R documentation

of
‘/home/bertvs/fileserver/home/ARCHIVE_bertvs/R/MEDSCOPE/WP3.4/cstools/
etc.

May 16, 2019

`areave_data` *Sample Of Experimental And Observational Climate Data Averaged Over A Region*

Description

This sample data set contains area-averaged seasonal forecast and corresponding observational data from the Copernicus Climate Change ECMWF-System 5 forecast system, and from the Copernicus Climate Change ERA-5 reconstruction. Specifically, for the 'tas' (2-meter temperature) variable, for the 15 first forecast ensemble members, monthly averaged, for the 3 first forecast time steps (lead months 1 to 4) of the November start dates of 2000 to 2005, for the Mediterranean region (27N-48N, 12W-40E).

Details

It is recommended to use the data set as follows:

```
require(zeallot)
c(exp, obs)
```

The ‘CST_Load‘ call used to generate the data set in the infrastructure of the Earth Sciences Department of the Barcelona Supercomputing Center is shown next. Note that ‘CST_Load‘ internally calls ‘s2dverification::Load‘, which would require a configuration file (not provided here) expressing the distribution of the 'system5c3s' and 'era5' NetCDF files in the file system.

```
library(CSTools)
require(zeallot)

startDates <- c('20001101', '20011101', '20021101',
                '20031101', '20041101', '20051101')

areave_data <-
  CST_Load(
    var = 'tas',
```

```
  exp = 'system5c3s',
  obs = 'era5',
  nmember = 15,
  sdates = startDates,
  leadtimemax = 3,
  latmin = 27, latmax = 48,
  lonmin = -12, lonmax = 40,
  output = 'areave',
  nprocs = 1
)
```

Author(s)

Nicolau Manubens <nicolau.manubens@bsc.es>

`CST_Anomaly` *Anomalies relative to a climatology along selected dimension with or without cross-validation*

Description

This function computes the anomalies relative to a climatology computed along the selected dimension (usually starting dates or forecast time) allowing the application or not of crossvalidated climatologies. The computation is carried out independently for experimental and observational data products.

Usage

```
CST_Anomaly(exp = NULL, obs = NULL, cross = FALSE, memb = TRUE,
  dim_anom = 3)
```

Arguments

`exp`	an object of class `s2dv_cube` as returned by `CST_Load` function, containing the seasonal forecast experiment data in the element named `$data`.
`obs`	an object of class `s2dv_cube` as returned by `CST_Load` function, containing the observed data in the element named `$data`.'
`cross`	A logical value indicating whether cross-validation should be applied or not. Default = FALSE.
`memb`	A logical value indicating whether Clim() computes one climatology for each experimental data product member(TRUE) or it computes one sole climatology for all members (FALSE). Default = TRUE.
`dim_anom`	An integer indicating the dimension along which the climatology will be computed. It usually corresponds to 3 (sdates) or 4 (ftime). Default = 3.

Value

A list with two S3 objects, 'exp' and 'obs', of the class 's2dv_cube', containing experimental and date-corresponding observational anomalies, respectively. These 's2dv_cube's can be ingested by other functions in CSTools.

Author(s)

Perez-Zanon Nuria, <nuria.perez@bsc.es>

Pena Jesus, <jesus.pena@bsc.es>

See Also

Ano_CrossValid, Clim and CST_Load

Examples

```
# Example 1:
mod <- 1 : (2 * 3 * 4 * 5 * 6 * 7)
dim(mod) <- c(dataset = 2, member = 3, sdate = 4, ftime = 5, lat = 6, lon = 7)
obs <- 1 : (1 * 1 * 4 * 5 * 6 * 7)
dim(obs) <- c(dataset = 1, member = 1, sdate = 4, ftime = 5, lat = 6, lon = 7)
lon <- seq(0, 30, 5)
lat <- seq(0, 25, 5)
exp <- list(data = mod, lat = lat, lon = lon)
obs <- list(data = obs, lat = lat, lon = lon)
attr(exp, 'class') <- 's2dv_cube'
attr(obs, 'class') <- 's2dv_cube'

anom1 <- CST_Anomaly(exp = exp, obs = obs, cross = FALSE, memb = TRUE)
str(anom1)
anom2 <- CST_Anomaly(exp = exp, obs = obs, cross = TRUE, memb = TRUE)
str(anom2)

anom3 <- CST_Anomaly(exp = exp, obs = obs, cross = TRUE, memb = FALSE)
str(anom3)

anom4 <- CST_Anomaly(exp = exp, obs = obs, cross = FALSE, memb = FALSE)
str(anom4)
```

CST_BiasCorrection *Bias Correction based on the mean and standard deviation adjustment*

Description

This function applies the simple bias adjustment technique described in Torralba et al. (2017). The adjusted forecasts have an equivalent standard deviation and mean to that of the reference dataset.

Usage

```
CST_BiasCorrection(exp, obs)
```

Arguments

exp	an object of class s2dv_cube as returned by CST_Load function, containing the seasonal forecast experiment data in the element named $data
obs	an object of class s2dv_cube as returned by CST_Load function, containing the observed data in the element named $data.

Value

an object of class s2dv_cube containing the bias corrected forecasts in the element called $data with the same dimensions of the experimental data.

Author(s)

Verónica Torralba, <veronica.torralba@bsc.es>

References

Torralba, V., F.J. Doblas-Reyes, D. MacLeod, I. Christel and M. Davis (2017). Seasonal climate prediction: a new source of information for the management of wind energy resources. Journal of Applied Meteorology and Climatology, 56, 1231-1247, doi:10.1175/JAMC-D-16-0204.1. (CLIM4ENERGY, EUPORIAS, NEWA, RESILIENCE, SPECS)

Examples

```
# Example
# Creation of sample s2dverification objects. These are not complete
# s2dverification objects though. The Load function returns complete objects.
mod1 <- 1 : (1 * 3 * 4 * 5 * 6 * 7)
dim(mod1) <- c(dataset = 1, member = 3, sdate = 4, ftime = 5, lat = 6, lon = 7)
obs1 <- 1 : (1 * 1 * 4 * 5 * 6 * 7)
dim(obs1) <- c(dataset = 1, member = 1, sdate = 4, ftime = 5, lat = 6, lon = 7)
lon <- seq(0, 30, 5)
lat <- seq(0, 25, 5)
exp <- list(data = mod1, lat = lat, lon = lon)
obs <- list(data = obs1, lat = lat, lon = lon)
attr(exp, 'class') <- 's2dv_cube'
attr(obs, 'class') <- 's2dv_cube'
a <- CST_BiasCorrection(exp = exp, obs = obs)
str(a)
```

CST_Calibration *Forecast Calibration based on the ensemble inflation*

Description

This function applies a variance inflation technique described in Doblas-Reyes et al. (2005) in leave-one-out cross-validation. This bias adjustment method produces calibrated forecasts with equivalent mean and variance to that of the reference dataset, but at the same time preserve reliability.

