Package: CSIndicators (via r-universe)

October 21, 2024

Title Climate Services' Indicators Based on Sub-Seasonal to Decadal Predictions

Version 1.1.1

Description Set of generalised tools for the flexible computation of climate related indicators defined by the user. Each method represents a specific mathematical approach which is combined with the possibility to select an arbitrary time period to define the indicator. This enables a wide range of possibilities to tailor the most suitable indicator for each particular climate service application (agriculture, food security, energy, water management, ...). This package is intended for sub-seasonal, seasonal and decadal climate predictions, but its methods are also applicable to other time-scales, provided the dimensional structure of the input is maintained. Additionally, the outputs of the functions in this package are compatible with 'CSTools'. This package is described in 'Pérez-Zanón et al. (2023) <doi:10.1016/j.cliser.2023.100393>' and it was developed in the context of 'H2020 MED-GOLD' (776467) and 'S2S4E' (776787) projects. See 'Lledó et al. (2019) <doi:10.1016/j.renene.2019.04.135>' and 'Chou et al., 2023 <doi:10.1016/j.cliser.2023.100345>' for details.

Depends R (>= 3.6.0)

Imports multiApply (>= 2.1.1), stats, ClimProjDiags, CSTools, SPEI, lmom, lmomco, zoo, s2dv

Suggests testthat, knitr, markdown, rmarkdown

VignetteBuilder knitr

License GPL-3

URL https://earth.bsc.es/gitlab/es/csindicators/

BugReports https://earth.bsc.es/gitlab/es/csindicators/-/issues

Encoding UTF-8

RoxygenNote 7.2.3

Config/testthat/edition 3

NeedsCompilation no

Author Victòria Agudetse [cre], Eva Rifà [ctb], Nuria Perez-Zanon [aut] (<https://orcid.org/0000-0001-8568-3071>), Chou Chihchung [aut], Llorenç Lledó [aut], González-Reviriego Nube [ctb], Marcos Raül [ctb], Palma Lluis [ctb], An-Chi Ho [ctb], BSC-CNS [cph]

Maintainer Victòria Agudetse <victoria.agudetse@bsc.es>

Repository CRAN

Date/Publication 2024-01-24 14:43:01 UTC

Contents

`AbsToProbs` *Transform ensemble forecast into probabilities*

Description

The Cumulative Distribution Function of a forecast is used to obtain the probabilities of each value in the ensemble. If multiple initializations (start dates) are provided, the function will create the Cumulative Distribution Function excluding the corresponding initialization.

Usage

```
AbsToProbs(
  data,
  dates = NULL,
  start = NULL,
  end = NULL,
  time_dim = "time",
  memb_dim = "member",
  sdate_dim = "sdate",
  ncores = NULL
)
```

Arguments

`data`	A multidimensional array with named dimensions.
`dates`	An optional parameter containing a vector of dates or a multidimensional array of dates with named dimensions matching the dimensions on parameter 'data'. By default it is NULL, to select a period this parameter must be provided. All common dimensions with 'data' need to have the same length.
`start`	An optional parameter to define the initial date of the period to select from the data by providing a list of two elements: the initial date of the period and the initial month of the period. By default it is set to NULL and the indicator is computed using all the data provided in `data`.
`end`	An optional parameter to define the final date of the period to select from the data by providing a list of two elements: the final day of the period and the final month of the period. By default it is set to NULL and the indicator is computed using all the data provided in `data`.
`time_dim`	A character string indicating the name of the temporal dimension. By default, it is set to 'time'. More than one dimension name matching the dimensions provided in the object `data$data` can be specified. This dimension is required to subset the data in a requested period.

memb_dim	A character string indicating the name of the dimension in which the ensemble members are stored.
sdate_dim	A character string indicating the name of the dimension in which the initialization dates are stored.
ncores	An integer indicating the number of cores to use in parallel computation.

Value

A multidimensional array with named dimensions containing the probabilites in the element data.

Examples

```
exp <- array(rnorm(216), dim = c(dataset = 1, member = 2, sdate = 3,
                                 time = 9, lat = 2, lon = 2))
exp_probs <- AbsToProbs(exp)
data <- array(rnorm(5 * 3 * 61 * 1),
              c(member = 5, sdate = 3, time = 61, lon = 1))
Dates <- c(seq(as.Date("01-05-2000", format = "%d-%m-%Y"),
               as.Date("30-06-2000", format = "%d-%m-%Y"), by = 'day'),
           seq(as.Date("01-05-2001", format = "%d-%m-%Y"),
               as.Date("30-06-2001", format = "%d-%m-%Y"), by = 'day'),
           seq(as.Date("01-05-2002", format = "%d-%m-%Y"),
               as.Date("30-06-2002", format = "%d-%m-%Y"), by = 'day'))
dim(Dates) <- c(time = 61, sdate = 3)
exp_probs <- AbsToProbs(data, dates = Dates, start = list(21, 4),
                        end = list(21, 6))
```

AccumulationExceedingThreshold

Accumulation of a variable when Exceeding (not exceeding) a Threshold

---

Description

The accumulation (sum) of a variable in the days (or time steps) that the variable is exceeding (or not exceeding) a threshold during a period. The threshold provided must be in the same units than the variable units, i.e. to use a percentile as a scalar, the function Threshold or QThreshold may be needed. Providing mean daily temperature data, the following agriculture indices for heat stress can be obtained by using this function:

- 'GDD', Summation of daily differences between daily average temperatures and 10°C between April 1st and October 31st.

Usage

```
AccumulationExceedingThreshold(
  data,
  threshold,
  op = ">",
  diff = FALSE,
  dates = NULL,
  start = NULL,
  end = NULL,
  time_dim = "time",
  na.rm = FALSE,
  ncores = NULL
)
```

Arguments

data	A multidimensional array with named dimensions.
threshold	If only one threshold is used: it can be a multidimensional array with named dimensions. It must be in the same units and with the common dimensions of the same length as parameter 'data'. It can also be a vector with the same length of 'time_dim' from 'data' or a scalar. If we want to use two thresholds: it can be a vector of two scalars, a list of two vectors with the same length of 'time_dim' from 'data' or a list of two multidimensional arrays with the common dimensions of the same length as parameter 'data'. If two thresholds are used, parameter 'op' must be also a vector of two elements.
op	An operator '>' (by default), '<', '>=' or '<='. If two thresholds are used it has to be a vector of a pair of two logical operators: c('<', '>'), c('<', '>='), c('<=', '>'), c('<=', '>='), c('>', '<'), c('>', '<='), c('>=', '<'),c('>=', '<=')).
diff	A logical value indicating whether to accumulate the difference between data and threshold (TRUE) or not (FALSE by default). It can only be TRUE if a unique threshold is used.
dates	A multidimensional array of dates with named dimensions matching the temporal dimensions on parameter 'data'. By default it is NULL, to select aperiod this parameter must be provided.
start	An optional parameter to define the initial date of the period to select from the data by providing a list of two elements: the initial date of the period and the initial month of the period. By default it is set to NULL and the indicator is computed using all the data provided in data.
end	An optional parameter to define the final date of the period to select from the data by providing a list of two elements: the final day of the period and the final month of the period. By default it is set to NULL and the indicator is computed using all the data provided in data.
time_dim	A character string indicating the name of the dimension to compute the indicator. By default, it is set to 'time'. It can only indicate one time dimension.
na.rm	A logical value indicating whether to ignore NA values (TRUE) or not (FALSE).
ncores	An integer indicating the number of cores to use in parallel computation.

Value

A multidimensional array with named dimensions containing the aggregated values with dimensions of the input parameter 'data' except the dimension where the indicator has been computed.

Examples

```
# Assuming data is already (tasmax + tasmin)/2 - 10
data <- array(rnorm(5 * 3 * 214 * 2, mean = 25, sd = 3),
                  c(memb = 5, sdate = 3, time = 214, lon = 2))
GDD <- AccumulationExceedingThreshold(data, threshold = 0, start = list(1, 4),
                                      end = list(31, 10))
```

`CST_AbsToProbs` *Transform ensemble forecast into probabilities*

Description

The Cumulative Distribution Function of a forecast is used to obtain the probabilities of each value in the ensemble. If multiple initializations (start dates) are provided, the function will create the Cumulative Distribution Function excluding the corresponding initialization.

Usage

```
CST_AbsToProbs(
  data,
  start = NULL,
  end = NULL,
  time_dim = "time",
  memb_dim = "member",
  sdate_dim = "sdate",
  ncores = NULL
)
```

Arguments

`data`	An 's2dv_cube' object as provided function `CST_Start` or `CST_Load` in package CSTools.
`start`	An optional parameter to define the initial date of the period to select from the data by providing a list of two elements: the initial date of the period and the initial month of the period. By default it is set to NULL and the indicator is computed using all the data provided in `data`.
`end`	An optional parameter to define the final date of the period to select from the data by providing a list of two elements: the final day of the period and the final month of the period. By default it is set to NULL and the indicator is computed using all the data provided in `data`.

time_dim	A character string indicating the name of the temporal dimension. By default, it is set to 'time'. More than one dimension name matching the dimensions provided in the object data$data can be specified. This dimension is required to subset the data in a requested period.
memb_dim	A character string indicating the name of the dimension in which the ensemble members are stored.
sdate_dim	A character string indicating the name of the dimension in which the initialization dates are stored.
ncores	An integer indicating the number of cores to use in parallel computation.

Value

An 's2dv_cube' object containing the probabilites in the element data.

Examples

```
exp <- NULL
exp$data <- array(rnorm(216), dim = c(dataset = 1, member = 2, sdate = 3,
                 time = 9, lat = 2, lon = 2))
class(exp) <- 's2dv_cube'
exp_probs <- CST_AbsToProbs(exp)
exp$data <- array(rnorm(5 * 3 * 214 * 2),
                 c(member = 5, sdate = 3, time = 214, lon = 2))
exp$attrs$Dates <- c(seq(as.Date("01-05-2000", format = "%d-%m-%Y"),
                         as.Date("30-11-2000", format = "%d-%m-%Y"), by = 'day'),
                     seq(as.Date("01-05-2001", format = "%d-%m-%Y"),
                         as.Date("30-11-2001", format = "%d-%m-%Y"), by = 'day'),
                     seq(as.Date("01-05-2002", format = "%d-%m-%Y"),
                         as.Date("30-11-2002", format = "%d-%m-%Y"), by = 'day'))
dim(exp$attrs$Dates) <- c(time = 214, sdate = 3)
exp_probs <- CST_AbsToProbs(data = exp, start = list(21, 4), end = list(21, 6))
```

CST_AccumulationExceedingThreshold

Accumulation of a variable when Exceeding (not exceeding) a Threshold

---

Description

The accumulation (sum) of a variable in the days (or time steps) that the variable is exceeding (or not exceeding) a threshold during a period. The threshold provided must be in the same units than the variable units, i.e. to use a percentile as a scalar, the function Threshold or QThreshold may be needed. Providing mean daily temperature data, the following agriculture indices for heat stress can be obtained by using this function:

- 'GDD', Summation of daily differences between daily average temperatures and 10°C between April 1st and October 31st.

