <- "month"
           # Fix - sometimes retrieve.nc doesn't manage to attribute the
           # correct time units
class(field.gcm)
class(field.gcm) <- c("field","monthly.field.object") # Fix.

ds1 <- objDS(field.obs=t2m.obs,field.gcm=t2m.gcm,station=oslo)

data(DNMI.t2m)
ds2 <- objDS(field.obs=DNMI.t2m,field.gcm=t2m.gcm,station=oslo)
plotDSobj(ds2)

## End(Not run)

```

Description

A station record of daily mean temperature and daily precipitation from Oslo-Blindern.

Usage

```
data(oslo.dm)
```

Format

a "daily.station.record"-class object.

t2m	a vector holding daily mean temperature.
precip	a vector holding daily precipitation.
dd	a vector holding day of month.
mm	a vector holding the month.
yy	a vector holding the year.
obs.name	the name of observation: eg c("Daily mean temperature", "Daily precipitation").
unit	the unite of observation: eg c("deg C", "mm/day").
ele	element code: eg c("tam", "rr").
station	local (national) station number.
lat	latitude.
lon	longitude.
alt	altitude.
location	name of location.
wmo.no	WMO number of station.
start	start of measurements.
yy0	first year of record.
country	name of country.
ref	reference to the data.

Source

The Norwegian Meteorological Institute, Climatology deivision.

References

The Norwegian Meteorological Institute, P.O. Box 43, 0313 Oslo, Norway (<http://www.met.no>).

Examples

```
data(oslo.dm)
plot(oslo.dm$precip)
```

oslo.t2m

*Monthly mean temperature in Oslo.***Description**

A station record of monthly mean temperature from Oslo-Blindern.

Usage

```
data(oslo.t2m)
```

Format

list of "monthly.station.record" class:

val	The monthly values (a 12-column matrix with one column for each year).
station	station number.
yy	The years of observation (vector).
lat,lon	Latitude and longitude of the location.
x.0E65N,y.0E65N	Distance in km from 0E, 65N.
location	Name of location .
wmo.no	WMO number.
start	Start of observations from this location.
yy0	First year in current record.
ele	Code of the element.
obs.name	Name of the element.
unit	Unit of the element.
country	The country in which the location is located.
quality	Code/description for data quality.
found	Flag: T - the data requested was found.
ref	Reference for the data set.

Source

The Nordklim data set: http://www.smhi.se/hfa_coord/nordklim/

References

Tuomenvirta et al. (2001), "Nordklim data set 1.0", DNMI KLIMA 08/01, pp. 26;
The Norwegian Meteorological Institute, P.O. Box 43, 0313 Oslo, Norway (<http://www.met.no>).

Examples

```
data(oslo.t2m)
data(DNMI.t2m)
corField(DNMI.t2m,oslo.t2m,mon=7)
```

 patternIndex

Create a Index for a spatial pattern

Description

Create a Index for a spatial pattern identified from a map object, eg link{corField}, link{mapField}, or link{composite.field}. The index is computed using a spatial correlation.

Usage

```
patternIndex(map, field, anomaly=TRUE)
```

Arguments

map	Map describing spatial structure.
field	Field in which the pattern is searched.
anomaly	If true, compute and use the anomalies of the given field.

Value

list containing the fields:

index	index
yy	years
mm	months
dd	days

Author(s)

R.E. Benestad

Examples

```
## Not run:
sst <- retrieve.nc("DNMI_sst.nc")
data(oslo.t2m)
csst <- composite.field(sst,oslo.t2m)
patternIndex(csst,sst,anomaly=FALSE)

lsst<-mapField(sst)
patternIndex(lsst,sst)

## End(Not run)
```

plotDS *Plot downscaled results*

Description

Plots the results from [DS](#).

Usage

```
plotDS(ds.obj,leps=FALSE,plot.ts=TRUE,plot.map=TRUE,plot.res=FALSE,
plot.rate=FALSE,add=FALSE,col="darkred",lwd=2,lty=1,
      direc="output",main=NULL,sub=NULL,xlab=NULL,ylab=NULL,
newplot=TRUE)
```

Arguments

ds.obj	A DS object.
leps	'TRUE' produces EPS files.
plot.ts	'TRUE' shows the time series.
plot.map	'TRUE' shows the spatial predictor pattern.
plot.res	'TRUE' shows statistics for residuals.
plot.rate	'TRUE' shows analysis of rate-of-change.
add	TRUE: adds a scenario to old time series plot.
col	Colour of scenario time series.
lwd	Line width of scenario time series.
lty	Line style of scenario time series.
direc	Directory for graphical output.
main	Preset main title (see plot).
sub	subtitle
xlab	x label
ylab	y label
newplot	TRUE: opens new windows

Value

plotDS returns a map object for the large-scale synoptic (anomaly) pattern, which can be plotted on a sphere ([satellite](#)) or coloured map ([map](#)).

Author(s)

R.E. Benestad

Examples

```
data(helsinki.t2m)
data(eof.mc)
ds.helsinki<-DS(dat=helsinki.t2m,preds=eof.mc,plot=FALSE)
plotDS(ds.helsinki,leps=TRUE)
```

plotEOF

Plot EOFs

Description

Plots the results from an (mixed-common) EOF analysis ([EOF](#)). This function produces 3 plots: i) EOF map, ii) variances, and iii) PC time series. Hard copies are also produced in EPS format under the current working directory: "plotEOF_1.eps", "plotEOF_2.eps" and "plotEOF_3.eps" respectively.

Usage

```
plotEOF(x,i.eof=1,nlevs=5,
        col=c("red","blue","darkgreen","steelblue"),
        main=NULL,sub=NULL,plot=TRUE)
```

Arguments

x	An EOF object.
i.eof	EOF to plot.
nlevs	Contour levels.
col	Colour.
main	Preset main title (see plot).
sub	subtitle
plot	if false, only return a map object of chosen EOF.

Author(s)

R.E. Benestad

Examples

```
## Not run:
data(eof.mc)
plotEOF(eof.mc)
x11()
data(eof.dmc)
plotEOF(eof.dmc)