Usage

```
CST_Calibration(exp, obs)
```

Arguments

exp	an object of class s2dv_cube as returned by CST_Load function, containing the seasonal forecast experiment data in the element named $data.
obs	an object of class s2dv_cube as returned by CST_Load function, containing the observed data in the element named $data.

Value

an object of class s2dv_cube containing the calibrated forecasts in the element $data with the same dimensions of the experimental data.

Author(s)

Verónica Torralba, <veronica.torralba@bsc.es>

References

Doblas-Reyes F.J, Hagedorn R, Palmer T.N. The rationale behind the success of multi-model ensembles in seasonal forecasting-II calibration and combination. Tellus A. 2005;57:234-252. doi:10.1111/j.1600-0870.2005.00104.x

See Also

CST_Load

Examples

```
# Example
# Load data using CST_Load or use the sample data provided:
library(zeallot)
c(exp, obs) %<-% areave_data
exp_calibrated <- CST_Calibration(exp = exp, obs = obs)
str(exp_calibrated)
```

CST_CategFc *Make categorical forecast based on a multi-model forecast with potential for calibrate*

Description

This function converts an multi-model ensemble forecast into a categorical forecast. The categories are quantiles (e.g. terciles) and different methods are included to combine the different models. These include ensemble pooling (equal weight for all ensemble members), model combination (equal weight for each model) and model weighting. The weighting method is described in Rajagopalan et al. (2002), Robertson et al. 2004 and Van Schaeybroeck and Vannitsem (2019). The method is based on optimization of the ignorance score and weighs the occurence probabilities of the different models and the climatology.

Usage

```
CST_CategFc(exp, obs, method = "pool", amt.cat = 3, ...)
```

Arguments

exp	an object of class s2dv_cube as returned by CST_Load function, containing the seasonal forecast experiment data in the element named $data.
obs	an object of class s2dv_cube as returned by CST_Load function, containing the observed data in the element named $data.

method	method used to produce the categorical forecast, can be either "pool", "comb", "mmw" or "obs". The method pool assumes equal weight for all ensemble members while the method comb assumes equal weight for each model. The weighting method is descirbed in Rajagopalan et al. (2002), Robertson et al. 2004 and Van Schaeybroeck and Vannitsem (2019). The obs method returns the categories of the observation. The models corresponding to each ensemble member should be introduced through dimnames of the exp object. All results are obtained through cross validation.
amt.cat	is the amount of categories. Equally-sized quantiles will be calculated based on the amount of categories.

Value

an object of class s2dv_cube containing the categorical forecasts in the element called $data.

Author(s)

Bert Van Schaeybroeck, <bertvs@meteo.be>

References

Rajagopalan, B., Lall, U., & Zebiak, S. E. (2002). Categorical climate forecasts through regularization and optimal combination of multiple GCM ensembles. Monthly Weather Review, 130(7), 1792-1811.

Robertson, A. W., Lall, U., Zebiak, S. E., & Goddard, L. (2004). Improved combination of multiple atmospheric GCM ensembles for seasonal prediction. Monthly Weather Review, 132(12), 2732-2744.

Van Schaeybroeck, B., & Vannitsem, S. (2019). Postprocessing of Long-Range Forecasts. In Statistical Postprocessing of Ensemble Forecasts (pp. 267-290).