Usage

```
CST_AccumulationExceedingThreshold(
  data,
  threshold,
  op = ">",
  diff = FALSE,
  start = NULL,
  end = NULL,
  time_dim = "time",
  na.rm = FALSE,
  ncores = NULL
)
```

Arguments

data	An 's2dv_cube' object as provided function CST_Start or CST_Load in package CSTools.
threshold	If only one threshold is used, it can be an 's2dv_cube' object or a multidimensional array with named dimensions. It must be in the same units and with the common dimensions of the same length as parameter 'data'. It can also be a vector with the same length of 'time_dim' from 'data' or a scalar. If we want to use two thresholds: it can be a vector of two scalars, a list of two vectors with the same length of 'time_dim' from 'data' or a list of two multidimensional arrays with the common dimensions of the same length as parameter 'data'. If two thresholds are used, parameter 'op' must be also a vector of two elements.
op	An operator '>' (by default), '<', '>=' or '<='. If two thresholds are used it has to be a vector of a pair of two logical operators: c('<', '>'), c('<', '>='), c('<=', '>'), c('<=', '>='), c('>', '<'), c('>', '<='), c('>=', '<'),c('>=', '<=')).
diff	A logical value indicating whether to accumulate the difference between data and threshold (TRUE) or not (FALSE by default). It can only be TRUE if a unique threshold is used.
start	An optional parameter to define the initial date of the period to select from the data by providing a list of two elements: the initial date of the period and the initial month of the period. By default it is set to NULL and the indicator is computed using all the data provided in data.
end	An optional parameter to define the final date of the period to select from the data by providing a list of two elements: the final day of the period and the final month of the period. By default it is set to NULL and the indicator is computed using all the data provided in data.
time_dim	A character string indicating the name of the dimension to compute the indicator. By default, it is set to 'time'. It can only indicate one time dimension.
na.rm	A logical value indicating whether to ignore NA values (TRUE) or not (FALSE).
ncores	An integer indicating the number of cores to use in parallel computation.

Value

An 's2dv_cube' object containing the aggregated values in the element `data` with dimensions of the input parameter 'data' except the dimension where the indicator has been computed. The 'Dates' array is updated to the dates corresponding to the beginning of the aggregated time period. A new element called 'time_bounds' will be added into the 'attrs' element in the 's2dv_cube' object. It consists of a list containing two elements, the start and end dates of the aggregated period with the same dimensions of 'Dates' element.

Examples

```
exp <- NULL
exp$data <- array(abs(rnorm(5 * 3 * 214 * 2)*100),
                  c(memb = 5, sdate = 3, time = 214, lon = 2))
class(exp) <- 's2dv_cube'
Dates <- c(seq(as.Date("01-05-2000", format = "%d-%m-%Y"),
               as.Date("30-11-2000", format = "%d-%m-%Y"), by = 'day'),
           seq(as.Date("01-05-2001", format = "%d-%m-%Y"),
               as.Date("30-11-2001", format = "%d-%m-%Y"), by = 'day'),
           seq(as.Date("01-05-2002", format = "%d-%m-%Y"),
               as.Date("30-11-2002", format = "%d-%m-%Y"), by = 'day'))
dim(Dates) <- c(sdate = 3, time = 214)
exp$attrs$Dates <- Dates
AT <- CST_AccumulationExceedingThreshold(data = exp, threshold = 100,
                                         start = list(21, 4),
                                         end = list(21, 6))
```

`CST_MergeRefToExp` *Merge a Reference To Experiments*

Description

Some indicators are defined for specific temporal periods (e.g.: summer from June 21st to September 21st). If the initialization forecast date is later than the one required for the indicator (e.g.: July 1st), the user may want to merge past observations, or other references, to the forecast (or hindcast) to compute the indicator. If the forecast simulation doesn't cover the required period because it is initialized too early (e.g.: Initialization on November 1st the forecast covers until the beginning of June next year), a climatology (or other references) could be added at the end of the forecast lead time to cover the desired period (e.g.: until the end of summer).

Usage

```
CST_MergeRefToExp(
  data1,
  data2,
  start1 = NULL,
  end1 = NULL,
  start2 = NULL,
```

```
  end2 = NULL,
  time_dim = "time",
  memb_dim = "member",
  ncores = NULL
)
```

Arguments

data1	An 's2dv_cube' object with the element 'data' being a multidimensional array with named dimensions. All dimensions must be equal to 'data2' dimensions except for the ones specified with 'memb_dim' and 'time_dim'. If 'start1' and 'end1' are used to subset a period, the Dates must be stored in element '$attrs$Dates' of the object. Dates must have same time dimensions as element 'data'.
data2	An 's2dv_cube' object with the element 'data' being a multidimensional array of named dimensions matching the dimensions of parameter 'data1'. All dimensions must be equal to 'data1' except for the ones specified with 'memb_dim' and 'time_dim'. If 'start2' and 'end2' are used to subset a period, the Dates must be stored in element '$attrs$Dates' of the object. Dates must have same time dimensions as element 'data'.
start1	A list to define the initial date of the period to select from 'data1' by providing a list of two elements: the initial date of the period and the initial month of the period.
end1	A list to define the final date of the period to select from 'data1' by providing a list of two elements: the final day of the period and the final month of the period.
start2	A list to define the initial date of the period to select from 'data2' by providing a list of two elements: the initial date of the period and the initial month of the period.
end2	A list to define the final date of the period to select from 'data2' by providing a list of two elements: the final day of the period and the final month of the period.
time_dim	A character string indicating the name of the temporal dimension that will be used to combine the two arrays. By default, it is set to 'time'. Also, it will be used to subset the data in a requested period.
memb_dim	A character string indicating the name of the member dimension. If the data are not ensemble ones, set as NULL. The default value is 'member'.
ncores	An integer indicating the number of cores to use in parallel computation.

Details

This function is created to merge observations and forecasts, known as the 'blending' strategy (see references). The basis for this strategy is that the predictions are progressively replaced with observational data as soon as they become available (i.e., when entering the indicator definition period). This key strategy aims to increase users' confidence in the reformed predictions.

Value

An 's2dv_cube' object containing the indicator in the element data. The element data will be a multidimensional array created from the combination of 'data1' and 'data2'. The resulting array will contain the following dimensions: the original dimensions of the input data, which are common to both arrays and for the 'time_dim' dimension, the sum of the corresponding dimension of 'data1' and 'data2'. If 'memb_dim' is not null, regarding member dimension, two different situations can occur: (1) in the case that one of the arrays does not have member dimension or is equal to 1 and the other array has multiple member dimension, the result will contain the repeated values of the array one up to the lenght of member dimension of array two; (2) in the case that both arrays have member dimension and is greater than 1, all combinations of member dimension will be returned. The other elements of the 's2dv_cube' will be updated with the combined information of both datasets.

References

Chou, C., R. Marcos-Matamoros, L. Palma Garcia, N. Pérez-Zanón, M. Teixeira, S. Silva, N. Fontes, A. Graça, A. Dell'Aquila, S. Calmanti and N. González-Reviriego (2023). Advanced seasonal predictions for vine management based on bioclimatic indicators tailored to the wine sector. Climate Services, 30, 100343, doi: 10.1016/j.cliser.2023.100343.

Examples

```
data_dates <- c(seq(as.Date("01-07-1993", "%d-%m-%Y", tz = 'UTC'),
                    as.Date("01-12-1993","%d-%m-%Y", tz = 'UTC'), "day"),
                seq(as.Date("01-07-1994", "%d-%m-%Y", tz = 'UTC'),
                    as.Date("01-12-1994","%d-%m-%Y", tz = 'UTC'), "day"))
dim(data_dates) <- c(time = 154, sdate = 2)
data <- NULL
data$data <- array(1:(2*154*2), c(time = 154, sdate = 2, member = 2))
data$attrs$Dates<- data_dates
class(data) <- 's2dv_cube'
ref_dates <- seq(as.Date("01-01-1993", "%d-%m-%Y", tz = 'UTC'),
                 as.Date("01-12-1994","%d-%m-%Y", tz = 'UTC'), "day")
dim(ref_dates) <- c(time = 350, sdate = 2)
ref <- NULL
ref$data <- array(1001:1700, c(time = 350, sdate = 2))
ref$attrs$Dates <- ref_dates
class(ref) <- 's2dv_cube'
new_data <- CST_MergeRefToExp(data1 = ref, data2 = data,
                              start1 = list(21, 6), end1 = list(30, 6),
                              start2 = list(1, 7), end2 = list(21, 9))
```

CST_PeriodAccumulation

Period Accumulation on 's2dv_cube' objects

Description

Period Accumulation computes the sum (accumulation) of a given variable in a period. Providing precipitation data, two agriculture indices can be obtained by using this function:

- 'SprR', Spring Total Precipitation: The total precipitation from April 21th to June 21st.
- 'HarR', Harvest Total Precipitation: The total precipitation from August 21st to October 21st.

Usage

```
CST_PeriodAccumulation(
  data,
  start = NULL,
  end = NULL,
  time_dim = "time",
  rollwidth = NULL,
  sdate_dim = "sdate",
  frequency = "monthly",
  na.rm = FALSE,
  ncores = NULL
)
```

Arguments

`data`	An 's2dv_cube' object as provided function `CST_Start` or `CST_Load` in package CSTools.
`start`	An optional parameter to defined the initial date of the period to select from the data by providing a list of two elements: the initial date of the period and the initial m onth of the period. By default it is set to NULL and the indicator is computed using all the data provided in `data`.
`end`	An optional parameter to defined the final date of the period to select from the data by providing a list of two elements: the final day of the period and the final month of the period. By default it is set to NULL and the indicator is computed using all the data provided in `data`.
`time_dim`	A character string indicating the name of the dimension to compute the indicator. By default, it is set to 'time'. More than one dimension name matching the dimensions provided in the object `data$data` can be specified.
`rollwidth`	An optional parameter to indicate the number of time steps the rolling sum is applied to. If it is positive, the rolling sum is applied backwards 'time_dim', if it is negative, it will be forward it. When this parameter is NULL, the sum is applied over all 'time_dim', in a specified period. It is NULL by default.
`sdate_dim`	(Only needed when rollwidth is used). A character string indicating the name of the start date dimension to compute the rolling accumulation. By default, it is set to 'sdate'.
`frequency`	(Only needed when rollwidth is used). A character string indicating the time frequency of the data to apply the rolling accumulation. It can be 'daily' or 'monthly'. If it is set to 'monthly', values from continuous months will be accumulated; if it is 'daliy', values from continuous days will be accumulated. It is set to 'monthly' by default.

na.rm	A logical value indicating whether to ignore NA values (TRUE) or not (FALSE).
ncores	An integer indicating the number of cores to use in parallel computation.