## End(Not run)
```

plotField

*plotField***Description**

Produce 2D plots like maps and Hovmuller diagrams. A poor man's version of Ferret's <http://ferret.wrc.noaa.gov/Ferret/> plot function. `plot.field` is a high level command that utilises `mapField` or `grd.box.ts` whenever appropriate.

Usage

```
plotField(x,lon=NULL,lat=NULL,tim=NULL,mon=NULL,val.rng=NULL,
          col="black",col.coast="grey",lty=1,lwd=1,what="ano",
          type="l",pch=".",my.col=NULL,add=FALSE,
          main=NULL,sub=NULL,xlab=NULL,ylab=NULL,
          xlim=NULL,ylim=NULL)
```

Arguments

<code>x</code>	A field object.
<code>lon</code>	Position for longitude. One of <code>lon</code> , <code>lat</code> , <code>tim</code> must be set.
<code>lat</code>	Position for latitude.
<code>tim</code>	Position for time as time index (1:length(tim)).
<code>mon</code>	Month to extract.
<code>val.rng</code>	Valid range: interval used for colour scale.
<code>col</code>	Contour line colour for levels.
<code>col.coast</code>	Contour line colour for coast lines.
<code>lty</code>	Contour line type.
<code>lwd</code>	Contour line width.
<code>what</code>	Choose between "ano" - anomaly; "abs" absolute; "cli" climatological.
<code>type</code>	same as in <code>plot()</code> , only works for single time series.
<code>pch</code>	same as in <code>plot()</code> , only works for single time series.
<code>my.col</code>	colour palette (see <code>link{rgb}</code>).
<code>add</code>	TRUE adds a time series to previous (see <code>link{grd.box.ts}</code>).
<code>main</code>	Preset main title (see <code>plot</code>).
<code>sub</code>	subtitle
<code>xlab</code>	x label
<code>ylab</code>	y label
<code>xlim</code>	see <code>plot</code>
<code>ylim</code>	see <code>plot</code>

Author(s)

R.E. Benestad

Examples

```
## Not run:
skt <- retrieve.nc("skt.mon.mean.nc", x.rng=c(-90,50), y.rng=c(0,75))

# Maps of monthly mean skin temperatures:
plotField(skt, tim=1, val.rng=c(-20,20))
dev2bitmap("ncep.skt_194801.jpg", type="jpeg")

plotField(skt, tim=100, col="blue", col.coast="darkgreen", val.rng=c(-10,10))

# For adding extra points/contours:

# From filled.contour in base
mar.orig <- (par.orig <- par(c("mar", "las", "mfrow")))$margin
on.exit(par(par.orig))

w <- (3 + mar.orig[2]) * par('csi') * 2.54
layout(matrix(c(2, 1), nc=2), widths=c(1, lcm(w)))

par(las = 1)
mar <- mar.orig
mar[4] <- 1
par(mar=mar)
# End of section affecting the window set up.

points(0,50,pch=21,col="red")
grid()
dev2bitmap("ncep.skt_195604.jpg", type="jpeg")

# A hovmuller diagram:
plotField(skt, lon=0, val.rng=c(-10,10))
dev2bitmap("ncep.skt_lontim.jpg", type="jpeg")

# A single time series:
plotField(skt, lon=-20, lat=50)

## End(Not run)
```

Description

Plots data in the monthly station records. The data may be read through [getnacd](#), [getnordklim](#), [getdnmi](#), or created using [station.obj](#). The commands [avail.elem](#), [avail.locs](#) can be used to identify the station records available (in a given subdirectory).

Usage

```
plotStation(obs,l.anom=TRUE,mon=NULL,leps=FALSE,
            out.dir="output",what="b",trend=TRUE,std.lev=TRUE,
            type="b",pch=21,col="grey30",lwd=1,lty=1,add=FALSE,
            main=NULL,sub=NULL,xlab=NULL,ylab=NULL,
            normal.period=NULL,method="rowMeans",
            ylim=NULL,xlim=NULL)
```

Arguments

obs	A climate station series.
l.anom	flag: T -> plot anomalies.
mon	select month to plot, A value of 0 plots all months. mon=c(12,1,2) plots the DJF mean.
leps	Flag: T -> produce EPS files (hard copy).
out.dir	Directory where to store hard copies.
what	"t"=timeseries; "d"=distribution; "b"=both; "n"=no graphics but return results.
trend	show best-fit polynomial (5th order) trend
std.lev	show +/- sd levels.
type	same as in plot .
pch	same as in plot .
col	colour, same as in plot .
lwd	line width, same as in plot .
lty	line type, same as in plot .
add	TRUE adds a time series to previous (see link{grd.box.ts}).
main	Preset main title (see plot).
sub	subtitle
xlab	x label
ylab	y label
xlim	see link{plot}
ylim	see link{plot}
normal.period	Normal period. NULL uses the entire data. Set to c(1961,1990) for the last normal period.
method	what to plot (e.g. rowMeans or rowSums)

Value

a <- list of "monthly.station.record" class:

val	The monthly values (a 12-column matrix with one column for each year).
station	station number.
yy	The years of observation (vector).
lat,lon	Latitude and longitude of the location.
x.0E65N,y.0E65N	Distance in km from 0E, 65N.
location	Name of location .
wmo.no	WMO number.
start	Start of observations from this location.
yy0	First year in current record.
ele	Code of the element.
obs.name	Name of the element.
unit	Unit of the element.
country	The country in which the location is located.
quality	Code/description for data quality.
found	Flag: T - the data requested was found.
ref	Reference for the data set.

Author(s)

R.E. Benestad

Examples

```
## Not run:
data(tromsoe.t2m)
plotStation(tromsoe.t2m)

## End(Not run)
```

plumePlot

Plot downscaled time series as plumes

Description

Retrieves an ensemble of time series from [DS](#) and plots these as a plume. The routine retrieves the ds objects from disc.

Usage