Examples

```
# Example 1
library(abind)
mod1 <- 1 : (1 * 3 * 4 * 5 * 6 * 7)
dim(mod1) <- c(dataset = 1, member = 3, sdate = 4, ftime = 5, lat = 6, lon = 7)
dimnames(mod1)[[2]] <- c("MF", "MF", "UKMO")
obs1 <- 1 : (1 * 1 * 4 * 5 * 6 * 7)
dim(obs1) <- c(dataset = 1, member = 1, sdate = 4, ftime = 5, lat = 6, lon = 7)
lon <- seq(0, 30, 5)
lat <- seq(0, 25, 5)
exp <- list(data = mod1, lat = lat, lon = lon)
obs <- list(data = obs1, lat = lat, lon = lon)
attr(exp, 'class') <- 's2dv_cube'
attr(obs, 'class') <- 's2dv_cube'
a <- CST_CategFc(exp = exp, obs = obs, amt.cat = 3, method = "mmw")

# Example 2
library(abind)
amt.dataset <- 1
lst.dataset <- paste0("dataset_", seq(1,amt.dataset))
lst.model <- c("MF", "ECMWF", "UKMO")
amt.model <- length(lst.model)
```

```
amt.member.per.model <- c(2, 5, 7)
amt.member <- sum(amt.member.per.model)
lst.member <- unlist(mapply(function(x, y){rep(x, y)}, lst.model,
 amt.member.per.model, USE.NAMES = FALSE))
lst.member.cat <- unlist(mapply(function(x, y){rep(x, y)},
 seq(1,amt.model), amt.member.per.model, USE.NAMES = FALSE))
msk.mdl <- sapply(lst.model,function(x){lst.member == x})
amt.sdate <- 100
lst.sdate <- paste0(1980 + seq(1, amt.sdate), "-01-01")
amt.ftime <- 3
lst.ftime <- paste0("month_", seq(1, amt.ftime))
amt.lon <- 2
lst.lon <- paste0("lon_",seq(1,amt.lon))
amt.lat <- 3
lst.lat <- paste0("lat_",seq(1,amt.lat))
bias <- c(0.3, 0.6, 0.3)
slope <- c(1, 1.4, 1.8)
noise.level.fc <- 0.1 * slope
noise.level.obs <- 0.1
amt.data.pts <- amt.dataset * amt.sdate * amt.ftime * amt.lon * amt.lat
obs1 <- array(data = rnorm(amt.data.pts),
 dim = c(dataset = amt.dataset, member = 1, sdate = amt.sdate
   , ftime = amt.ftime, lon = amt.lon, lat = amt.lat),
 dimnames = list(dataset = lst.dataset, member = c("obs"),
   sdate = lst.sdate, ftime = lst.ftime, lon = lst.lon, lat = lst.lat))
tmp.noise.obs <- array(data = rnorm(amt.data.pts, 0, noise.level.obs),
 dim = c(dataset = amt.dataset, member = 1, sdate = amt.sdate,
   ftime = amt.ftime, lon = amt.lon, lat = amt.lat),
 dimnames = list(dataset = lst.dataset, member = c("obs"),
   sdate = lst.sdate, ftime = lst.ftime, lon = lst.lon, lat = lst.lat))
mod1 <- array(data = rnorm(amt.data.pts * amt.member, 0, noise.level.fc),
 dim = c(dataset = amt.dataset, member = amt.member, sdate = amt.sdate,
   ftime = amt.ftime, lon = amt.lon, lat = amt.lat),
 dimnames = list(dataset = lst.dataset, member = lst.member,
   sdate = lst.sdate, ftime = lst.ftime, lon = lst.lon, lat = lst.lat))

for(i.mbr in seq(1, amt.member)){
 mdl.to.use <- lst.member.cat[i.mbr]
 mod1[ , i.mbr, , , , ] <- (obs1[ , 1, , , , ] + mod1[ , i.mbr, , , , ]) /
   slope[mdl.to.use] + bias[mdl.to.use]
}
obs1 <- obs1 + tmp.noise.obs

lon <- seq(0, 30, amt.lon)
lat <- seq(0, 25, amt.lat)
exp <- list(data = mod1, lat = lat, lon = lon)
obs <- list(data = obs1, lat = lat, lon = lon)
attr(exp, 'class') <- 's2dv_cube'
attr(obs, 'class') <- 's2dv_cube'
amt.cat <- 3
cat.comb <- CST_CategFc(exp, obs, amt.cat, method = "comb")
cat.pool <- CST_CategFc(exp, obs, amt.cat, method = "pool")
cat.obs <- CST_CategFc(exp, obs, amt.cat, method = "obs")
cat.mmw <- CST_CategFc(exp, obs, amt.cat, method = "mmw")
rps.comb <- mean((cat.comb$data - cat.obs$data)^2, na.rm = TRUE)
rps.pool <- mean((cat.pool$data - cat.obs$data)^2, na.rm = TRUE)
rps.mmw <- mean((cat.mmw$data - cat.obs$data)^2, na.rm = TRUE)
```

```
rps.clim <- mean((1/amt.cat - cat.obs$data)^2, na.rm = TRUE)
cat("RPSS comb / pool / mmw: ",1 - rps.comb / rps.clim,
 1 - rps.pool / rps.clim, 1 - rps.mmw / rps.clim)
```

CST_Load *CSTools Data Retreival Function*

Description

This function aggregates, subsets and retrieves sub-seasonal, seasonal, decadal or climate projection data from NetCDF files in a local file system or on remote OPeNDAP servers, and arranges it for easy application of the CSTools functions.

Usage

```
CST_Load(...)
```

Arguments

...	Parameters that are automatically forwarded to the 's2dverification::Load' function. See details in '?s2dverification::Load'.

Details

It receives any number of parameters ('...') that are automatically forwarded to the 's2dverification::Load' function. See details in '?s2dverification::Load'.

It is recommended to use this function in combination with the 'zeallot::"

Value

A list with one or two S3 objects, named 'exp' and 'obs', of the class 's2dv_cube', containing experimental and date-corresponding observational data, respectively. These 's2dv_cube's can be ingested by other functions in CSTools. If the parameter 'exp' in the call to 'CST_Load' is set to 'NULL', then only the 'obs' component is returned, and viceversa.

Author(s)

Nicolau Manubens, <nicolau.manubens@bsc.es>

Examples

```
## Not run:
library(zeallot)
startDates <- c('20001101', '20011101', '20021101',
                '20031101', '20041101', '20051101')
c(exp, obs) %<-%
  CST_Load(
    var = 'tas',
    exp = 'system5c3s',
    obs = 'era5',
    nmember = 15,
    sdates = startDates,
```

```
    leadtimemax = 3,
    latmin = 27, latmax = 48,
    lonmin = -12, lonmax = 40,
    output = 'lonlat',
    nprocs = 1
  )

## End(Not run)
```

CST_MultiMetric *Multiple Metrics applied in Multiple Model Anomalies*

Description

This function calculates correlation (Anomaly Correlation Coefficient; ACC), root mean square error (RMS) and the root mean square error skill score (RMSSS) of individual anomaly models and multi-models mean (if desired) with the observations.

Usage

```
CST_MultiMetric(exp, obs, metric = "correlation", multimodel = TRUE)
```

Arguments

exp	an object of class s2dv_cube as returned by CST_Anomaly function, containing the anomaly of the seasonal forecast experiment data in the element named $data.
obs	an object of class s2dv_cube as returned by CST_Anomaly function, containing the anomaly of observed data in the element named $data.
metric	a character string giving the metric for computing the maximum skill. This must be one of the strings 'correlation', 'rms' or 'rmsss.
multimodel	a logical value indicating whether a Multi-Model Mean should be computed.

Value

an object of class s2dv_cube containing the statistics of the selected metric in the element $data which is an array with two datset dimensions equal to the 'dataset' dimension in the exp$data and obs$data inputs. If multimodel is TRUE, the greatest first dimension correspons to the Multi-Model Mean. The third dimension contains the statistics selected. For metric correlation, the third dimension is of length four and they corresponds to the lower limit of the 95% confidence interval, the statistics itselfs, the upper limit of the 95% confidence interval and the 95% significance level. For metric rms, the third dimension is length three and they corresponds to the lower limit of the 95% confidence interval, the RMSE and the upper limit of the 95% confidence interval. For metric rmsss, the third dimension is length two and they corresponds to the statistics itselfs and the p-value of the one-sided Fisher test with Ho: RMSSS = 0.