Details

There are two possible ways of performing the accumulation. The default one is by accumulating a variable over a dimension specified with 'time_dim'. To chose a specific time period, 'start' and 'end' must be used. The other method is by using 'rollwidth' parameter. When this parameter is a positive integer, the cumulative backward sum is applied to the time dimension. If it is negative, the rolling sum is applied backwards. This function is build to be compatible with other tools in that work with 's2dv_cube' object class. The input data must be this object class. If you don't work with 's2dv_cube', see PeriodAccumulation.

Value

An 's2dv_cube' object containing the accumulated data in the element data. If parameter 'rollwidth' is not used, it will have the dimensions of the input parameter 'data' except the dimension where the accumulation has been computed (specified with 'time_dim'). If 'rollwidth' is used, it will be of same dimensions as input data. The 'Dates' array is updated to the dates corresponding to the beginning of the aggregated time period. A new element called 'time_bounds' will be added into the 'attrs' element in the 's2dv_cube' object. It consists of a list containing two elements, the start and end dates of the aggregated period with the same dimensions of 'Dates' element. If 'rollwidth' is used, it will contain the same dimensions of parameter 'data' and the other elements of the 's2dv_cube' will not be modified.

Examples

```
exp <- NULL
exp$data <- array(rnorm(216)*200, dim = c(dataset = 1, member = 2, sdate = 3,
                 ftime = 9, lat = 2, lon = 2))
class(exp) <- 's2dv_cube'
TP <- CST_PeriodAccumulation(exp, time_dim = 'ftime')
exp$data <- array(rnorm(5 * 3 * 214 * 2),
                 c(memb = 5, sdate = 3, ftime = 214, lon = 2))
Dates <- c(seq(as.Date("01-05-2000", format = "%d-%m-%Y"),
              as.Date("30-11-2000", format = "%d-%m-%Y"), by = 'day'),
           seq(as.Date("01-05-2001", format = "%d-%m-%Y"),
              as.Date("30-11-2001", format = "%d-%m-%Y"), by = 'day'),
           seq(as.Date("01-05-2002", format = "%d-%m-%Y"),
              as.Date("30-11-2002", format = "%d-%m-%Y"), by = 'day'))
dim(Dates) <- c(sdate = 3, ftime = 214)
exp$attrs$Dates <- Dates
SprR <- CST_PeriodAccumulation(exp, start = list(21, 4), end = list(21, 6),
                               time_dim = 'ftime')
dim(SprR$data)
head(SprR$attrs$Dates)
HarR <- CST_PeriodAccumulation(exp, start = list(21, 8), end = list(21, 10),
                               time_dim = 'ftime')
dim(HarR$data)
head(HarR$attrs$Dates)
```

CST_PeriodMax *Period Max on 's2dv_cube' objects*

Description

Period Max computes the maximum (max) of a given variable in a period. Two bioclimatic indicators can be obtained by using this function:

- 'BIO5', (Providing temperature data) Max Temperature of Warmest Month. The maximum monthly temperature occurrence over a given year (time-series) or averaged span of years (normal).
- 'BIO13', (Providing precipitation data) Precipitation of Wettest Month. This index identifies the total precipitation that prevails during the wettest month.

Usage

```
CST_PeriodMax(
  data,
  start = NULL,
  end = NULL,
  time_dim = "time",
  na.rm = FALSE,
  ncores = NULL
)
```

Arguments

data	An 's2dv_cube' object as provided function CST_Start or CST_Load in package CSTools.
start	An optional parameter to defined the initial date of the period to select from the data by providing a list of two elements: the initial date of the period and the initial month of the period. By default it is set to NULL and the indicator is computed using all the data provided in data.
end	An optional parameter to defined the final date of the period to select from the data by providing a list of two elements: the final day of the period and the final month of the period. By default it is set to NULL and the indicator is computed using all the data provided in data.
time_dim	A character string indicating the name of the dimension to compute the indicator. By default, it is set to 'time'. More than one dimension name matching the dimensions provided in the object data$data can be specified.
na.rm	A logical value indicating whether to ignore NA values (TRUE) or not (FALSE).
ncores	An integer indicating the number of cores to use in parallel computation.

Value

An 's2dv_cube' object containing the indicator in the element `data` with dimensions of the input parameter 'data' except the dimension where the max has been computed (specified with 'time_dim'). A new element called 'time_bounds' will be added into the 'attrs' element in the 's2dv_cube' object. It consists of a list containing two elements, the start and end dates of the aggregated period with the same dimensions of 'Dates' element.

Examples

```
exp <- NULL
exp$data <- array(rnorm(45), dim = c(member = 7, sdate = 4, time = 3))
Dates <- c(seq(as.Date("2000-11-01", "%Y-%m-%d", tz = "UTC"),
               as.Date("2001-01-01", "%Y-%m-%d", tz = "UTC"), by = "month"),
           seq(as.Date("2001-11-01", "%Y-%m-%d", tz = "UTC"),
               as.Date("2002-01-01", "%Y-%m-%d", tz = "UTC"), by = "month"),
           seq(as.Date("2002-11-01", "%Y-%m-%d", tz = "UTC"),
               as.Date("2003-01-01", "%Y-%m-%d", tz = "UTC"), by = "month"),
           seq(as.Date("2003-11-01", "%Y-%m-%d", tz = "UTC"),
               as.Date("2004-01-01", "%Y-%m-%d", tz = "UTC"), by = "month"))
dim(Dates) <- c(sdate = 4, time = 3)
exp$attrs$Dates <- Dates
class(exp) <- 's2dv_cube'

res <- CST_PeriodMax(exp, start = list(01, 12), end = list(01, 01))
```

`CST_PeriodMean` *Period Mean on 's2dv_cube' objects*

Description

Period Mean computes the average (mean) of a given variable in a period. Providing temperature data, two agriculture indices can be obtained by using this function:

- 'GST', Growing Season average Temperature: The average temperature from April 1st to Octobe 31st.
- 'SprTX', Spring Average Maximum Temperature: The average daily maximum temperature from April 1st to May 31st.

Usage

```
CST_PeriodMean(
  data,
  start = NULL,
  end = NULL,
  time_dim = "time",
  na.rm = FALSE,
  ncores = NULL
)
```

Arguments

data	An 's2dv_cube' object as provided function CST_Start or CST_Load in package CSTools.
start	An optional parameter to defined the initial date of the period to select from the data by providing a list of two elements: the initial date of the period and the initial month of the period. By default it is set to NULL and the indicator is computed using all the data provided in data.
end	An optional parameter to defined the final date of the period to select from the data by providing a list of two elements: the final day of the period and the final month of the period. By default it is set to NULL and the indicator is computed using all the data provided in data.
time_dim	A character string indicating the name of the dimension to compute the indicator. By default, it is set to 'time'. More than one dimension name matching the dimensions provided in the object data$data can be specified.
na.rm	A logical value indicating whether to ignore NA values (TRUE) or not (FALSE).
ncores	An integer indicating the number of cores to use in parallel computation.

Value

An 's2dv_cube' object containing the indicator in the element data with dimensions of the input parameter 'data' except the dimension where the mean has been computed (specified with 'time_dim'). The 'Dates' array is updated to the dates corresponding to the beginning of the aggregated time period. A new element called 'time_bounds' will be added into the 'attrs' element in the 's2dv_cube' object. It consists of a list containing two elements, the start and end dates of the aggregated period with the same dimensions of 'Dates' element.

Examples

```
exp <- NULL
exp$data <- array(rnorm(45), dim = c(member = 7, sdate = 4, time = 3))
Dates <- c(seq(as.Date("2000-11-01", "%Y-%m-%d", tz = "UTC"),
               as.Date("2001-01-01", "%Y-%m-%d", tz = "UTC"), by = "month"),
           seq(as.Date("2001-11-01", "%Y-%m-%d", tz = "UTC"),
               as.Date("2002-01-01", "%Y-%m-%d", tz = "UTC"), by = "month"),
           seq(as.Date("2002-11-01", "%Y-%m-%d", tz = "UTC"),
               as.Date("2003-01-01", "%Y-%m-%d", tz = "UTC"), by = "month"),
           seq(as.Date("2003-11-01", "%Y-%m-%d", tz = "UTC"),
               as.Date("2004-01-01", "%Y-%m-%d", tz = "UTC"), by = "month"))
dim(Dates) <- c(sdate = 4, time = 3)
exp$attrs$Dates <- Dates
class(exp) <- 's2dv_cube'

SA <- CST_PeriodMean(exp, start = list(01, 12), end = list(01, 01))
```

CST_PeriodMin *Period Min on 's2dv_cube' objects*

Description

Period Min computes the average (min) of a given variable in a period. Two bioclimatic indicators can be obtained by using this function:

- 'BIO6', (Providing temperature data) Min Temperature of Coldest Month. The minimum monthly temperature occurrence over a given year (time-series) or averaged span of years (normal).
- 'BIO14', (Providing precipitation data) Precipitation of Driest Month. This index identifies the total precipitation that prevails during the driest month.

Usage

```
CST_PeriodMin(
  data,
  start = NULL,
  end = NULL,
  time_dim = "time",
  na.rm = FALSE,
  ncores = NULL
)
```

Arguments

data	An 's2dv_cube' object as provided function CST_Start or CST_Load in package CSTools.
start	An optional parameter to defined the initial date of the period to select from the data by providing a list of two elements: the initial date of the period and the initial month of the period. By default it is set to NULL and the indicator is computed using all the data provided in data.
end	An optional parameter to defined the final date of the period to select from the data by providing a list of two elements: the final day of the period and the final month of the period. By default it is set to NULL and the indicator is computed using all the data provided in data.
time_dim	A character string indicating the name of the dimension to compute the indicator. By default, it is set to 'time'. More than one dimension name matching the dimensions provided in the object data$data can be specified.
na.rm	A logical value indicating whether to ignore NA values (TRUE) or not (FALSE).
ncores	An integer indicating the number of cores to use in parallel computation.

Value

An 's2dv_cube' object containing the indicator in the element `data` with dimensions of the input parameter 'data' except the dimension where the min has been computed (specified with 'time_dim'). A new element called 'time_bounds' will be added into the 'attrs' element in the 's2dv_cube' object. It consists of a list containing two elements, the start and end dates of the aggregated period with the same dimensions of 'Dates' element.