```
plumePlot(ds.name.list=NULL,location,mon,direc="output",
          t.rng=c(1850,2074),r2.th=50,p.th=0.05,
          col="darkred",lwd=2,lty=1)
```

Arguments

<code>ds.name.list</code>	A list of file names holding the ds objects.
<code>location</code>	Name of location to plot.
<code>mon</code>	Month to plot.
<code>direc</code>	The subdirectory in which the ds objects are stored.
<code>t.rng</code>	Time interval to plot.
<code>r2.th</code>	R-squared threshold: only use scenarios that account for equal to or more than <code>r2.th</code> of the variance in %.
<code>p.th</code>	p-value threshold: only use scenarios that have p-values equal to or less than <code>p.th</code> .
<code>col</code>	Colour of scenario time series.
<code>lwd</code>	Line width of scenario time series.
<code>lty</code>	Line style of scenario time series.

Author(s)

R.E. Benestad

Examples

```
## Not run:  
ds.list<-avail.ds()  
plumePlot(ds.list,location="OSLO-BLINDERN",mon=1)  
  
## End(Not run)
```

Description

After von Storch & Zwiers (1999), Statistical Analysis in Climate Research, p. 338.

Usage

```
POP(x,plot=TRUE,main="POP analysis",sub="",test=FALSE,  
i.eofs=1:8,LINPACK=TRUE, mode = 1)  
plotPOP(pop,mode = 1, main = "POP analysis", sub = "")
```

Arguments

x	A field or an eof object.
plot	Flag: plot the diagnostics.
test	Flag: test by reconstructing one series (leading EOF or a grid-box series).
i.eofs	Which EOFs to include (only when the input is given as eof objects).
LINPACK	'TRUE': svd; 'FALSE':La.svd
main	main title (see <code>link{plot}</code>).
sub	subtitle (see <code>link{plot}</code>).
mode	POP mode to display
pop	a POP object.

Value

A POP object: an `link{eigen}` object with additional variables. `maps` (map of the absolute of POP patterns), `pop.Im` (the imaginary POP pattern), `pop.Re`(the real POP pattern), `lon` (longitude coordinates), and `lat` (latitude coordinates).

Author(s)

R.E. Benestad

Examples

```
## Not run:
data(DNMI.t2m)
eof.1 <- EOF(DNMI.t2m,mon=1)
pop <- POP(eof.1)

## End(Not run)
```

r2cdf

Save as netCDF file.

Description

Saves a field object, map or eof object as a netCDF file by creating a CDF file (ASCII) and then calling 'ncgen' through `system()`.

Usage

```
r2cdf(filename,x,missing=-999.99,cleanup=TRUE,
       ofs=NULL,scal=NULL,precision="short")
```

Arguments

filename	Name of file.
x	a Field, map, or eof object
missing	Missing value
cleanup	TRUE removes the CDF-file after the call.
ofs	offset: 'NULL' -> automatic setting
scal	scaling factor: 'NULL' -> automatic setting
precision	Precision for "byte", "char", "short", "long", "float", and "double". "int" may be used as a synonym for "long" and "real" may be used as a synonym for "float" in the CDL notation

Author(s)

R.E. Benestad

Examples

```
## Not run:
# Save EOFs as netCDF (use ncview or Ferret to view)
data(eof.slp)
r2cdf("test.nc",eof.slp)
#
#
slp <- retrieve.nc("data/DNMI_slp.nc")
mslp <- meanField(slp)
r2cdf("test.nc",mslp)
r2cdf("test.nc",slp)
#
slp <- cdfextract("data/nmc_slp.nc","slp",x.rng=c(-80,40),y.rng=c(20,75),
                 t.rng=c(times[is],times[is]+499),plot=FALSE)
r2cdf("test.nc",slp)
#
data(oslo.t2m)
map <- composite.field(slp,oslo.t2m)
r2cdf("test.nc",map)
#
Xdum=list(dat=slp$dat[1:10,,],lon=slp$lon,lat=slp$lat,tim=slp$tim[1:10],
         lev=NULL,v.name=slp$v.name,attributes=slp$attributes)
class(Xdum)="field"
r2cdf("test.nc",Xdum)
#
# The definition of a 'field' object is:
ny<-length(slp$lat); nx<-length(slp$lon)
slp <- list(dat=slp$dat,lon=slp$lon,lat=slp$lat,tim=slp$tim,lev=slp$lev,
          v.name=slp$v.name,id.x=slp$id.x,id.t=slp$id.t,
          yy=slp$yy,mm=slp$mm,dd=slp$dd,n.fld=1,
          id.lon=rep(slp$v.name,nx),id.lat=rep(slp$v.name,ny),
          attributes=dat.att)
class(slp) <- c("field")
```

```
## End(Not run)
```

```
retrieve.nc
```

```
Retrieve data from a netCDF file
```

Description

retrieve.nc reads a netCDF file and picks out vectors that look like lngitude, latitude and time. Returns the first 3-D field in the file. See also `cdfextract` (large data files).

This version uses `cdfcont` to obtain vital meta data for handling the data in the netCDF file and constructing a 'field' object.

This routine reequres the `ncdf`-package is currently only available for Linux systems at the CRAN web sites, but the Windows version can can be obtained from <http://www.stats.ox.ac.uk/pub/RWin/>.

`fixField` is a function for eg changing the time stamp if the given time origin is wrong.

Usage