Author(s)

Mishra Niti, <niti.mishra@bsc.es>

Perez-Zanon Nuria, <nuria.perez@bsc.es>

References

Mishra, N., Prodhomme, C., & Guemas, V. (n.d.). Multi-Model Skill Assessment of Seasonal Temperature and Precipitation Forecasts over Europe, 29-31.http://link.springer.com/10.1007/s00382-018-4404-z

See Also

Corr, RMS, RMSSS and CST_Load

Examples

```
library(zeallot)
mod <- 1 : (2 * 3 * 4 * 5 * 6 * 7)
dim(mod) <- c(dataset = 2, member = 3, sdate = 4, ftime = 5, lat = 6, lon = 7)
obs <- 1 : (1 * 1 * 4 * 5 * 6 * 7)
dim(obs) <- c(dataset = 1, member = 1, sdate = 4, ftime = 5, lat = 6, lon = 7)
lon <- seq(0, 30, 5)
lat <- seq(0, 25, 5)
exp <- list(data = mod, lat = lat, lon = lon)
obs <- list(data = obs, lat = lat, lon = lon)
attr(exp, 'class') <- 's2dv_cube'
attr(obs, 'class') <- 's2dv_cube'
c(ano_exp, ano_obs) %<-% CST_Anomaly(exp = exp, obs = obs, cross = TRUE, memb = TRUE)
a <- CST_MultiMetric(exp = ano_exp, obs = ano_obs)
str(a)
```

CST_MultivarRMSE *Multivariate Root Mean Square Error (RMSE)*

Description

This function calculates the RMSE from multiple variables, as the mean of each variable's RMSE scaled by its observed standard deviation. Variables can be weighted based on their relative importance (defined by the user).

Usage

```
CST_MultivarRMSE(exp, obs, weight = NULL)
```

Arguments

exp	a list of objects, one for each variable, of class s2dv_cube as returned by CST_Anomaly function, containing the anomaly of the seasonal forecast experiment data in the element named $data.
obs	a list of objects, one for each variable (in the same order than the input in 'exp') of class s2dv_cube as returned by CST_Anomaly function, containing the observed anomaly data in the element named $data.
weight	(optional) a vector of weight values to assign to each variable. If no weights are defined, a value of 1 is assigned to every variable.

Value

an object of class s2dv_cube containing the RMSE in the element $data which is an array with two datset dimensions equal to the 'dataset' dimension in the exp$data and obs$data inputs. An array with dimensions: c(number of exp, number of obs, 1 (the multivariate RMSE value), number of lat, number of lon)

Author(s)

Deborah Verfaillie, <deborah.verfaillie@bsc.es>

See Also

RMS and CST_Load

Examples

```
# Creation of sample s2dverification objects. These are not complete
# s2dverification objects though. The Load function returns complete objects.
# using package zeallot is optional:
library(zeallot)
# Example with 2 variables
mod1 <- 1 : (1 * 3 * 4 * 5 * 6 * 7)
mod2 <- 1 : (1 * 3 * 4 * 5 * 6 * 7)
dim(mod1) <- c(dataset = 1, member = 3, sdate = 4, ftime = 5, lat = 6, lon = 7)
dim(mod2) <- c(dataset = 1, member = 3, sdate = 4, ftime = 5, lat = 6, lon = 7)
obs1 <- 1 : (1 * 1 * 4 * 5 * 6 * 7)
obs2 <- 1 : (1 * 1 * 4 * 5 * 6 * 7)
dim(obs1) <- c(dataset = 1, member = 1, sdate = 4, ftime = 5, lat = 6, lon = 7)
dim(obs2) <- c(dataset = 1, member = 1, sdate = 4, ftime = 5, lat = 6, lon = 7)
lon <- seq(0, 30, 5)
lat <- seq(0, 25, 5)
exp1 <- list(data = mod1, lat = lat, lon = lon, Datasets = "EXP1",
             source_files = "file1", Variable = list('pre'))
attr(exp1, 'class') <- 's2dv_cube'
exp2 <- list(data = mod2, lat = lat, lon = lon, Datasets = "EXP2",
             source_files = "file2", Variable = list('tas'))
attr(exp2, 'class') <- 's2dv_cube'
obs1 <- list(data = obs1, lat = lat, lon = lon, Datasets = "OBS1",
             source_files = "file1", Variable = list('pre'))
attr(obs1, 'class') <- 's2dv_cube'
obs2 <- list(data = obs2, lat = lat, lon = lon, Datasets = "OBS2",
             source_files = "file2", Variable = list('tas'))
attr(obs2, 'class') <- 's2dv_cube'

c(ano_exp1, ano_obs1) %<-% CST_Anomaly(exp1, obs1, cross = TRUE, memb = TRUE)
c(ano_exp2, ano_obs2) %<-% CST_Anomaly(exp2, obs2, cross = TRUE, memb = TRUE)
ano_exp <- list(exp1, exp2)
ano_obs <- list(ano_obs1, ano_obs2)
weight <- c(1, 2)
a <- CST_MultivarRMSE(exp = ano_exp, obs = ano_obs, weight = weight)
str(a)
```

CST_RainFARM *RainFARM stochastic precipitation downscaling of a CSTools object*

Description

This function implements the RainFARM stochastic precipitation downscaling method and accepts a CSTools object (an object of the class 's2dv_cube' as provided by 'CST_Load') as input. Adapted for climate downscaling and including orographic correction as described in Terzago et al. 2018.

Usage

```
CST_RainFARM(data, nf, weights = 1, slope = 0, kmin = 1, nens = 1,
  fglob = FALSE, fsmooth = TRUE, time_dim = NULL, verbose = FALSE,
  drop_realization_dim = FALSE)
```

Arguments

data	An object of the class 's2dv_cube' as returned by 'CST_Load', containing the spatial precipitation fields to downscale. The data object is expected to have an element named $data with at least two spatial dimensions named "lon" and "lat" and one or more dimensions over which to compute average spectral slopes (unless specified with parameter slope), which can be specified by parameter time_dim. The number of longitudes and latitudes in the input data is expected to be even and the same. If not the function will perform a subsetting to ensure this condition.
nf	Refinement factor for downscaling (the output resolution is increased by this factor).
weights	Matrix with climatological weights which can be obtained using the CST_RFWeights function. If weights=1. (default) no weights are used. The matrix should have dimensions (lon, lat) in this order. The names of these dimensions are not checked.
slope	Prescribed spectral slope. The default is slope=0. meaning that the slope is determined automatically over the dimensions specified by time_dim.
kmin	First wavenumber for spectral slope (default: kmin=1).
nens	Number of ensemble members to produce (default: nens=1).
fglob	Logical to conserve global precipitation over the domain (default: FALSE).
fsmooth	Logical to conserve precipitation with a smoothing kernel (default: TRUE).
time_dim	String or character array with name(s) of dimension(s) (e.g. "ftime", "sdate", "member" ...) over which to compute spectral slopes. If a character array of dimension names is provided, the spectral slopes will be computed as an average over all elements belonging to those dimensions. If omitted one of c("ftime", "sdate", "time") is searched and the first one with more than one element is chosen.
verbose	Logical for verbose output (default: FALSE).
drop_realization_dim	Logical to remove the "realization" stochastic ensemble dimension (default: FALSE) with the following behaviour if set to TRUE: 1) if nens==1: the dimension is dropped;