Examples

```
exp <- NULL
exp$data <- array(rnorm(45), dim = c(member = 7, sdate = 4, time = 3))
Dates <- c(seq(as.Date("2000-11-01", "%Y-%m-%d", tz = "UTC"),
               as.Date("2001-01-01", "%Y-%m-%d", tz = "UTC"), by = "month"),
           seq(as.Date("2001-11-01", "%Y-%m-%d", tz = "UTC"),
               as.Date("2002-01-01", "%Y-%m-%d", tz = "UTC"), by = "month"),
           seq(as.Date("2002-11-01", "%Y-%m-%d", tz = "UTC"),
               as.Date("2003-01-01", "%Y-%m-%d", tz = "UTC"), by = "month"),
           seq(as.Date("2003-11-01", "%Y-%m-%d", tz = "UTC"),
               as.Date("2004-01-01", "%Y-%m-%d", tz = "UTC"), by = "month"))
dim(Dates) <- c(sdate = 4, time = 3)
exp$attrs$Dates <- Dates
class(exp) <- 's2dv_cube'

res <- CST_PeriodMin(exp, start = list(01, 12), end = list(01, 01))
```

CST_PeriodPET *Compute the Potential Evapotranspiration*

Description

Compute the Potential evapotranspiration (PET) that is the amount of evaporation and transpiration that would occur if a sufficient water source were available. This function calculate PET according to the Thornthwaite, Hargreaves or Hargreaves-modified equations.

Usage

```
CST_PeriodPET(
  data,
  pet_method = "hargreaves",
  time_dim = "syear",
  leadtime_dim = "time",
  lat_dim = "latitude",
  na.rm = FALSE,
  ncores = NULL
)
```

Arguments

data	A named list with the needed s2dv_cube objects containing the seasonal forecast experiment in the 'data' element for each variable. Specific variables are needed for each method used in computing the Potential Evapotranspiration (see parameter 'pet_method'). The accepted variable names are fixed in order to be recognized by the function. The accepted name corresponding to the Minimum Temperature is 'tmin', for Maximum Temperature is 'tmax', for Mean Temperature is 'tmean' and for Precipitation is 'pr'. The accepted variable names for each method are: For 'hargreaves': 'tmin' and 'tmax'; for 'hargreaves_modified' are 'tmin', 'tmax' and 'pr'; for method 'thornthwaite' 'tmean' is required. The units for temperature variables ('tmin', 'tmax' and 'tmean') need to be in Celcius degrees; the units for precipitation ('pr') need to be in mm/month. Currently the function works only with monthly data from different years.
pet_method	A character string indicating the method used to compute the potential evapotranspiration. The accepted methods are: 'hargreaves' and 'hargreaves_modified', that require the data to have variables tmin and tmax; and 'thornthwaite', that requires variable 'tmean'.
time_dim	A character string indicating the name of the temporal dimension. By default, it is set to 'syear'.
leadtime_dim	A character string indicating the name of the temporal dimension. By default, it is set to 'time'.
lat_dim	A character string indicating the name of the latitudinal dimension. By default it is set by 'latitude'.
na.rm	A logical value indicating whether NA values should be removed from data. It is FALSE by default.
ncores	An integer value indicating the number of cores to use in parallel computation.

Details

This function is build to be compatible with other tools in that work with 's2dv_cube' object class. The input data must be this object class. If you don't work with 's2dv_cube', see PeriodPET. For more information on the SPEI calculation, see functions CST_PeriodStandardization and CST_PeriodAccumulation.

Examples

```
dims <- c(time = 3, syear = 3, ensemble = 1, latitude = 1)
exp_tasmax <- array(rnorm(360, 27.73, 5.26), dim = dims)
exp_tasmin <- array(rnorm(360, 14.83, 3.86), dim = dims)
exp_prlr <- array(rnorm(360, 21.19, 25.64), dim = dims)
end_year <- 2012
dates_exp <- as.POSIXct(c(paste0(2010:end_year, "-08-16"),
                          paste0(2010:end_year, "-09-15"),
                          paste0(2010:end_year, "-10-16")), "UTC")
dim(dates_exp) <- c(syear = 3, time = 3)
lat <- c(40)
```

```
exp1 <- list('tmax' = exp_tasmax, 'tmin' = exp_tasmin, 'pr' = exp_prlr)
res <- PeriodPET(data = exp1, lat = lat, dates = dates_exp)
```

CST_PeriodStandardization *Compute the Standardization of Precipitation-Evapotranspiration Index*

Description

The Standardization of the data is the last step of computing the SPEI (Standarized Precipitation-Evapotranspiration Index). With this function the data is fit to a probability distribution to transform the original values to standardized units that are comparable in space and time and at different SPEI time scales.

Usage

```
CST_PeriodStandardization(
  data,
  data_cor = NULL,
  time_dim = "syear",
  leadtime_dim = "time",
  memb_dim = "ensemble",
  ref_period = NULL,
  handle_infinity = FALSE,
  method = "parametric",
  distribution = "log-Logistic",
  params = NULL,
  return_params = FALSE,
  na.rm = FALSE,
  ncores = NULL
)
```

Arguments

data	An 's2dv_cube' that element 'data' stores a multidimensional array containing the data to be standardized.
data_cor	An 's2dv_cube' that element 'data' stores a multidimensional array containing the data in which the standardization should be applied using the fitting parameters from 'data'.
time_dim	A character string indicating the name of the temporal dimension. By default, it is set to 'syear'.
leadtime_dim	A character string indicating the name of the temporal dimension. By default, it is set to 'time'.

`memb_dim` A character string indicating the name of the dimension in which the ensemble members are stored. When set it to NULL, threshold is computed for individual members.

`ref_period` A list with two numeric values with the starting and end points of the reference period used for computing the index. The default value is NULL indicating that the first and end values in data will be used as starting and end points.

`handle_infinity` A logical value wether to return infinite values (TRUE) or not (FALSE). When it is TRUE, the positive infinite values (negative infinite) are substituted by the maximum (minimum) values of each computation step, a subset of the array of dimensions time_dim, leadtime_dim and memb_dim.

`method` A character string indicating the standardization method used. If can be: 'parametric' or 'non-parametric'. It is set to 'parametric' by default.

`distribution` A character string indicating the name of the distribution function to be used for computing the SPEI. The accepted names are: 'log-Logistic' and 'Gamma'. It is set to 'log-Logistic' by default. The 'Gamma' method only works when only precipitation is provided and other variables are 0 because it is positive defined (SPI indicator).

`params` An optional parameter that needs to be a multidimensional array with named dimensions. This option overrides computation of fitting parameters. It needs to be of same time dimensions (specified in 'time_dim' and 'leadtime_dim') of 'data' and a dimension named 'coef' with the length of the coefficients needed for the used distribution (for 'Gamma' coef dimension is of lenght 2, for 'log-Logistic' is of length 3). It also needs to have a leadtime dimension (specified in 'leadtime_dim') of length 1. It will only be used if 'data_cor' is not provided.

`return_params` A logical value indicating wether to return parameters array (TRUE) or not (FALSE). It is FALSE by default.

`na.rm` A logical value indicating whether NA values should be removed from data. It is FALSE by default. If it is FALSE and there are NA values, standardization cannot be carried out for those coordinates and therefore, the result will be filled with NA for the specific coordinates. If it is TRUE, if the data from other dimensions except time_dim and leadtime_dim is not reaching 4 values, it is not enough values to estimate the parameters and the result will include NA.

`ncores` An integer value indicating the number of cores to use in parallel computation.

Details

Next, some specifications for the calculation of the standardization will be discussed. If there are NAs in the data and they are not removed with the parameter 'na.rm', the standardization cannot be carried out for those coordinates and therefore, the result will be filled with NA for the specific coordinates. When NAs are not removed, if the length of the data for a computational step is smaller than 4, there will not be enough data for standarize and the result will be also filled with NAs for that coordinates. About the distribution used to fit the data, there are only two possibilities: 'log-logistic' and 'Gamma'. The 'Gamma' method only works when only precipitation is provided and other variables are 0 because it is positive defined (SPI indicator). When only 'data' is provided ('data_cor' is NULL) the standardization is computed with cross validation. This function is build

to be compatible with other tools in that work with 's2dv_cube' object class. The input data must be this object class. If you don't work with 's2dv_cube', see PeriodStandardization. For more information on the SPEI indicator calculation, see CST_PeriodPET and CST_PeriodAccumulation.

Value

An object of class `s2dv_cube` containing the standardized data. If 'data_cor' is provided the array stored in element data will be of the same dimensions as 'data_cor'. If 'data_cor' is not provided, the array stored in element data will be of the same dimensions as 'data'. The parameters of the standardization will only be returned if 'return_params' is TRUE, in this case, the output will be a list of two objects one for the standardized data and one for the parameters.

Examples

```
dims <-  c(syear = 6, time = 3, latitude = 2, ensemble = 25)
data <- NULL
data$data <- array(rnorm(600, -204.1, 78.1), dim = dims)
class(data) <- 's2dv_cube'
SPEI <- CST_PeriodStandardization(data = data)
```

CST_PeriodVariance *Period Variance on 's2dv_cube' objects*

Description

Period Variance computes the average (var) of a given variable in a period. Two bioclimatic indicators can be obtained by using this function:

- 'BIO4', (Providing temperature data) Temperature Seasonality (Standard Deviation). The amount of temperature variation over a given year (or averaged years) based on the standard deviation (variation) of monthly temperature averages.
- 'BIO15', (Providing precipitation data) Precipitation Seasonality (CV). This is a measure of the variation in monthly precipitation totals over the course of the year. This index is the ratio of the standard deviation of the monthly total precipitation to the mean monthly total precipitation (also known as the coefficient of variation) and is expressed as a percentage.

Usage

```
CST_PeriodVariance(
  data,
  start = NULL,
  end = NULL,
  time_dim = "time",
  na.rm = FALSE,
  ncores = NULL
)
```

Arguments

data	An 's2dv_cube' object as provided function CST_Start or CST_Load in package CSTools.
start	An optional parameter to defined the initial date of the period to select from the data by providing a list of two elements: the initial date of the period and the initial month of the period. By default it is set to NULL and the indicator is computed using all the data provided in data.
end	An optional parameter to defined the final date of the period to select from the data by providing a list of two elements: the final day of the period and the final month of the period. By default it is set to NULL and the indicator is computed using all the data provided in data.
time_dim	A character string indicating the name of the dimension to compute the indicator. By default, it is set to 'time'. More than one dimension name matching the dimensions provided in the object data$data can be specified.
na.rm	A logical value indicating whether to ignore NA values (TRUE) or not (FALSE).
ncores	An integer indicating the number of cores to use in parallel computation.

Value

An 's2dv_cube' object containing the indicator in the element data with dimensions of the input parameter 'data' except the dimension where the var has been computed (specified with 'time_dim'). A new element called 'time_bounds' will be added into the 'attrs' element in the 's2dv_cube' object. It consists of a list containing two elements, the start and end dates of the aggregated period with the same dimensions of 'Dates' element.