```
retrieve.nc(filename=file.path("data", "air.mon.mean.nc"), v.nam="AUTO",
            l.scale=TRUE, greenwich=TRUE, silent=FALSE,
            x.nam="lon", y.nam="lat", z.nam="lev", t.nam="tim",
            x.rng=NULL, y.rng=NULL, z.rng=NULL, t.rng=NULL,
            force.chron=TRUE, force365.25=FALSE, regular=TRUE,
            daysayear=365.25, forceBC=TRUE,
            use.cdfcont=FALSE, torg=NULL, t.unit=NULL, miss2na=TRUE)
fixField(x, torg=NULL, t.unit=NULL, scal=NULL, ofs=NULL,
         x.rng=NULL, y.rng=NULL, z.rng=NULL, t.rng=NULL, greenwich=TRUE)
test.retrieve.nc(filename="sst.wkmean.1981-1989.nc")
monthly(x, param="t2m", method="mean")
```

Arguments

filename	name of netCDF file.
v.nam	name of variable. "AUTO" -> smart search.
l.scale	'TRUE' uses scaling.factor and add.offset.
greenwich	'TRUE' centres maps on Greenwich meridian (0 deg E).
x.nam	Name of x-dimension.
y.nam	Name of y-dimension.
z.nam	Name of z-dimension.
t.nam	Name of time-axis.
x.rng	Region to extract.
y.rng	Region to extract.

z.rng	Region to extract.
t.rng	Time interval to extract. Numerical values are used to identify indices, e.g. <code>as.numeric(1)</code> refers to first field, <code>as.numeric(2)</code> the second field, etc. Character strings, on the other hand, refers to date. E.g. "1-Jan-1990", or "1990" (see datestr2num for various formats).
force.chron	Check for monotonic chronological order (no jumping back and forth in time).
force365.25	TRUE forces a natural 365.25 day year as opposed to a 360-day model year. '-1' forces a 360-day year (commonly used in climate modelling).
regular	TRUE for regular spacing in time (i.e. no skipping, but one field every month or one field every day).
daysyear	Number of days in the year on average.
forceBC	TRUE for not accepting year 0 (see e.g. Press et al. (1989), Numerical Recipes in Pascal, Cambridge).
x	a field object.
scal	scaling factor
offs	constant offset
silent	To turn off printing to console.
param	Parameter to convert to monthly value
use.cdfcont	Flag for Linux versions only: if TRUE use old lines calling 'cdfcont()'
torg	Time origin, such as the 'time_origin' attribute in netCDF files. e.g. '15-Dec-1949'. A NULL value (default) will try to detect from the file header.
t.unit	Time unit, similar to the 'time_unit' attribute in netCDF files. e.g. 'day'. A NULL value (default) will try to detect from the file header.
method	Function to use when computing monthly values (e.g. mean, sd, max, min)
miss2na	FLAG: TRUE - set missing value from netCDF meta data to NA

Value

A "field.object" list:

dat	a 3-D matrix with the data.
lon	a vector of longitudes.
lat	a vector of latitudes.
tim	a vector of times from time.0 (see attributes).
lev	a vector of vertical levels (NULL for single level).
v.name	variable name.
id.x	ID labels for the spatial grid (for mixed fields, see mixFields).
id.t	ID labels for the time axis (for combined fields).
yy	a vector of years corresponding to tim.
mm	a vector of months corresponding to tim.
dd	a vector of days corresponding to tim.
n.fld	number of fields (for mixed fields, see mixFields).
id.lon	ID labels along the longitudes (for mixed fields, see mixFields).
id.lat	ID labels along the latitudes (for mixed fields, see mixFields).

Author(s)

R.E. Benestad

Examples

```
## Not run:
X.1 <- retrieve.nc("data/mpi-gsdiot2m.nc",
                  x.rng=c(-60,40),y.rng=c(50,75))
X.2 <- retrieve.nc("data/mpi-gsdiot_slp.nc",
                  x.rng=c(-60,40),y.rng=c(50,75))

# The definition of a 'field' object is:
ny<-length(slp$lat); nx<-length(slp$lon)
slp <- list(dat=X.2$dat,lon=X.2$lon,lat=X.2$lat,tim=X.2$tim,lev=X.2$lev,
            v.name=X.2$v.nam,id.x=X.2$id.x,id.t=X.2$id.t,
            yy=X.2$yy,mm=X.2$mm,dd=X.2$dd,n.fld=1,
            id.lon=rep(X.2$v.name,nx),id.lat=rep(X.2$v.name,ny),
            attributes=dat.att)
class(slp) <- c("field")

# For reading the IPCC FoAR netCDF files that uses a 365-day year (no leap years)
# and starts on time count year 0:
gcm <- retrieve.nc(fname,v.nam="tas",x.rng=c(-50,50),y.rng=c(30,75),forceBC=FALSE)

## End(Not run)
```

reverse

*Reverse***Description**

Reverses the order of a vector. `reverse.sort` returns a sorted vector in reverse order.

Usage

```
reverse(x)
reverse.sort(x)
```

Arguments

x a vector

Value

a vector.

Author(s)

R.E. Benestad

Examples

```
reverse(c(1,3,5,7,2,4,6,8))      # 8 6 4 2 7 5 3 1
reverse.sort(c(1,3,5,7,2,4,6,8)) # 8 7 6 5 4 3 2 1
```

rotate

*Rotate spherical coordinates***Description**

The function uses the 3-dimensional space to find vectors describing the locations on a sphere that correspond to a given longitude and latitude. Two reference points are given, one for the new 'north pole': $\vec{r}_0 = [\cos(\phi_0) \cos(\theta_0), \sin(\phi_0), \cos(\phi_0) \sin(\theta_0)]$ and one for a point on the same meridian \vec{r}_x , but closer to the equator. The vector corresponding to each of the longitude-latitude point is given by: $\vec{r}_i = [\cos(\phi_i) \cos(\theta_i), \sin(\phi_i), \cos(\phi_i) \sin(\theta_i)]$.

The default method is based on "Cayley-Klein" ([http://en.wikipedia.org/wiki/Rotation_representation_\(mathematics\)](http://en.wikipedia.org/wiki/Rotation_representation_(mathematics))), but an optional approach uses inner products of 3D-vectors where the new spherical coordinates are computed as follows: new latitude (phi) is the angle between the the vectors \vec{r}_0 and \vec{r}_i and estimated from the inner products (http://en.wikipedia.org/wiki/Inner_product_space). The new longitude (theta) is the angle between the the vectors $\vec{r}_i - \vec{r}_0$ and $\vec{r}_x - \vec{r}_0$. The angles are estimated by taking the inner-product and the arc-cosine.

Usage

```
rotate(lon, lat, lon.0=NULL, lat.0=NULL, method = "Cayley-Klein", test = TRUE)
```

Arguments

lon	Longitude coordinates to be rotated.
lat	Latitude coordinates to be rotated.
lon.0	Longitude coordinate of point corresponding to new 'north' pole.
lat.0	Latitude coordinate of point corresponding to new 'north' pole.
method	Method to use for rotation, e.g. "Cayley-Klein" or "old"
test	Flag for testing the function.

Value

list containing phi and theta, the new spherical coordinates.

Author(s)

R.E. Benestad

Examples