2) if nens>1 and a "member" dimension exists: the "realization" and "member" dimensions are compacted (multiplied) and the resulting dimension is named "member";

3) if nens>1 and a "member" dimension does not exist: the "realization" dimension is renamed to "member".

Value

CST_RainFARM() returns a downscaled CSTools object (i.e., of the class 's2dv_cube'). If nens>1 an additional dimension named "realization" is added to the $data array after the "member" dimension (unless drop_realization_dim=TRUE is specified). The ordering of the remaining dimensions in the $data element of the input object is maintained.

Author(s)

Jost von Hardenberg - ISAC-CNR, <j.vonhardenberg@isac.cnr.it>

References

Terzago, S. et al. (2018). NHESS 18(11), 2825-2840. http://doi.org/10.5194/nhess-18-2825-2018 ; D'Onofrio et al. (2014), J of Hydrometeorology 15, 830-843; Rebora et. al. (2006), JHM 7, 724.

Examples

```
#Example using CST_RainFARM for a CSTools object
nf <- 8   # Choose a downscaling by factor 8
exp <- 1 : (2 * 3 * 4 * 8 * 8)
dim(exp) <- c(dataset = 1, member = 2, sdate = 3, ftime = 4, lat = 8, lon = 8)
lon <- seq(10, 13.5, 0.5)
dim(lon) <- c(lon = length(lon))
lat <- seq(40, 43.5, 0.5)
dim(lat) <- c(lat = length(lat))
data <- list(data = exp, lon = lon, lat = lat)
# Create a test array of weights
ww <- array(1., dim = c(8 * nf, 8 * nf))
res <- CST_RainFARM(data, nf, ww, nens=3)
str(res)
#List of 3
# $ data: num [1:4, 1:64, 1:64, 1, 1:2, 1:3] 201 115 119 307 146 ...
# $ lon : num [1:64] 9.78 9.84 9.91 9.97 10.03 ...
# $ lat : num [1:64] 39.8 39.8 39.9 40 40 ...
dim(res$data)
# dataset  member   sdate   ftime     lat     lon    realization
#       1       2       3       4      64      64     3
```

CST_RFSlope *RainFARM spectral slopes from a CSTools object*

Description

This function computes spatial spectral slopes from a CSTools object to be used for RainFARM stochastic precipitation downscaling method and accepts a CSTools object (of the class 's2dv_cube') as input.

Usage

```
CST_RFSlope(data, kmin = 1, time_dim = NULL)
```

Arguments

data	An object of the class 's2dv_cube', containing the spatial precipitation fields to downscale. The data object is expected to have an element named $data with at least two spatial dimensions named "lon" and "lat" and one or more dimensions over which to average these slopes, which can be specified by parameter time_dim.
kmin	First wavenumber for spectral slope (default kmin=1).
time_dim	String or character array with name(s) of dimension(s) (e.g. "ftime", "sdate", "member" ...) over which to compute spectral slopes. If a character array of dimension names is provided, the spectral slopes will be computed as an average over all elements belonging to those dimensions. If omitted one of c("ftime", "sdate", "time") is searched and the first one with more than one element is chosen.

Value

CST_RFSlope() returns spectral slopes using the RainFARM convention (the logarithmic slope of k*|A(k)|^2 where A(k) are the spectral amplitudes). The returned array has the same dimensions as the exp element of the input object, minus the dimensions specified by lon_dim, lat_dim and time_dim.

Author(s)

Jost von Hardenberg - ISAC-CNR, <j.vonhardenberg@isac.cnr.it>

Examples

```
#Example using CST_RFSlope for a CSTools object
exp <- 1 : (2 * 3 * 4 * 8 * 8)
dim(exp) <- c(dataset = 1, member = 2, sdate = 3, ftime = 4, lat = 8, lon = 8)
lon <- seq(10, 13.5, 0.5)
dim(lon) <- c(lon = length(lon))
lat <- seq(40, 43.5, 0.5)
dim(lat) <- c(lat = length(lat))
data <- list(data = exp, lon = lon, lat = lat)
slopes <- CST_RFSlope(data)
dim(slopes)
# dataset  member   sdate
#       1       2       3
slopes
#         [,1]     [,2]     [,3]
#[1,] 1.893503 1.893503 1.893503
#[2,] 1.893503 1.893503 1.893503
```

CST_RFWeights *Compute climatological weights for RainFARM stochastic precipitation downscaling*

Description

Compute climatological ("orographic") weights from a fine-scale precipitation climatology file.

Usage

```
CST_RFWeights(climfile, nf, lon, lat, varname = "", fsmooth = TRUE)
```

Arguments

climfile	Filename of a fine-scale precipitation climatology. The file is expected to be in NetCDF format and should contain at least one precipitation field. If several fields at different times are provided, a climatology is derived by time averaging. Suitable climatology files could be for example a fine-scale precipitation climatology from a high-resolution regional climate model (see e.g. Terzago et al. 2018), a local high-resolution gridded climatology from observations, or a reconstruction such as those which can be downloaded from the WORLDCLIM (http://www.worldclim.org) or CHELSA (http://chelsa-climate.org) websites. The latter data will need to be converted to NetCDF format before being used (see for example the GDAL tools (https://www.gdal.org).
nf	Refinement factor for downscaling (the output resolution is increased by this factor).
lon	Vector of longitudes.
lat	Vector of latitudes. The number of longitudes and latitudes is expected to be even and the same. If not the function will perform a subsetting to ensure this condition.
varname	Name of the variable to be read from reffile.
fsmooth	Logical to use smooth conservation (default) or large-scale box-average conservation.

Value

A matrix containing the weights with dimensions (lon, lat).