Examples

```
exp <- NULL
exp$data <- array(rnorm(45), dim = c(member = 7, sdate = 4, time = 3))
Dates <- c(seq(as.Date("2000-11-01", "%Y-%m-%d", tz = "UTC"),
               as.Date("2001-01-01", "%Y-%m-%d", tz = "UTC"), by = "month"),
           seq(as.Date("2001-11-01", "%Y-%m-%d", tz = "UTC"),
               as.Date("2002-01-01", "%Y-%m-%d", tz = "UTC"), by = "month"),
           seq(as.Date("2002-11-01", "%Y-%m-%d", tz = "UTC"),
               as.Date("2003-01-01", "%Y-%m-%d", tz = "UTC"), by = "month"),
           seq(as.Date("2003-11-01", "%Y-%m-%d", tz = "UTC"),
               as.Date("2004-01-01", "%Y-%m-%d", tz = "UTC"), by = "month"))
dim(Dates) <- c(sdate = 4, time = 3)
exp$attrs$Dates <- Dates
class(exp) <- 's2dv_cube'

res <- CST_PeriodVariance(exp, start = list(01, 12), end = list(01, 01))
```

CST_QThreshold *Transform an absolute threshold into probabilities*

Description

From the user's perspective, an absolute threshold can be very useful for a specific needs (e.g.: grape variety). However, this absolute threshold could be transformed to a relative threshold in order to get its frequency in a given dataset. Therefore, the function `QThreshold` returns the probability of an absolute threshold. This is done by computing the Cumulative Distribution Function of a sample and leaving one out. The sample used will depend on the dimensions of the data provided and the dimension names provided in sdate_dim and memb_dim parameters:

Usage

```
CST_QThreshold(
  data,
  threshold,
  start = NULL,
  end = NULL,
  time_dim = "time",
  memb_dim = "member",
  sdate_dim = "sdate",
  ncores = NULL
)
```

Arguments

`data`	An 's2dv_cube' object as provided function `CST_Start` or `CST_Load` in package CSTools.
`threshold`	An 's2dv_cube' object as output of a 'CST_' function in the same units as parameter 'data' and with the common dimensions of the element 'data' of the same length. A single scalar is also possible.
`start`	An optional parameter to defined the initial date of the period to select from the data by providing a list of two elements: the initial date of the period and the initial month of the period. By default it is set to NULL and the indicator is computed using all the data provided in `data`.
`end`	An optional parameter to defined the final date of the period to select from the data by providing a list of two elements: the final day of the period and the final month of the period. By default it is set to NULL and the indicator is computed using all the data provided in `data`.
`time_dim`	A character string indicating the name of the temporal dimension. By default, it is set to 'time'. More than one dimension name matching the dimensions provided in the object `data$data` can be specified. This dimension is required to subset the data in a requested period.
`memb_dim`	A character string indicating the name of the dimension in which the ensemble members are stored.

sdate_dim	A character string indicating the name of the dimension in which the initialization dates are stored.
ncores	An integer indicating the number of cores to use in parallel computation.

Details

- If a forecast (hindcast) has dimensions member and start date, and both must be used in the sample, their names should be passed in sdate_dim and memb_dim.
- If a forecast (hindcast) has dimensions member and start date, and only start date must be used in the sample (the calculation is done in each separate member), memb_dim can be set to NULL.
- If a reference (observations) has start date dimension, the sample used is the start date dimension.
- If a reference (observations) doesn't have start date dimension, the sample used must be especified in sdate_dim parameter.

Value

An 's2dv_cube' object containing the probability of an absolute threshold in the element data.

Examples

```
threshold <- 26
exp <- NULL
exp$data <- array(abs(rnorm(112)*26), dim = c(member = 7, sdate = 8, time = 2))
class(exp) <- 's2dv_cube'
exp_probs <- CST_QThreshold(exp, threshold)

exp$data <- array(abs(rnorm(5 * 3 * 214 * 2)*50),
                  c(member = 5, sdate = 3, time = 214, lon = 2))
exp$attrs$Dates <- c(seq(as.Date("01-05-2000", format = "%d-%m-%Y"),
                         as.Date("30-11-2000", format = "%d-%m-%Y"), by = 'day'),
                     seq(as.Date("01-05-2001", format = "%d-%m-%Y"),
                         as.Date("30-11-2001", format = "%d-%m-%Y"), by = 'day'),
                     seq(as.Date("01-05-2002", format = "%d-%m-%Y"),
                         as.Date("30-11-2002", format = "%d-%m-%Y"), by = 'day'))
dim(exp$attrs$Dates) <- c(sdate = 3, time = 214)
class(exp) <- 's2dv_cube'
exp_probs <- CST_QThreshold(exp, threshold, start = list(21, 4),
                            end = list(21, 6))
```

CST_SelectPeriodOnData

Select a period on Data on 's2dv_cube' objects

Description

Auxiliary function to subset data for a specific period.

Usage

```
CST_SelectPeriodOnData(data, start, end, time_dim = "time", ncores = NULL)
```

Arguments

data	An 's2dv_cube' object as provided function CST_Start or CST_Load in package CSTools.
start	A parameter to defined the initial date of the period to select from the data by providing a list of two elements: the initial date of the period and the initial month of the period.
end	A parameter to defined the final date of the period to select from the data by providing a list of two elements: the final day of the period and the final month of the period.
time_dim	A character string indicating the name of the dimension to compute select the dates. By default, it is set to 'time'. More than one dimension name matching the dimensions provided in the object data$data can be specified.
ncores	An integer indicating the number of cores to use in parallel computation.

Value

A 's2dv_cube' object containing the subset of the object data$data during the period requested from start to end.

Examples

```
exp <- NULL
exp$data <- array(rnorm(5 * 3 * 214 * 2),
                  c(memb = 5, sdate = 3, time = 214, lon = 2))
exp$attrs$Dates <- c(seq(as.Date("01-05-2000", format = "%d-%m-%Y"),
                         as.Date("30-11-2000", format = "%d-%m-%Y"), by = 'day'),
                     seq(as.Date("01-05-2001", format = "%d-%m-%Y"),
                         as.Date("30-11-2001", format = "%d-%m-%Y"), by = 'day'),
                     seq(as.Date("01-05-2002", format = "%d-%m-%Y"),
                         as.Date("30-11-2002", format = "%d-%m-%Y"), by = 'day'))
dim(exp$attrs$Dates) <- c(time = 214, sdate = 3)
class(exp) <- 's2dv_cube'
Period <- CST_SelectPeriodOnData(exp, start = list(21, 6), end = list(21, 9))
```

CST_Threshold *Absolute value of a relative threshold (percentile)*

Description

Frequently, thresholds are defined by a percentile that may correspond to a different absolute value depending on the variable, gridpoint and also julian day (time). This function calculates the corresponding value of a percentile given a dataset.

Usage

```
CST_Threshold(
  data,
  threshold,
  start = NULL,
  end = NULL,
  time_dim = "time",
  memb_dim = "member",
  sdate_dim = "sdate",
  na.rm = FALSE,
  ncores = NULL
)
```

Arguments

data	An 's2dv_cube' object as provided function CST_Start or CST_Load in package CSTools.
threshold	A single scalar or vector indicating the relative threshold(s). It must contain values between 0 and 1.
start	An optional parameter to defined the initial date of the period to selectfrom the data by providing a list of two elements: the initial date of the period and the initial month of the period. By default it is set to NULL and the indicator is computed using all the data provided in data.
end	An optional parameter to defined the final date of the period to select from the data by providing a list of two elements: the final day of the period and the final month of the period. By default it is set to NULL and the indicator is computed using all the data provided in data.
time_dim	A character string indicating the name of the temporal dimension. By default, it is set to 'time'. More than one dimension name matching the dimensions provided in the object data$data can be specified. This dimension is required to subset the data in a requested period.
memb_dim	A character string indicating the name of the dimension in which the ensemble members are stored. When set it to NULL, threshold is computed for individual members.
sdate_dim	A character string indicating the name of the dimension in which the initialization dates are stored.

na.rm	A logical value indicating whether to ignore NA values (TRUE) or not (FALSE).
ncores	An integer indicating the number of cores to use in parallel computation.

Value

An 's2dv_cube' object containing the corresponding values of a percentile in the element data.

Examples

```
threshold <- 0.9
exp <- NULL
exp$data <- array(rnorm(5 * 3 * 214 * 2),
                  c(member = 5, sdate = 3, time = 214, lon = 2))
exp$attrs$Dates <- c(seq(as.Date("01-05-2000", format = "%d-%m-%Y"),
                         as.Date("30-11-2000", format = "%d-%m-%Y"), by = 'day'),
                     seq(as.Date("01-05-2001", format = "%d-%m-%Y"),
                         as.Date("30-11-2001", format = "%d-%m-%Y"), by = 'day'),
                     seq(as.Date("01-05-2002", format = "%d-%m-%Y"),
                         as.Date("30-11-2002", format = "%d-%m-%Y"), by = 'day'))
dim(exp$attrs$Dates) <- c(sdate = 3, time = 214)
class(exp) <- 's2dv_cube'
exp_probs <- CST_Threshold(exp, threshold, start = list(21, 4), end = list(21, 6))
```

CST_TotalSpellTimeExceedingThreshold

Total Spell Time Exceeding Threshold

Description

The number of days (when daily data is provided) that are part of a spell (defined by its minimum length e.g. 6 consecutive days) that exceed (or not exceed) a threshold are calculated with TotalSpellTimeExceedingThreshold. This function allows to compute indicators widely used in Climate Services, such as:

- 'WSDI', Warm Spell Duration Index that count the total number of days with at least 6 consecutive days when the daily temperature maximum exceeds its 90th percentile.

This function requires the data and the threshold to be in the same units. The 90th percentile can be translate into absolute values given a reference dataset using function Threshold or the data can be transform into probabilites by using function AbsToProbs. See section @examples.

Usage

```
CST_TotalSpellTimeExceedingThreshold(
  data,
  threshold,
  spell,
  op = ">",
```

```
  start = NULL,
  end = NULL,
  time_dim = "time",
  ncores = NULL
)
```

Arguments

data	An 's2dv_cube' object as provided function CST_Start or CST_Load in package CSTools.
threshold	If only one threshold is used, it can be an 's2dv_cube' object or a multidimensional array with named dimensions. It must be in the same units and with the common dimensions of the same length as parameter 'data'. It can also be a vector with the same length of 'time_dim' from 'data' or a scalar. If we want to use two thresholds: it can be a vector of two scalars, a list of two vectors with the same length of 'time_dim' from 'data' or a list of two multidimensional arrays with the common dimensions of the same length as parameter 'data'. If two thresholds are used, parameter 'op' must be also a vector of two elements.
spell	A scalar indicating the minimum length of the spell.
op	An operator '>' (by default), '<', '>=' or '<='. If two thresholds are used it has to be a vector of a pair of two logical operators: c('<', '>'), c('<', '>='), c('<=', '>'), c('<=', '>='), c('>', '<'), c('>', '<='), c('>=', '<'),c('>=', '<=')).
start	An optional parameter to define the initial date of the period to select from the data by providing a list of two elements: the initial date of the period and the initial month of the period. By default it is set to NULL and the indicator is computed using all the data provided in data.
end	An optional parameter to defined the final date of the period to select from the data by providing a list of two elements: the final day of the period and the final month of the period. By default it is set to NULL and the indicator is computed using all the data provided in data.
time_dim	A character string indicating the name of the dimension to compute the indicator. By default, it is set to 'time'. It can only indicate one time dimension.
ncores	An integer indicating the number of cores to use in parallel computation.