```
data(addland1)
rot <- rotate(lon.cont,lat.cont,lon.0=0,lat.0=0)
```

satellite

Satellite view / polar stereographic

Description

Produces polar stereographic maps / satellite views. `stereogr` is simpler, newer, and faster than `satellite`.

Usage

```
stereogr(map.obj,NH=TRUE,lat.0=0,inv.col=FALSE,levels=NULL,sym=TRUE,
         dr=0.01,main=NULL)
satelliteOld(map.obj,col="black",lwd=2,lty=1,add=FALSE,
             las = 1,lon.0=NULL,lat.0=NULL,method="normal",
             ni=100,nj=100,n.nearest=4,max.dist=3,landdata=addland2)
map2sphere(z,X=seq(-180,180,by=1),Y=seq(-90,90,by=1),pal="rainbow",
          nlevs=21,breaks=NULL,quick=FALSE,add=FALSE)
satellite(map,lon.0=0,lat.0=0,pal="rainbow",nlevs=21,breaks=NULL,
          quick=FALSE,add=FALSE)
```

Arguments

<code>map.obj</code>	a map object (mapField).
<code>col</code>	contour colours.
<code>lwd</code>	contour line width.
<code>lty</code>	contour line style.
<code>add</code>	FLAG: true adds contour onto old plot.
<code>las</code>	see par .
<code>lon.0</code>	Reference longitude: centre of map; NULL selects automatically.
<code>lat.0</code>	Reference latitude: centre of map; NULL selects automatically.
<code>method</code>	A choice between "normal", "polarstereo", and "distance".
<code>ni</code>	Number of grid points along x-axis in new grid.
<code>nj</code>	Number of grid points along y-axis in new grid.
<code>n.nearest</code>	Number of points to use in re-gridding.
<code>max.dist</code>	The maximum inter-point distance used for re-gridding.
<code>landdata</code>	"addland" gives higher resolution for coast lines, but is slower.
<code>inv.col</code>	Inverse color scheme.
<code>NH</code>	Northern Hemisphere?.

levels	contour/shading levels.
main	Main title - see plot .
sym	symmetric colour scale?.
dr	used for masking interpolated values in data voids. Look of result may be sensitive to spatial grid resolution.
z	The gridded data.
X	x-coordinate of z
Y	y-coordinate of z
map	mpa object to be mapped.
pal	palette.
quick	Faster high-resolution plot.
nlevs	number of levels for contour/shading levels
breaks	break points for contour/shading levels

Author(s)

R.E. Benestad

Examples

```
## Not run:
x <- retrieve.nc("T2M_p.nc")
a <- mapField(x)
satellite(a)

## End(Not run)
```

Description

After von Storch & Zwiers (1999), *Statistical Analysis in Climate Research*, p. 312

Usage

```
SSA(x,m,plot=TRUE,main="SSA analysis",sub="",LINPACK=TRUE,
    param = "t2m", anom = TRUE,i.eof=1)
plotSSA(ssa,main="SSA analysis",sub="")
```

Arguments

x	A station or eof object.
m	Window length.
plot	Flag: plot the diagnostics.
LINPACK	'TRUE': svd; 'FALSE':La.svd
main	main title (see <code>link{plot}</code>).
sub	subtitle (see <code>link{plot}</code>).
ssa	An 'SSA' object returned by SSA().
param	Which parameter ("daily.station.record") to use: "precip", "t2m" or other.
anom	TRUE if analysis on anomalies
i.eof	If x is an eof-object, which PC to use.

Value

A SSA object: An `link{svd}` object with additional parameters: m (window length), nt (original length of series), Nm (effective length of series= nt - m), anom (FLAG for use of anomaly), param (name of parameter, typically 'precip' or 't2m'), station (the station object to which SSA is applied).

Author(s)

R.E. Benestad

Examples

```
## Not run:
data(DNMI.t2m)
eof.1 <- EOF(DNMI.t2m,mon=1)
pop <- POP(eof.1)

## End(Not run)
```

station.obj

Make monthly climate station series object.

Description

Create a station object for use as predictand in empirical downscaling on monthly data. Also see [station.obj.dm](#).

The ele codes are taken from the Nordklm and NACD data sets: 101='mean T(2m)',111='mean maximum T(2m)',112='highest maximum T(2m)',113='day of Th date Thd',121='mean minimum T(2m)',122='lowest minimum T(2m)',123='day of Tl date Tld',401='mean SLP',601='monthly accum. precip.',602='maximum precip.',701='Number of days with snow cover (> 50% covered) days dsc',801='Mean cloud cover

Usage

```
station.obj(x,yy,obs.name,unit,ele=NULL, mm=NULL,
            station=NULL,lat=NULL,lon=NULL,alt=NULL,
            location="unspecified",wmo.no=NULL,
            start=NULL,yy0=NULL,country=NULL,ref=NULL)
```

Arguments

x	the data: a matrix of 12 columns holding the observations of each calendar month: column 1 holds January values, col 2 holds February, col 12 holds December values.
yy	A vector holding the year of observation of the same length as each of the 12 columns. Or a vector with the same length as the data if mm is given.
mm	a vector of months with the same length as the data (optional).
obs.name	the name of observation (e.g. "Temperature").
unit	the unite of observation (e.g. "deg C").
ele	element code.
station	local (national) station number.
lat	latitude.
lon	longitude.
alt	altitude.
location	name of location.
wmo.no	WMO number of station.
start	start of measurements.
yy0	first year of record.
country	name of country.
ref	reference to the data.

Value

a "monthly.station.record"-class object.

val	The monthly values (a 12-column matrix with one column for each year)
station	station number.
yy	The years of observation (vector).
lat,lon	Latitude and longitude of the location.
x.0E65N,y.0E65N	Distance in km from 0E, 65N.
location	Name of location .
wmo.no	WMO number.
start	Start of observatins from this location.
yy0	First year in current record.
ele	Code of the element.
obs.name	Name of the element.
unit	Unit of the element.

country	The country in which the location is located.
quality	Code/description for data quality.
found	Flag: T-> the data requested was found.
ref	Reference for the data set.

Author(s)

R.E. Benestad

Examples