Author(s)

Jost von Hardenberg - ISAC-CNR, <j.vonhardenberg@isac.cnr.it>

References

Terzago, S., Palazzi, E., & von Hardenberg, J. (2018). Stochastic downscaling of precipitation in complex orography: A simple method to reproduce a realistic fine-scale climatology. Natural Hazards and Earth System Sciences, 18(11), 2825-2840. http://doi.org/10.5194/nhess-18-2825-2018 .

Examples

```
# Create weights to be used with the CST_RainFARM() or RainFARM() functions
# using an external fine-scale climatology file.

## Not run:
# Specify lon and lat of the input
lon <- seq(10,13.5,0.5)
lat <- seq(40,43.5,0.5)
nf <- 8
ww <- CST_RFWeights("./worldclim.nc", nf, lon, lat, fsmooth = TRUE)

## End(Not run)
```

---

lonlat_data *Sample Of Experimental And Observational Climate Data In Function Of Longitudes And Latitudes*

Description

This sample data set contains gridded seasonal forecast and corresponding observational data from the Copernicus Climate Change ECMWF-System 5 forecast system, and from the Copernicus Climate Change ERA-5 reconstruction. Specifically, for the 'tas' (2-meter temperature) variable, for the 15 first forecast ensemble members, monthly averaged, for the 3 first forecast time steps (lead months 1 to 4) of the November start dates of 2000 to 2005, for the Mediterranean region (27N-48N, 12W-40E). The data was generated on (or interpolated onto, for the reconstruction) a rectangular regular grid of size 360 by 181.

Details

It is recommended to use the data set as follows:

```
require(zeallot)
c(exp, obs)
```

The 'CST_Load' call used to generate the data set in the infrastructure of the Earth Sciences Department of the Barcelona Supercomputing Center is shown next. Note that 'CST_Load' internally calls 's2dverification::Load', which would require a configuration file (not provided here) expressing the distribution of the 'system5c3s' and 'era5' NetCDF files in the file system.

```
library(CSTools)
require(zeallot)

startDates <- c('20001101', '20011101', '20021101',
                '20031101', '20041101', '20051101')

lonlat_data <-
  CST_Load(
    var = 'tas',
    exp = 'system5c3s',
    obs = 'era5',
    nmember = 15,
```

```
  sdates = startDates,
  leadtimemax = 3,
  latmin = 27, latmax = 48,
  lonmin = -12, lonmax = 40,
  output = 'lonlat',
  nprocs = 1
)
```

Author(s)

Nicolau Manubens <nicolau.manubens@bsc.es>

`lonlat_prec` *Sample Of Experimental Precipitation Data In Function Of Longitudes And Latitudes*

Description

This sample data set contains a small cutout of gridded seasonal precipitation forecast data from the Copernicus Climate Change ECMWF-System 5 forecast system, to be used to demonstrate downscaling. Specifically, for the 'pr' (precipitation) variable, for the first 6 forecast ensemble members, daily values, for all 31 days in March following the forecast starting dates in November of years 2010 to 2012, for a small 4x4 pixel cutout in a region in the North-Western Italian Alps (44N-47N, 6E-9E). The data resolution is 1 degree.

Details

The 'CST_Load' call used to generate the data set in the infrastructure of the Marconi machine at CINECA is shown next, working on files which were extracted from forecast data available in the MEDSCOPE internal archive.

```
library(CSTools)

infile <- list(path = '../medscope/nwalps/data/$VAR_NAME$_$START_DATE$_nwalps.nc')
lonlat_prec <- CST_Load('prlr', exp = list(infile), obs = NULL,
                        sdates = c('20101101', '20111101', '20121101'),
                        leadtimemin = 121, leadtimemax = 151,
                        latmin = 44, latmax = 47,
                        lonmin = 5, lonmax = 9,
                        nmember = 25,
                        storefreq = "daily", sampleperiod = 1,
                        output = "lonlat"
                       )$exp
```

Author(s)

Jost von Hardenberg <j.vonhardenberg@isac.cnr.it>

PlotForecastPDF *Plot one or multiple ensemble forecast pdfs for the same event*

Description

This function plots the probability distribution function of several ensemble forecasts for the same event, either initialized at different moments or by different models. Probabilities for tercile categories are computed, plotted in colors and annotated. An asterisk marks the tercile with higher probabilities. Probabilities for extreme categories (above P90 and below P10) can also be included as hatched areas. Individual ensemble members can be plotted as jittered points. The observed value is optionally shown as a diamond.

Usage

```
PlotForecastPDF(fcst, tercile.limits, extreme.limits = NULL,
  obs = NULL, plotfile = NULL, title = "Set a title",
  var.name = "Varname (units)", fcst.names = NULL,
  add.ensmemb = c("above", "below", "no"), color.set = c("ggplot",
  "s2s4e", "hydro"))
```

Arguments

fcst	a dataframe or array containing all the ensember members for each frecast. If 'fcst' is an array, it should have two labelled dimensions, and one of them should be 'members'. If 'fcsts' is a data.frame, each column shoul be a separate forecast, with the rows beeing the different ensemble members.
tercile.limits	an array with P33 and P66 values that define the tercile categories.
extreme.limits	(optional) an array with P10 and P90 values that define the extreme categories. (Default: extreme categories are not shown).
obs	(optional) a number with the observed value. (Default: observation is not shown).
plotfile	(optional) a filename (pdf, png...) where the plot will be saved. (Default: the plot is not saved).
title	a string with the plot title.
var.name	a string with the variable name and units.
fcst.names	(optional) an array of strings with the titles of each individual forecast.
add.ensmemb	either to add the ensemble members 'above' (default) or 'below' the pdf, or not ('no').
color.set	a selection of predefined color sets: use 'ggplot' (default) for blue/green/red, 's2s4e' for blue/grey/orange, or 'hydro' for yellow/gray/blue (suitable for precipitation and inflows).

Value

a ggplot object containing the plot.