Value

An 's2dv_cube' object containing the number of days that are part of a spell within a threshold in element data with dimensions of the input parameter 'data' except the dimension where the indicator has been computed. The 'Dates' array is updated to the dates corresponding to the beginning of the aggregated time period. A new element called 'time_bounds' will be added into the 'attrs' element in the 's2dv_cube' object. It consists of a list containing two elements, the start and end dates of the aggregated period with the same dimensions of 'Dates' element.

See Also

[Threshold()] and [AbsToProbs()].

Examples

```
exp <- NULL
exp$data <- array(rnorm(5 * 3 * 214 * 2)*23,
                  c(member = 5, sdate = 3, time = 214, lon = 2))
exp$attrs$Dates <- c(seq(as.Date("01-05-2000", format = "%d-%m-%Y"),
                         as.Date("30-11-2000", format = "%d-%m-%Y"), by = 'day'),
                     seq(as.Date("01-05-2001", format = "%d-%m-%Y"),
                         as.Date("30-11-2001", format = "%d-%m-%Y"), by = 'day'),
                     seq(as.Date("01-05-2002", format = "%d-%m-%Y"),
                         as.Date("30-11-2002", format = "%d-%m-%Y"), by = 'day'))
dim(exp$attrs$Dates) <- c(sdate = 3, time = 214)
class(exp) <- 's2dv_cube'
TTSET <- CST_TotalSpellTimeExceedingThreshold(exp, threshold = 23, spell = 3,
                                              start = list(21, 4),
                                              end = list(21, 6))
```

`CST_TotalTimeExceedingThreshold` *Total Time of a variable Exceeding (not exceeding) a Threshold*

Description

The Total Time of a variable exceeding (or not) a Threshold. It returns the total number of days (if the data provided is daily, or the corresponding units of the data frequency) that a variable is exceeding a threshold during a period. The threshold provided must be in the same units as the variable units, i.e. to use a percentile as a scalar, the function `AbsToProbs` or `QThreshold` may be needed (see examples). Providing maximum temperature daily data, the following agriculture indices for heat stress can be obtained by using this function:

- 'SU35', Total count of days when daily maximum temperatures exceed 35°C in the seven months from the start month given (e.g. from April to October for start month of April).
- 'SU36', Total count of days when daily maximum temperatures exceed 36 between June 21st and September 21st.
- 'SU40', Total count of days when daily maximum temperatures exceed 40 between June 21st and September 21st.
- 'Spr32', Total count of days when daily maximum temperatures exceed 32 between April 21st and June 21st.

Usage

```
CST_TotalTimeExceedingThreshold(
  data,
  threshold,
  op = ">",
  start = NULL,
  end = NULL,
```

```
  time_dim = "time",
  na.rm = FALSE,
  ncores = NULL
)
```

Arguments

data	An 's2dv_cube' object as provided function CST_Start or CST_Load in package CSTools.
threshold	If only one threshold is used, it can be an 's2dv_cube' object or a multidimensional array with named dimensions. It must be in the same units and with the common dimensions of the same length as parameter 'data'. It can also be a vector with the same length of 'time_dim' from 'data' or a scalar. If we want to use two thresholds: it can be a vector of two scalars, a list of two vectors with the same length of 'time_dim' from 'data' or a list of two multidimensional arrays with the common dimensions of the same length as parameter 'data'. If two thresholds are used, parameter 'op' must be also a vector of two elements.
op	An operator '>' (by default), '<', '>=' or '<='. If two thresholds are used it has to be a vector of a pair of two logical operators: c('<', '>'), c('<', '>='), c('<=', '>'), c('<=', '>='), c('>', '<'), c('>', '<='), c('>=', '<'),c('>=', '<=')).
start	An optional parameter to define the initial date of the period to select from the data by providing a list of two elements: the initial date of the period and the initial month of the period. By default it is set to NULL and the indicator is computed using all the data provided in data.
end	An optional parameter to define the final date of the period to select from the data by providing a list of two elements: the final day of the period and the final month of the period. By default it is set to NULL and the indicator is computed using all the data provided in data.
time_dim	A character string indicating the name of the dimension to compute the indicator. By default, it is set to 'time'. It can only indicate one time dimension.
na.rm	A logical value indicating whether to ignore NA values (TRUE) or not (FALSE).
ncores	An integer indicating the number of cores to use in parallel computation.

Value

An 's2dv_cube' object containing in element data the total number of the corresponding units of the data frequency that a variable is exceeding a threshold during a period with dimensions of the input parameter 'data' except the dimension where the indicator has been computed. The 'Dates' array is updated to the dates corresponding to the beginning of the aggregated time period. A new element called 'time_bounds' will be added into the 'attrs' element in the 's2dv_cube' object. It consists of a list containing two elements, the start and end dates of the aggregated period with the same dimensions of 'Dates' element.

Examples

```
exp <- NULL
exp$data <- array(rnorm(5 * 3 * 214 * 2)*23,
```

```
                  c(member = 5, sdate = 3, time = 214, lon = 2))
exp$attrs$Dates <- c(seq(as.Date("01-05-2000", format = "%d-%m-%Y"),
                         as.Date("30-11-2000", format = "%d-%m-%Y"), by = 'day'),
                     seq(as.Date("01-05-2001", format = "%d-%m-%Y"),
                         as.Date("30-11-2001", format = "%d-%m-%Y"), by = 'day'),
                     seq(as.Date("01-05-2002", format = "%d-%m-%Y"),
                         as.Date("30-11-2002", format = "%d-%m-%Y"), by = 'day'))
dim(exp$attrs$Dates) <- c(sdate = 3, time = 214)
class(exp) <- 's2dv_cube'
DOT <- CST_TotalTimeExceedingThreshold(exp, threshold = 23, start = list(21, 4),
                                       end = list(21, 6))
```

`CST_WindCapacityFactor` *Wind capacity factor on s2dv_cube objects*

Description

Wind capacity factor computes the wind power generated by a specific wind turbine model under specific wind speed conditions, and expresses it as a fraction of the rated capacity (i.e. maximum power) of the turbine.

It is computed by means of a tabular power curve that relates wind speed to power output. The tabular values are interpolated with a linear piecewise approximating function to obtain a smooth power curve. Five different power curves that span different IEC classes can be selected (see below).

Usage

```
CST_WindCapacityFactor(
  wind,
  IEC_class = c("I", "I/II", "II", "II/III", "III"),
  start = NULL,
  end = NULL,
  time_dim = "time",
  ncores = NULL
)
```

Arguments

`wind`	An s2dv_cube object with instantaneous wind speeds expressed in m/s.
`IEC_class`	A string indicating the IEC wind class (see IEC 61400-1) of the turbine to be selected. Classes `'I'`, `'II'` and `'III'` are suitable for sites with an annual mean wind speed of 10, 8.5 and 7.5 m/s respectively. Classes `'I/II'` and `'II/III'` indicate intermediate turbines that fit both classes. More details of the five turbines and a plot of its power curves can be found in Lledó et al. (2019).

start	An optional parameter to defined the initial date of the period to select from the data by providing a list of two elements: the initial date of the period and the initial month of the period. By default it is set to NULL and the indicator is computed using all the data provided in data.
end	An optional parameter to defined the final date of the period to select from the data by providing a list of two elements: the final day of the period and the final month of the period. By default it is set to NULL and the indicator is computed using all the data provided in data.
time_dim	A character string indicating the name of the dimension to compute the indicator. By default, it is set to 'time'. More than one dimension name matching the dimensions provided in the object data$data can be specified.
ncores	An integer indicating the number of cores to use in parallel computation for temporal subsetting.

Value

An s2dv_cube object containing the Wind Capacity Factor (unitless).

Author(s)

Llorenç Lledó, <llledo@bsc.es>

References

Lledó, Ll., Torralba, V., Soret, A., Ramon, J., & Doblas-Reyes, F. J. (2019). Seasonal forecasts of wind power generation. Renewable Energy, 143, 91–100. https://doi.org/10.1016/j.renene.2019.04.135

International Standard IEC 61400-1 (third ed.) (2005)

Examples

```
wind <- NULL
wind$data <- array(rweibull(n = 100, shape = 2, scale = 6),
                   c(member = 5, sdate = 3, time = 214, lon = 2, lat = 5))
wind$coords <- list(lat =  c(40, 41), lon = 1:5)
variable <- list(varName = 'sfcWind',
                 metadata = list(sfcWind = list(level = 'Surface')))
wind$attrs <- list(Variable = variable, Datasets = 'synthetic',
                   when = Sys.time(), Dates = '1990-01-01 00:00:00')
Dates <- c(seq(as.Date("01-05-2000", format = "%d-%m-%Y"),
                       as.Date("30-11-2000", format = "%d-%m-%Y"), by = 'day'),
                     seq(as.Date("01-05-2001", format = "%d-%m-%Y"),
                       as.Date("30-11-2001", format = "%d-%m-%Y"), by = 'day'),
                     seq(as.Date("01-05-2002", format = "%d-%m-%Y"),
                       as.Date("30-11-2002", format = "%d-%m-%Y"), by = 'day'))
dim(Dates) <- c(sdate = 3, time = 214)
wind$attrs$Dates <- Dates
class(wind) <- 's2dv_cube'
WCF <- CST_WindCapacityFactor(wind, IEC_class = "III",
                              start = list(21, 4), end = list(21, 6))
```

`CST_WindPowerDensity` *Wind power density on s2dv_cube objects*

Description

Wind Power Density computes the wind power that is available for extraction per square meter of swept area.

It is computed as 0.5*ro*wspd^3. As this function is non-linear, it will give inaccurate results if used with period means.