```
## Not run:
a <- read.table("data/bjornholt.dat",
               col.names=c("station","year","month","rr",
                           "tam","sam","sdm","uum","pom","tax","tan"))
obs <- station.obj(x=a$rr,yy=a$year,mm=a$month,
                 obs.name="Precipitation",unit="mm",ele=601,
                 lat=60.03,lon=10.41,alt=360,
                 station=a$station[1],location="Bjornholt",
                 country="Norway",ref="met.no Climate data base")

plot(obs,mon=11)

## End(Not run)
```

station.obj.dm

*Make daily climate station series object.***Description**

Create a station object for use as predictand in empirical downscaling on monthly data. Also see [station.obj](#).

Usage

```
station.obj.dm(t2m,precip,dd,mm,yy,
              obs.name=NULL,unit=NULL,ele=NULL,
              station=NULL,lat=NULL,lon=NULL,alt=NULL,
              location="unspecified",wmo.no=NULL,
              start=NULL,yy0=NULL,country=NULL,ref=NULL)
```

Arguments

t2m	a vector holding daily mean temperature.
precip	a vector holding daily precipitation.
dd	a vector holding day of month.
mm	a vector holding the month.

yy	a vector holding the year.
obs.name	the name of observation: eg c("Daily mean temperature", "Daily precipitation").
unit	the unite of observation: eg c("deg C", "mm/day").
ele	element code: eg c("tam", "rr").
station	local (national) station number.
lat	latitude.
lon	longitude.
alt	altitude.
location	name of location.
wmo.no	WMO number of station.
start	start of measurements.
yy0	first year of record.
country	name of country.
ref	reference to the data.

Value

a "daily.station.record"-class object.

t2m	a vector holding daily mean temperature.
precip	a vector holding daily precipitation.
day	a vector holding day of month.
month	a vector holding the month.
year	a vector holding the year.
obs.name	the name of observation: eg c("Daily mean temperature", "Daily precipitation").
unit	the unite of observation: eg c("deg C", "mm/day").
ele	element code: eg c("tam", "rr").
station	local (national) station number.
lat	latitude.
lon	longitude.
alt	altitude.
location	name of location.
wmo.no	WMO number of station.
start	start of measurements.
yy0	first year of record.
country	name of country.
ref	reference to the data.

Author(s)

R.E. Benestad

Examples

```
## Not run:
blindern.raw <- read.table("~/data/stations/blindern_rr_day.dat", header=TRUE)
blindern.raw$rr[blindern.raw$rr < 0] <- NA
yy <- floor(blindern.raw$yyyymmdd/10000)
mm <- floor(blindern.raw$yyyymmdd/100) - 10000*yy
dd <- blindern.raw$yyyymmdd - 100*mm - 10000*yy
blindern <- station.obj.dm(t2m=rep(NA, length(blindern.raw$rr)),
                          precip=blindern.raw$rr,
                          dd=dd, mm=mm, yy=yy,
                          obs.name=c("T(2m)", "recip"),
                          unit=c("deg C", "mm/day"), ele=NULL,
                          station=18700, lat=59.95, lon=10.71, alt=94,
                          location="Oslo-Blindern", wmo.no=NULL,
                          start=NULL, yy0=1937, country="Norway",
                          ref="www.met.no")

## End(Not run)
```

stationmap

Plot climate station map.

Description

Plot a map showing the locations of NACD ([getnacd](#)) and Nordklim ([getnordklim](#)) station network (monthly data).

Usage

```
stationmap(ele=101, NORDKLIM=TRUE, NACD=TRUE, silent=TRUE, names=FALSE,
           name.len=4, x.off=0.1, y.off=-0.5, str.cex=0.7,
           countries=NULL, x.rng=NULL, y.rng=NULL)
```

Arguments

<code>ele</code>	The code for element (101 = T(2m), 401=SLP, 601= precip).
<code>NORDKLIM</code>	Show Nordklim stations.
<code>NACD</code>	Show NACD stations.
<code>silent</code>	Print station names if false.
<code>names</code>	Plot station names if true.
<code>name.len</code>	Station name text length.
<code>x.off</code>	Station name text position.
<code>y.off</code>	Station name text position.
<code>str.cex</code>	Station name text size.
<code>countries</code>	Countries to include.
<code>x.rng</code>	Longitudes to include.
<code>y.rng</code>	Latitudes to include.

Author(s)

R.E. Benestad

Examples

```
## Not run:
stationmap()

## End(Not run)
```

stockholm.t2m	<i>Monthly mean temperature in Stockholm.</i>
---------------	---

Description

A station record of monthly mean temperature Stockholm.

Usage

```
data(stockholm.t2m)
```

Format

list of "monthly.station.record" class:

val	The monthly values (a 12-column matrix with one column for each year).
station	station number.
yy	The years of observation (vector).
lat,lon	Latitude and longitude of the location.
x.0E65N,y.0E65N	Distance in km from 0E, 65N.
location	Name of location .
wmo.no	WMO number.
start	Start of observations from this location.
yy0	First year in current record.
ele	Code of the element.
obs.name	Name of the element.
unit	Unit of the element.
country	The country in which the location is located.
quality	Code/description for data quality.
found	Flag: T - the data requested was found.
ref	Reference for the data set.

Source

The Nordklm data set: http://www.smhi.se/hfa_coord/nordklm/

References

Tuomenvirta et al. (2001), "Nordklim data set 1.0", DNMI KLIMA 08/01, pp. 26;
The Norwegian Meteorological Institute, P.O. Box 43, 0313 Oslo, Norway (<http://www.met.no>).

Examples

```
data(stockholm.t2m)
```

strip	<i>String operation functions</i>
-------	-----------------------------------

Description

The function strips off trailing space (strips the strings by cutting off at the first space).

Usage