Author(s)

Llorenç Lledó <llledo@bsc.es>

Examples

```
fcsts <- data.frame(fcst1 = rnorm(10), fcst2 = rnorm(10, 0.5, 1.2),
                    fcst3 = rnorm(10, -0.5, 0.9))
PlotForecastPDF(fcsts,c(-1,1))

fcsts2 <- array(rnorm(100), dim = c(members = 20, fcst = 5))
PlotForecastPDF(fcsts2, c(-0.66, 0.66), extreme.limits = c(-1.2, 1.2),
               fcst.names = paste0('random fcst ', 1 : 5), obs = 0.7)
```

`PlotMostLikelyQuantileMap` *Plot Maps of Most Likely Quantiles*

Description

This function receives as main input (via the parameter `probs`) a collection of longitude-latitude maps, each containing the probabilities (from 0 to 1) of the different grid cells of belonging to a category. As many categories as maps provided as inputs are understood to exist. The maps of probabilities must be provided on a common rectangular regular grid, and a vector with the longitudes and a vector with the latitudes of the grid must be provided. The input maps can be provided in two forms, either as a list of multiple two-dimensional arrays (one for each category) or as a three-dimensional array, where one of the dimensions corresponds to the different categories.

Usage

```
PlotMostLikelyQuantileMap(probs, lon, lat, cat_dim = "bin",
  bar_titles = NULL, col_unknown_cat = "white", ...)
```

Arguments

`probs`	a list of bi-dimensional arrays with the named dimensions 'latitude' (or 'lat') and 'longitude' (or 'lon'), with equal size and in the same order, or a single tri-dimensional array with an additional dimension (e.g. 'bin') for the different categories. The arrays must contain probability values between 0 and 1, and the probabilities for all categories of a grid cell should not exceed 1 when added.
`lon`	a numeric vector with the longitudes of the map grid, in the same order as the values along the corresponding dimension in `probs`.
`lat`	a numeric vector with the latitudes of the map grid, in the same order as the values along the corresponding dimension in `probs`.
`cat_dim`	the name of the dimension along which the different categories are stored in `probs`. This only applies if `probs` is provided in the form of 3-dimensional array. The default expected name is 'bin'.
`bar_titles`	vector of character strings with the names to be drawn on top of the color bar for each of the categories. As many titles as categories provided in `probs` must be provided.
`col_unknown_cat`	character string with a colour representation of the colour to be used to paint the cells for which no category can be clearly assigned. Takes the value 'white' by default.
`...`	additional parameters to be sent to `PlotCombinedMap` and `PlotEquiMap`.

Author(s)

Veronica Torralba, <veronica.torralba@bsc.es>, Nicolau Manubens, <nicolau.manubens@bsc.es>

See Also

PlotCombinedMap and PlotEquiMap

Examples

```
# Simple example
x <- array(1:(20 * 10), dim = c(lat = 10, lon = 20)) / 200
a <- x * 0.6
b <- (1 - x) * 0.6
c <- 1 - (a + b)
lons <- seq(0, 359.5, length = 20)
lats <- seq(-89.5, 89.5, length = 10)
PlotMostLikelyQuantileMap(list(a, b, c), lons, lats,
                          toptitle = 'Most likely tercile map',
                          bar_titles = paste('% of belonging to', c('a', 'b', 'c')),
                          brks = 20, width = 10, height = 8)

# More complex example
n_lons <- 40
n_lats <- 20
n_timesteps <- 100
n_bins <- 4

# 1. Generation of sample data
lons <- seq(0, 359.5, length = n_lons)
lats <- seq(-89.5, 89.5, length = n_lats)

# This function builds a 3-D gaussian at a specified point in the map.
make_gaussian <- function(lon, sd_lon, lat, sd_lat) {
 w <- outer(lons, lats, function(x, y) dnorm(x, lon, sd_lon) * dnorm(y, lat, sd_lat))
 min_w <- min(w)
 w <- w - min_w
 w <- w / max(w)
 w <- t(w)
 names(dim(w)) <- c('lat', 'lon')
 w
}

# This function generates random time series (with values ranging 1 to 5)
# according to 2 input weights.
gen_data <- function(w1, w2, n) {
 r <- sample(1:5, n,
             prob = c(.05, .9 * w1, .05, .05, .9 * w2),
             replace = TRUE)
 r <- r + runif(n, -0.5, 0.5)
 dim(r) <- c(time = n)
 r
}

# We build two 3-D gaussians.
w1 <- make_gaussian(120, 80, 20, 30)
w2 <- make_gaussian(260, 60, -10, 40)
```

```
# We generate sample data (with dimensions time, lat, lon) according
# to the generated gaussians
sample_data <- multiApply::Apply(list(w1, w2), NULL,
                                 gen_data, n = n_timesteps)$output1

# 2. Binning sample data
prob_thresholds <- 1:n_bins / n_bins
prob_thresholds <- prob_thresholds[1:(n_bins - 1)]
thresholds <- quantile(sample_data, prob_thresholds)

binning <- function(x, thresholds) {
 n_samples <- length(x)
 n_bins <- length(thresholds) + 1

 thresholds <- c(thresholds, max(x))
 result <- 1:n_bins
 lower_threshold <- min(x) - 1
 for (i in 1:n_bins) {
   result[i] <- sum(x > lower_threshold & x <= thresholds[i]) / n_samples
   lower_threshold <- thresholds[i]
 }

 dim(result) <- c(bin = n_bins)
 result
}

bins <- multiApply::Apply(sample_data, 'time', binning, thresholds)$output1

# 3. Plotting most likely quantile/bin
PlotMostLikelyQuantileMap(bins, lons, lats,
                          toptitle = 'Most likely quantile map',
                          bar_titles = paste('% of belonging to', letters[1:n_bins]),
                          mask = 1 - (w1 + w2 / max(c(w1, w2))),
                          brks = 20, width = 10, height = 8)
```

RainFARM *RainFARM stochastic precipitation downscaling (reduced version)*

Description

This function implements the RainFARM stochastic precipitation downscaling method and accepts in input an array with named dims ("lon", "lat") and one or more dimension (such as "ftime", "sdate" or "time") over which to average automatically determined spectral slopes. Adapted for climate downscaling and including orographic correction. References: Terzago, S. et al. (2018). NHESS 18(11), 2825-2840. http://doi.org/10.5194/nhess-18-2825-2018, D'Onofrio et al. (2014), J of Hydrometeorology 15, 830-843; Rebora et. al. (2006), JHM 7, 724.