Usage

```
CST_WindPowerDensity(
  wind,
  ro = 1.225,
  start = NULL,
  end = NULL,
  time_dim = "time",
  ncores = NULL
)
```

Arguments

`wind`	An 's2dv_cube' object with instantaneous wind speeds expressed in m/s obtained from CST_Start or s2dv_cube functions from CSTools pacakge.
`ro`	A scalar, or alternatively a multidimensional array with the same dimensions as wind, with the air density expressed in kg/m^3. By default it takes the value 1.225, the standard density of air at 15ºC and 1013.25 hPa.
`start`	An optional parameter to defined the initial date of the period to select from the data by providing a list of two elements: the initial date of the period and the initial month of the period. By default it is set to NULL and the indicator is computed using all the data provided in `data`.
`end`	An optional parameter to defined the final date of the period to select from the data by providing a list of two elements: the final day of the period and the final month of the period. By default it is set to NULL and the indicator is computed using all the data provided in `data`.
`time_dim`	A character string indicating the name of the dimension to compute the indicator. By default, it is set to 'time'. More than one dimension name matching the dimensions provided in the object `data$data` can be specified.
`ncores`	An integer indicating the number of cores to use in parallel computation for temporal subsetting.

Value

An s2dv_cube object containing Wind Power Density expressed in W/m^2.

Author(s)

Llorenç Lledó, <lledo@bsc.es>

Examples

```
wind <- NULL
wind$data <- array(rweibull(n = 100, shape = 2, scale = 6),
                   c(member = 5, sdate = 3, time = 214, lon = 2, lat = 5))
wind$coords <- list(lat =  c(40, 41), lon = 1:5)
variable <- list(varName = 'sfcWind',
                 metadata = list(sfcWind = list(level = 'Surface')))
wind$attrs <- list(Variable = variable, Datasets = 'synthetic',
                   when = Sys.time(), Dates = '1990-01-01 00:00:00')
Dates <- c(seq(as.Date("01-05-2000", format = "%d-%m-%Y"),
                       as.Date("30-11-2000", format = "%d-%m-%Y"), by = 'day'),
                     seq(as.Date("01-05-2001", format = "%d-%m-%Y"),
                       as.Date("30-11-2001", format = "%d-%m-%Y"), by = 'day'),
                     seq(as.Date("01-05-2002", format = "%d-%m-%Y"),
                       as.Date("30-11-2002", format = "%d-%m-%Y"), by = 'day'))
dim(Dates) <- c(sdate = 3, time = 214)
wind$attrs$Dates <- Dates
class(wind) <- 's2dv_cube'
WPD <- CST_WindPowerDensity(wind, start = list(21, 4),
                            end = list(21, 6))
```

MergeRefToExp *Merge a Reference To Experiments*

Description

Some indicators are defined for specific temporal periods (e.g.: summer from June 21st to September 21st). If the initialization forecast date is later than the one required for the indicator (e.g.: July 1st), the user may want to merge past observations, or other references, to the forecast (or hindcast) to compute the indicator. If the forecast simulation doesn't cover the required period because it is initialized too early (e.g.: Initialization on November 1st the forecast covers until the beginning of June next year), a climatology (or other references) could be added at the end of the forecast lead time to cover the desired period (e.g.: until the end of summer).

Usage

```
MergeRefToExp(
  data1,
  data2,
  dates1 = NULL,
  dates2 = NULL,
  start1 = NULL,
  end1 = NULL,
```

```
  start2 = NULL,
  end2 = NULL,
  time_dim = "time",
  memb_dim = "member",
  ncores = NULL
)
```

Arguments

data1	A multidimensional array with named dimensions. All dimensions must be equal to 'data2' dimensions except for the ones specified with 'memb_dim' and 'time_dim'.
data2	A multidimensional array of named dimensions matching the dimensions of parameter 'data1'. All dimensions must be equal to 'data1' except for the ones specified with 'memb_dim' and 'time_dim'.
dates1	A multidimensional array of dates with named dimensions matching the temporal dimensions of parameter 'data1'. The common dimensions must be equal to 'data1' dimensions.
dates2	A multidimensional array of dates with named dimensions matching the temporal dimensions on parameter 'data2'. The common dimensions must be equal to 'data2' dimensions.
start1	A list to define the initial date of the period to select from 'data1' by providing a list of two elements: the initial date of the period and the initial month of the period. The initial date of the period must be included in the 'dates1' array.
end1	A list to define the final date of the period to select from 'data1' by providing a list of two elements: the final day of the period and the final month of the period. The final date of the period must be included in the 'dates1' array.
start2	A list to define the initial date of the period to select from 'data2' by providing a list of two elements: the initial date of the period and the initial month of the period. The initial date of the period must be included in the 'dates2' array.
end2	A list to define the final date of the period to select from 'data2' by providing a list of two elements: the final day of the period and the final month of the period. The final date of the period must be included in the 'dates2' array.
time_dim	A character string indicating the name of the temporal dimension that will be used to combine the two arrays. By default, it is set to 'time'. Also, it will be used to subset the data in a requested period.
memb_dim	A character string indicating the name of the member dimension. If the 'data1' and 'data2' have no member dimension, set it as NULL. It is set as 'member' by default.
ncores	An integer indicating the number of cores to use in parallel computation.

Details

This function is created to merge observations and forecasts, known as the 'blending' strategy (see references). The basis for this strategy is that the predictions are progressively replaced with observational data as soon as they become available (i.e., when entering the indicator definition period). This key strategy aims to increase users' confidence in the reformed predictions.

Value

A multidimensional array created from the combination of 'data1' and 'data2'. The resulting array will contain the following dimensions: the original dimensions of the input data, which are common to both arrays and for the 'time_dim' dimension, the sum of the corresponding dimension of 'data1' and 'data2'. If 'memb_dim' is not null, regarding member dimension, two different situations can occur: (1) in the case that one of the arrays does not have member dimension or is equal to 1 and the other array has multiple member dimension, the result will contain the repeated values of the array one up to the lenght of member dimension of array two; (2) in the case that both arrays have member dimension and is greater than 1, all combinations of member dimension will be returned.

References

Chou, C., R. Marcos-Matamoros, L. Palma Garcia, N. Pérez-Zanón, M. Teixeira, S. Silva, N. Fontes, A. Graça, A. Dell'Aquila, S. Calmanti and N. González-Reviriego (2023). Advanced seasonal predictions for vine management based on bioclimatic indicators tailored to the wine sector. Climate Services, 30, 100343, doi: 10.1016/j.cliser.2023.100343.

Examples

```
data_dates <- c(seq(as.Date("01-07-1993", "%d-%m-%Y", tz = 'UTC'),
                    as.Date("01-12-1993","%d-%m-%Y", tz = 'UTC'), "day"),
                seq(as.Date("01-07-1994", "%d-%m-%Y", tz = 'UTC'),
                    as.Date("01-12-1994","%d-%m-%Y", tz = 'UTC'), "day"))
dim(data_dates) <- c(time = 154, sdate = 2)
ref_dates <- seq(as.Date("01-01-1993", "%d-%m-%Y", tz = 'UTC'),
                 as.Date("01-12-1994","%d-%m-%Y", tz = 'UTC'), "day")
dim(ref_dates) <- c(time = 350, sdate = 2)
ref <- array(1001:1700, c(time = 350, sdate = 2))
data <- array(1:(2*154*2), c(time = 154, sdate = 2, member = 2))
new_data <- MergeRefToExp(data1 = ref, dates1 = ref_dates, start1 = list(21, 6),
                          end1 = list(30, 6), data2 = data, dates2 = data_dates,
                          start2 = list(1, 7), end = list(21, 9),
                          time_dim = 'time')
```

`PeriodAccumulation` *Period Accumulation on multidimensional array objects*

Description

Period Accumulation computes the sum (accumulation) of a given variable in a period. Providing precipitation data, two agriculture indices can be obtained by using this function:

- 'SprR', Spring Total Precipitation: The total precipitation from April 21th to June 21st.
- 'HarR', Harvest Total Precipitation: The total precipitation from August 21st to October 21st.

Usage

```
PeriodAccumulation(
  data,
  dates = NULL,
  start = NULL,
  end = NULL,
  time_dim = "time",
  rollwidth = NULL,
  sdate_dim = "sdate",
  frequency = "monthly",
  na.rm = FALSE,
  ncores = NULL
)
```

Arguments

data	A multidimensional array with named dimensions.
dates	A multidimensional array of dates with named dimensions matching the temporal dimensions on parameter 'data'. By default it is NULL, to select aperiod this parameter must be provided.
start	An optional parameter to defined the initial date of the period to select from the data by providing a list of two elements: the initial date of the period and the initial month of the period. By default it is set to NULL and the indicator is computed using all the data provided in data.
end	An optional parameter to defined the final date of the period to select from the data by providing a list of two elements: the final day of the period and the final month of the period. By default it is set to NULL and the indicator is computed using all the data provided in data.
time_dim	A character string indicating the name of the dimension to compute the indicator. By default, it is set to 'time'.
rollwidth	An optional parameter to indicate the number of time steps the rolling sum is applied to. If it is positive, the rolling sum is applied backwards 'time_dim', if it is negative, it will be forward it. When this parameter is NULL, the sum is applied over all 'time_dim', in a specified period. It is NULL by default.
sdate_dim	(Only needed when rollwidth is used). A character string indicating the name of the start date dimension to compute the rolling accumulation. By default, it is set to 'sdate'.
frequency	(Only needed when rollwidth is used). A character string indicating the time frequency of the data to apply the rolling accumulation. It can be 'daily' or 'monthly'. If it is set to 'monthly', values from continuous months will be accumulated; if it is 'daliy', values from continuous days will be accumulated. It is set to 'monthly' by default.
na.rm	A logical value indicating whether to ignore NA values (TRUE) or not (FALSE).
ncores	An integer indicating the number of cores to use in parallel computation.

Details

There are two possible ways of performing the accumulation. The default one is by accumulating a variable over a dimension specified with 'time_dim'. To chose a specific time period, 'start' and 'end' must be used. The other method is by using 'rollwidth' parameter. When this parameter is a positive integer, the cumulative backward sum is applied to the time dimension. If it is negative, the rolling sum is applied backwards.

Value

A multidimensional array with named dimensions containing the accumulated data in the element `data`. If parameter 'rollwidth' is not used, it will have the dimensions of the input 'data' except the dimension where the accumulation has been computed (specified with 'time_dim'). If 'rollwidth' is used, it will be of same dimensions as input data.