```
strip(in.str)
```

Arguments

in.str Strings or arrays of strings.

Value

converted strings or arrays of strings.

Author(s)

R.E. Benestad

Examples

```
print(upper.case(c("qwerty e", "asdf rT"))) # "QWERTY" "ASDF"  
print(lower.case(c("QWERTY", "ASDF"))) # "qwErty" "asdf"  
print(strip(c("Hello there!", "Oslo", " ", "NA "))) # "Hello" "Oslo" " " "NA"
```

 tromsoe.t2m

Monthly mean temperature in Tromsoe.

Description

A station record of monthly mean temperature Tromso.

Usage

```
data(tromsoe.t2m)
```

Format

a <- list of "monthly.station.record" class:

val	The monthly values (a 12-column matrix with one column for each year).
station	station number.
yy	The years of observation (vector).
lat,lon	Latitude and longitude of the location.
x.0E65N,y.0E65N	Distance in km from 0E, 65N.
location	Name of location .
wmo.no	WMO number.
start	Start of observatins from this location.
yy0	First year in current record.
ele	Code of theelement.
obs.name	Name of the element.
unit	Unit of the element.
country	The country in which the location is located.
quality	Code/description for data quality.
found	Flag: T - the data requested was found.
ref	Reference for the data set.

Source

The Nordklim data set: http://www.smhi.se/hfa_coord/nordklim/

References

Tuomenvirta et al. (2001), "Nordklim data set 1.0", DNMI KLIMA 08/01, pp. 26;
 The Norwegian Meteorological Institute, P.O. Box 43, 0313 Oslo, Norway (<http://www.met.no>).

Examples

```
data(tromsoe.t2m)
```

upper.case *convert to UPPER CASE*

Description

Converts characters to UPPER CASE.

Usage

```
upper.case(u.case)
```

Arguments

u.case Strings or arrays of strings.

Value

converted strings or arrays of strings.

Author(s)

R.E. Benestad

Examples

```
print(upper.case(c("qwerty e", "asdf rT")))        # "QWERTY" "ASDF"
print(lower.case(c("QWERTY", "ASDF")))            # "qwErty" "asdf"
print(strip(c("Hello there!", "Oslo", " ", "NA "))) # "Hello" "Oslo" " " "NA"
```

what.data *Data information*

Description

The data originally used by the clim.pact package: NACD, Nordklim and NCEP reanalysis and links to their sources.

Usage