Usage

```
RainFARM(data, lon, lat, nf, weights = 1, nens = 1, slope = 0,
  kmin = 1, fglob = FALSE, fsmooth = TRUE, time_dim = NULL,
  lon_dim = "lon", lat_dim = "lat", drop_realization_dim = FALSE,
  verbose = FALSE)
```

Arguments

data	Precipitation array to downscale. The input array is expected to have at least two dimensions named "lon" and "lat" by default (these default names can be changed with the lon_dim and lat_dim parameters) and one or more dimensions over which to average these slopes, which can be specified by parameter time_dim. The number of longitudes and latitudes in the input data is expected to be even and the same. If not the function will perform a subsetting to ensure this condition.
lon	Vector or array of longitudes.
lat	Vector or array of latitudes.
nf	Refinement factor for downscaling (the output resolution is increased by this factor).
weights	Matrix with climatological weights which can be obtained using the CST_RFWeights function. If weights=1. (default) no weights are used. The matrix should have dimensions (lon, lat) in this order. The names of these dimensions are not checked.
nens	Number of ensemble members to produce (default: nens=1).
slope	Prescribed spectral slope. The default is slope=0. meaning that the slope is determined automatically over the dimensions specified by time_dim.
kmin	First wavenumber for spectral slope (default: kmin=1).
fglob	Logical to conseve global precipitation over the domain (default: FALSE)
fsmooth	Logical to conserve precipitation with a smoothing kernel (default: TRUE)
time_dim	String or character array with name(s) of time dimension(s) (e.g. "ftime", "sdate", "time" ...) over which to compute spectral slopes. If a character array of dimension names is provided, the spectral slopes will be computed over all elements belonging to those dimensions. If omitted one of c("ftime", "sdate", "time") is searched and the first one with more than one element is chosen.
lon_dim	Name of lon dimension ("lon" by default).
lat_dim	Name of lat dimension ("lat" by default).
drop_realization_dim	Logical to remove the "realization" stochastic ensemble dimension (default: FALSE) 1) if nens==1: the dimension is dropped; 2) if nens>1 and a "member" dimension exists: the "realization" and "member" dimensions are compacted (multiplied) and the resulting dimension is named "member"; 3) if nens>1 and a "member" dimension does not exist: the "realization" dimension is renamed to "member".
verbose	logical for verbose output (default: FALSE).

Value

RainFARM() returns a list containing the fine-scale longitudes, latitudes and the sequence of nens downscaled fields. If nens>1 an additional dimension named "realization" is added to the output array after the "member" dimension (if it exists and unless drop_realization_dim=TRUE is specified). The ordering of the remaining dimensions in the exp element of the input object is maintained.

Author(s)

Jost von Hardenberg - ISAC-CNR, <j.vonhardenberg@isac.cnr.it>

Examples

```
# Example for the 'reduced' RainFARM function
nf <- 8   # Choose a downscaling by factor 8
nens <- 3 # Number of ensemble members
# create a test array with dimension 8x8 and 20 timesteps
# or provide your own read from a netcdf file
pr <- rnorm(8 * 8 * 20)
dim(pr) <- c(lon = 8, lat = 8, ftime = 20)
lon_mat <- seq(10, 13.5, 0.5) # could also be a 2d matrix
lat_mat <- seq(40, 43.5, 0.5)
# Create a test array of weights
ww <- array(1., dim = c(8 * nf, 8 * nf))
# or create proper weights using an external fine-scale climatology file
#     Specify a weightsfn filename if you wish to save the weights
## Not run:
ww <- CST_RFWeights("./worldclim.nc", nf, lon = lon_mat, lat = lat_mat,
                    fsmooth = TRUE)

## End(Not run)
# downscale using weights (ww=1. means do not use weights)
res <- RainFARM(pr, lon_mat, lat_mat, nf,
                fsmooth = TRUE, fglob = FALSE,
                weights = ww, nens = 2, verbose = TRUE)
str(res)
#List of 3
# $ data: num [1:3, 1:20, 1:64, 1:64] 0.186 0.212 0.138 3.748 0.679 ...
# $ lon : num [1:64] 9.78 9.84 9.91 9.97 10.03 ...
# $ lat : num [1:64] 39.8 39.8 39.9 40 40 ...
dim(res$data)
#  lon         lat       ftime realization
#   64          64          20           2
```

RFSlope *RainFARM spectral slopes from an array (reduced version)*

Description

This function computes spatial spectral slopes from an array, to be used for RainFARM stochastic precipitation downscaling method.

Usage

```
RFSlope(data, kmin = 1, time_dim = NULL, lon_dim = "lon",
  lat_dim = "lat")
```

Arguments

data	Array containing the spatial precipitation fields to downscale. The input array is expected to have at least two dimensions named "lon" and "lat" by default (these default names can be changed with the lon_dim and lat_dim parameters) and one or more dimensions over which to average the slopes, which can be specified by parameter time_dim.
kmin	First wavenumber for spectral slope (default kmin=1).
time_dim	String or character array with name(s) of dimension(s) (e.g. "ftime", "sdate", "member" ...) over which to compute spectral slopes. If a character array of dimension names is provided, the spectral slopes will be computed as an average over all elements belonging to those dimensions. If omitted one of c("ftime", "sdate", "time") is searched and the first one with more than one element is chosen.
lon_dim	Name of lon dimension ("lon" by default).
lat_dim	Name of lat dimension ("lat" by default).

Value

RFSlope() returns spectral slopes using the RainFARM convention (the logarithmic slope of k*|A(k)|^2 where A(k) are the spectral amplitudes). The returned array has the same dimensions as the input array, minus the dimensions specified by lon_dim, lat_dim and time_dim.

Author(s)

Jost von Hardenberg - ISAC-CNR, <j.vonhardenberg@isac.cnr.it>

Examples

```
# Example for the 'reduced' RFSlope function
# Create a test array with dimension 8x8 and 20 timesteps,
# 3 starting dates and 20 ensemble members.
pr <- 1:(4*3*8*8*20)
dim(pr) <- c(ensemble = 4, sdate = 3, lon = 8, lat = 8, ftime = 20)

# Compute the spectral slopes ignoring the wavenumber
# corresponding to the largest scale (the box)
slopes <- RFSlope(pr, kmin=2)
dim(slopes)
#  ensemble    sdate
#         4        3
slopes
#         [,1]     [,2]     [,3]
#[1,] 1.893503 1.893503 1.893503
#[2,] 1.893503 1.893503 1.893503
#[3,] 1.893503 1.893503 1.893503
#[4,] 1.893503 1.893503 1.893503
```

Index