```
what.data()
```

Author(s)

R.E. Benestad

Examples

```
what.data()
```

Index

- *Topic **arith**
 - mod, [70](#)
- *Topic **character**
 - instring, [56](#)
 - lower.case, [61](#)
 - num2str, [74](#)
 - strip, [102](#)
 - upper.case, [104](#)
- *Topic **datasets**
 - addland1, [6](#)
 - bergen.dm, [10](#)
 - bergen.t2m, [11](#)
 - DNMI.t2m, [28](#)
 - eof.c, [39](#)
 - eof.dc, [40](#)
 - eof.dmc, [42](#)
 - eof.mc, [44](#)
 - eof.slp, [45](#)
 - helsinki.t2m, [55](#)
 - koebenhavn.t2m, [60](#)
 - narp, [73](#)
 - oslo.dm, [77](#)
 - oslo.t2m, [79](#)
 - stockholm.t2m, [101](#)
 - tromsoe.t2m, [103](#)
- *Topic **data**
 - station.obj, [96](#)
 - station.obj.dm, [98](#)
 - what.data, [104](#)
- *Topic **device**
 - newFig, [74](#)
- *Topic **file**
 - avail.elem, [9](#)
 - cdfcont, [18](#)
 - cdfextract, [19](#)
 - getClimateExplorer, [46](#)
 - getdnmi, [47](#)
 - getecsn, [48](#)
 - getgiss, [49](#)
 - getnacd, [51](#)
 - getnordklim, [52](#)
 - r2cdf, [88](#)
 - retrieve.nc, [90](#)
- *Topic **hplot**
 - addland, [5](#)
 - anomaly.field, [7](#)
 - DSpdf.exp, [35](#)
 - map, [62](#)
 - mapEOF, [63](#)
 - mapField, [64](#)
 - plotDS, [81](#)
 - plotEOF, [82](#)
 - plotField, [83](#)
 - plotStation, [84](#)
 - plumePlot, [86](#)
 - satellite, [94](#)
 - stationmap, [100](#)
- *Topic **manip**
 - addClim, [4](#)
 - adjustEOF, [6](#)
 - anomaly.station, [8](#)
 - caldat, [12](#)
 - catFields, [13](#)
 - CCA, [15](#)
 - coherence, [20](#)
 - composite.field, [21](#)
 - CO_n0E65N, [22](#)
 - corEOF, [23](#)
 - corField, [24](#)
 - datestr2num, [25](#)
 - delta, [26](#)
 - dist2norm, [26](#)
 - ds2station, [34](#)
 - julday, [57](#)
 - km2lat, [58](#)
 - km2lon, [59](#)
 - lagStation, [61](#)
 - meanField, [65](#)

- mergeEOF, 66
 - mergeStation, 67
 - MVR, 71
 - patternIndex, 80
 - POP, 87
 - reverse, 92
 - rotate, 93
 - SSA, 95
 - *Topic **math**
 - distAB, 27
 - *Topic **methods**
 - ABC4ESD, 3
 - *Topic **models**
 - DS, 29
 - mixFields, 68
 - objDS, 75
 - *Topic **multivariate**
 - DS, 29
 - EOF, 36
 - objDS, 75
 - *Topic **spatial**
 - DS, 29
 - EOF, 36
 - objDS, 75
 - *Topic **ts**
 - catFields, 13
 - DS, 29
 - EOF, 36
 - grd.box.ts, 53
 - objDS, 75
-
- ABC4ESD, 3
 - addClim, 4
 - addland, 5
 - addland1, 6
 - addland2 (addland1), 6
 - adjustEOF, 6, 66
 - anm, 31, 76
 - anomaly.field, 7, 7, 8
 - anomaly.station, 4, 8
 - avail.ds (avail.elem), 9
 - avail.elem, 9, 85
 - avail.eofs (avail.elem), 9
 - avail.locs, 85
 - avail.locs (avail.elem), 9
 - avail.preds (avail.elem), 9
-
- bergen.dm, 10
 - bergen.t2m, 11
 - caldat, 12, 57
 - Canonical correlation analysis (CCA), 15
 - catFields, 13, 29, 31, 36, 38, 39, 41, 42, 44, 45, 76
 - CCA, 15, 71
 - cdfcont, 18, 19, 90
 - cdfextract, 19, 90
 - CDFtransfer (DSpdf.exp), 35
 - characters (num2str), 74
 - check.repeat (CCA), 15
 - climate analysis (objDS), 75
 - climate diagnostics (objDS), 75
 - coherence, 20
 - combineEOFs (mixFields), 68
 - combineFields (mixFields), 68
 - composite.field, 21
 - compositeField (composite.field), 21
 - CO_nO_E65N, 22, 58, 59
 - continental coast line (addland1), 6
 - contour, 62, 64
 - corEOF, 23
 - corField, 24
 - daily.station.record (station.obj.dm), 98
 - daily2monthly.field (anomaly.field), 7
 - daily2monthly.station (anomaly.station), 8
 - datestr2num, 25, 91
 - delta, 26
 - delta function (delta), 26
 - dist2norm, 26
 - distAB, 27
 - DNMI.slp (DNMI.t2m), 28
 - DNMI.sst (DNMI.t2m), 28
 - DNMI.t2m, 28
 - downscaling (objDS), 75
 - DS, 13, 29, 34, 38, 39, 41, 42, 44, 45, 69, 75–77, 81, 86
 - ds2station, 34
 - DSpdf.exp, 35
 - ele (station.obj), 96
 - Empirical orthogonal Functions (EOF), 36
 - empiricalRanking (DSpdf.exp), 35
 - EOF, 6, 13, 14, 29, 31, 36, 38, 40, 41, 43, 44, 46, 63, 66, 69, 71, 82
 - eof (EOF), 36
 - eof.c, 39

- eof.c_data (eof.c), 39
- eof.dc, 14, 40
- eof.dc_data (eof.dc), 40
- eof.dmc, 14, 42
- eof.dmc_data (eof.dmc), 42
- eof.mc, 44
- eof.mc_data (eof.mc), 44
- eof.slp, 45
- EOF2field, 14, 28, 71
- EOF2field (EOF), 36
- exp.par (DSpdf.exp), 35
- ExtEOF (EOF), 36

- family, 31
- fastRegrid, 76
- fastRegrid (catFields), 13
- field correlation (corField), 24
- field correlation, PCA (corEOF), 23
- field.object (retrieve.nc), 90
- filled.contour, 37
- fixField (retrieve.nc), 90

- getClimateExplorer, 46
- getdnmi, 46, 47, 49, 51, 52, 67, 85
- getecsn, 48
- getgiss, 48, 49
- getHadObs (getnordklim), 52
- getnacd, 9, 46, 47, 49, 51, 52, 67, 77, 85, 100
- getnarp, 73
- getnarp (getgiss), 49
- getnordklim, 9, 17, 46, 47, 49, 51, 52, 85, 100
- glm, 31
- grd.box.ts, 53, 83

- helsinki.t2m, 55

- instring, 56

- julday, 12, 57

- km2lat, 22, 58, 59
- km2lon, 22, 58, 59
- koebenhavn.t2m, 60

- lagField (EOF), 36
- lagStation, 61
- lat (C0n0E65N), 22
- lat.cont (addland1), 6
- lon (C0n0E65N), 22
- lon.cont (addland1), 6

- lower.case, 61

- map, 62, 65, 81
- map2sphere (satellite), 94
- mapEOF, 63
- mapField, 64, 83, 94
- meanField, 65
- mergeEOF, 66
- mergeStation, 66, 67
- meta (getnordklim), 52
- meta.nacd (getnacd), 51
- mixFields, 29, 31, 36, 38, 39, 41, 42, 44, 45, 68, 91
- mod, 70
- monthly (retrieve.nc), 90
- monthly.station.record (station.obj), 96
- Multivariate regression analysis (MVR), 71
- MVR, 37, 71

- nacd.meta (narp), 73
- narp, 73
- ncdf, 90
- newFig, 62, 64, 74
- nordklim.meta (narp), 73
- norm2dist (dist2norm), 26
- num2str, 74

- objDS, 30, 75, 76
- oslo.dm, 77
- oslo.t2m, 79

- par, 37, 62, 64, 94
- patternIndex, 80
- PCA (EOF), 36
- pdf, 74
- plot, 36, 54, 62, 64, 81–83, 85, 95
- plotCCA (CCA), 15
- plotDS, 81
- plotDSobj (objDS), 75
- plotEOF, 38, 39, 41, 42, 44, 45, 63, 82
- plotField, 83
- plotPOP (POP), 87
- plotSSA (SSA), 95
- plotStation, 54, 84
- plumePlot, 86
- polar stereographic (satellite), 94
- POP, 87
- postscript, 74

preClim.station (addClim), 4
predictCCA (CCA), 15
principal component analysis (EOF), 36
Principal Oscillating Patterns (POP), 87
Psi (CCA), 15

r2cdf, 19, 88
retrieve.nc, 19, 28, 37, 54, 64, 75, 90
reverse, 92
rgb, 24
RMSE (objDS), 75
rotate, 93

satellite, 81, 94
satelliteOld (satellite), 94
Singular Spectrum Analysis (SSA), 95
SSA, 95
station.obj, 31, 54, 61, 75, 85, 96, 98
station.obj.dm, 31, 54, 75, 96, 98
stationmap, 100
stations2field (CCA), 15
stereogr (satellite), 94
stockholm.t2m, 101
strings (num2str), 74
strip, 102
svd, 15
Sys.info, 18

test.retrieve.nc (retrieve.nc), 90
testCCA (CCA), 15
testcoherence (coherence), 20
testfastRegrid (catFields), 13
tromsoe.t2m, 103

upper.case, 104

what.data, 104

x11, 74