Examples

```
exp <- array(rnorm(216)*200, dim = c(dataset = 1, member = 2, sdate = 3,
            ftime = 9, lat = 2, lon = 2))
TP <- PeriodAccumulation(exp, time_dim = 'ftime')
data <- array(rnorm(5 * 3 * 214 * 2),
             c(memb = 5, sdate = 3, ftime = 214, lon = 2))
Dates <- c(seq(as.Date("01-05-2000", format = "%d-%m-%Y"),
              as.Date("30-11-2000", format = "%d-%m-%Y"), by = 'day'),
           seq(as.Date("01-05-2001", format = "%d-%m-%Y"),
              as.Date("30-11-2001", format = "%d-%m-%Y"), by = 'day'),
           seq(as.Date("01-05-2002", format = "%d-%m-%Y"),
              as.Date("30-11-2002", format = "%d-%m-%Y"), by = 'day'))
dim(Dates) <- c(sdate = 3, ftime = 214)
SprR <- PeriodAccumulation(data, dates = Dates, start = list(21, 4),
                           end = list(21, 6), time_dim = 'ftime')
HarR <- PeriodAccumulation(data, dates = Dates, start = list(21, 8),
                           end = list(21, 10), time_dim = 'ftime')
```

PeriodMax *Period max on multidimensional array objects*

Description

Period max computes the average (max) of a given variable in a period. Two bioclimatic indicators can be obtained by using this function:

- 'BIO5', (Providing temperature data) Max Temperature of Warmest Month. The maximum monthly temperature occurrence over a given year (time-series) or averaged span of years (normal).
- 'BIO13', (Providing precipitation data) Precipitation of Wettest Month. This index identifies the total precipitation that prevails during the wettest month.

Usage

```
PeriodMax(
  data,
  dates = NULL,
  start = NULL,
  end = NULL,
  time_dim = "time",
  na.rm = FALSE,
  ncores = NULL
)
```

Arguments

data	A multidimensional array with named dimensions.
dates	A multidimensional array of dates with named dimensions matching the temporal dimensions on parameter 'data'. By default it is NULL, to select aperiod this parameter must be provided.
start	An optional parameter to defined the initial date of the period to select from the data by providing a list of two elements: the initial date of the period and the initial month of the period. By default it is set to NULL and the indicator is computed using all the data provided in data.
end	An optional parameter to defined the final date of the period to select from the data by providing a list of two elements: the final day of the period and the final month of the period. By default it is set to NULL and the indicator is computed using all the data provided in data.
time_dim	A character string indicating the name of the dimension to compute the indicator. By default, it is set to 'time'. More than one dimension name matching the dimensions provided in the object data$data can be specified.
na.rm	A logical value indicating whether to ignore NA values (TRUE) or not (FALSE).
ncores	An integer indicating the number of cores to use in parallel computation.

Value

A multidimensional array with named dimensions containing the indicator in the element data.

Examples

```
data <- array(rnorm(45), dim = c(member = 7, sdate = 4, time = 3))
Dates <- c(seq(as.Date("2000-11-01", "%Y-%m-%d", tz = "UTC"),
              as.Date("2001-01-01", "%Y-%m-%d", tz = "UTC"), by = "month"),
           seq(as.Date("2001-11-01", "%Y-%m-%d", tz = "UTC"),
              as.Date("2002-01-01", "%Y-%m-%d", tz = "UTC"), by = "month"),
           seq(as.Date("2002-11-01", "%Y-%m-%d", tz = "UTC"),
              as.Date("2003-01-01", "%Y-%m-%d", tz = "UTC"), by = "month"),
           seq(as.Date("2003-11-01", "%Y-%m-%d", tz = "UTC"),
              as.Date("2004-01-01", "%Y-%m-%d", tz = "UTC"), by = "month"))
dim(Dates) <- c(sdate = 4, time = 3)
res <- PeriodMax(data, dates = Dates, start = list(01, 12), end = list(01, 01))
```

PeriodMean *Period Mean on multidimensional array objects*

Description

Period Mean computes the average (mean) of a given variable in a period. Providing temperature data, two agriculture indices can be obtained by using this function:

- 'GST', Growing Season average Temperature: The average temperature from April 1st to Octobe 31st.
- 'SprTX', Spring Average Maximum Temperature: The average daily maximum temperature from April 1st to May 31st.

Usage

```
PeriodMean(
  data,
  dates = NULL,
  start = NULL,
  end = NULL,
  time_dim = "time",
  na.rm = FALSE,
  ncores = NULL
)
```

Arguments

data	A multidimensional array with named dimensions.
dates	A multidimensional array of dates with named dimensions matching the temporal dimensions on parameter 'data'. By default it is NULL, to select aperiod this parameter must be provided.
start	An optional parameter to defined the initial date of the period to select from the data by providing a list of two elements: the initial date of the period and the initial month of the period. By default it is set to NULL and the indicator is computed using all the data provided in data.
end	An optional parameter to defined the final date of the period to select from the data by providing a list of two elements: the final day of the period and the final month of the period. By default it is set to NULL and the indicator is computed using all the data provided in data.
time_dim	A character string indicating the name of the dimension to compute the indicator. By default, it is set to 'time'. More than one dimension name matching the dimensions provided in the object data$data can be specified.
na.rm	A logical value indicating whether to ignore NA values (TRUE) or not (FALSE).
ncores	An integer indicating the number of cores to use in parallel computation.

Value

A multidimensional array with named dimensions containing the indicator in the element `data`.

Examples

```
data <- array(rnorm(45), dim = c(member = 7, sdate = 4, time = 3))
Dates <- c(seq(as.Date("2000-11-01", "%Y-%m-%d", tz = "UTC"),
               as.Date("2001-01-01", "%Y-%m-%d", tz = "UTC"), by = "month"),
           seq(as.Date("2001-11-01", "%Y-%m-%d", tz = "UTC"),
               as.Date("2002-01-01", "%Y-%m-%d", tz = "UTC"), by = "month"),
           seq(as.Date("2002-11-01", "%Y-%m-%d", tz = "UTC"),
               as.Date("2003-01-01", "%Y-%m-%d", tz = "UTC"), by = "month"),
           seq(as.Date("2003-11-01", "%Y-%m-%d", tz = "UTC"),
               as.Date("2004-01-01", "%Y-%m-%d", tz = "UTC"), by = "month"))
dim(Dates) <- c(sdate = 4, time = 3)
SA <- PeriodMean(data, dates = Dates, start = list(01, 12), end = list(01, 01))
```

`PeriodMin` *Period Min on multidimensional array objects*

Description

Period Min computes the average (min) of a given variable in a period. Two bioclimatic indicators can be obtained by using this function:

- 'BIO6', (Providing temperature data) Min Temperature of Coldest Month. The minimum monthly temperature occurrence over a given year (time-series) or averaged span of years (normal).
- 'BIO14', (Providing precipitation data) Precipitation of Driest Month. This index identifies the total precipitation that prevails during the driest month.

Usage

```
PeriodMin(
  data,
  dates = NULL,
  start = NULL,
  end = NULL,
  time_dim = "time",
  na.rm = FALSE,
  ncores = NULL
)
```

Arguments

data	A multidimensional array with named dimensions.
dates	A multidimensional array of dates with named dimensions matching the temporal dimensions on parameter 'data'. By default it is NULL, to select aperiod this parameter must be provided.
start	An optional parameter to defined the initial date of the period to select from the data by providing a list of two elements: the initial date of the period and the initial month of the period. By default it is set to NULL and the indicator is computed using all the data provided in data.
end	An optional parameter to defined the final date of the period to select from the data by providing a list of two elements: the final day of the period and the final month of the period. By default it is set to NULL and the indicator is computed using all the data provided in data.
time_dim	A character string indicating the name of the dimension to compute the indicator. By default, it is set to 'time'. More than one dimension name matching the dimensions provided in the object data$data can be specified.
na.rm	A logical value indicating whether to ignore NA values (TRUE) or not (FALSE).
ncores	An integer indicating the number of cores to use in parallel computation.

Value

A multidimensional array with named dimensions containing the indicator in the element data.

Examples

```
data <- array(rnorm(45), dim = c(member = 7, sdate = 4, time = 3))
Dates <- c(seq(as.Date("2000-11-01", "%Y-%m-%d", tz = "UTC"),
               as.Date("2001-01-01", "%Y-%m-%d", tz = "UTC"), by = "month"),
           seq(as.Date("2001-11-01", "%Y-%m-%d", tz = "UTC"),
               as.Date("2002-01-01", "%Y-%m-%d", tz = "UTC"), by = "month"),
           seq(as.Date("2002-11-01", "%Y-%m-%d", tz = "UTC"),
               as.Date("2003-01-01", "%Y-%m-%d", tz = "UTC"), by = "month"),
           seq(as.Date("2003-11-01", "%Y-%m-%d", tz = "UTC"),
               as.Date("2004-01-01", "%Y-%m-%d", tz = "UTC"), by = "month"))
dim(Dates) <- c(sdate = 4, time = 3)
res <- PeriodMin(data, dates = Dates, start = list(01, 12), end = list(01, 01))
```

PeriodPET *Compute the Potential Evapotranspiration*

Description

Compute the Potential Evapotranspiration (PET) that is the amount of evaporation and transpiration that would occur if a sufficient water source were available. This function calculate PET according to the Thornthwaite, Hargreaves or Hargreaves-modified equations.

Usage

```
PeriodPET(
  data,
  dates,
  lat,
  pet_method = "hargreaves",
  time_dim = "syear",
  leadtime_dim = "time",
  lat_dim = "latitude",
  na.rm = FALSE,
  ncores = NULL
)
```

Arguments

data	A named list of multidimensional arrays containing the seasonal forecast experiment data for each variable. Specific variables are needed for each method used in computing the Potential Evapotranspiration (see parameter 'pet_method'). The accepted variable names are fixed in order to be recognized by the function. The accepted name corresponding to the Minimum Temperature is 'tmin', for Maximum Temperature is 'tmax', for Mean Temperature is 'tmean' and for Precipitation is 'pr'. The accepted variable names for each method are: For 'hargreaves': 'tmin' and 'tmax'; for 'hargreaves_modified' are 'tmin', 'tmax' and 'pr'; for method 'thornthwaite' 'tmean' is required. The units for temperature variables ('tmin', 'tmax' and 'tmean') need to be in Celcius degrees; the units for precipitation ('pr') need to be in mm/month. Currently the function works only with monthly data from different years.
dates	An array of temporal dimensions containing the Dates of 'data'. It must be of class 'Date' or 'POSIXct'.
lat	A numeric vector containing the latitude values of 'data'.
pet_method	A character string indicating the method used to compute the potential evapotranspiration. The accepted methods are: 'hargreaves' and 'hargreaves_modified', that require the data to have variables tmin and tmax; and 'thornthwaite', that requires variable 'tmean'.
time_dim	A character string indicating the name of the temporal dimension. By default, it is set to 'syear'.
leadtime_dim	A character string indicating the name of the temporal dimension. By default, it is set to 'time'.
lat_dim	A character string indicating the name of the latitudinal dimension. By default it is set by 'latitude'.
na.rm	A logical value indicating whether NA values should be removed from data. It is FALSE by default.
ncores	An integer value indicating the number of cores to use in parallel computation.

Details

For more information on the SPEI calculation, see functions PeriodStandardization and PeriodAccumulation.

Examples

```
dims <- c(time = 3, syear = 3, ensemble = 1, latitude = 1)
exp_tasmax <- array(rnorm(360, 27.73, 5.26), dim = dims)
exp_tasmin <- array(rnorm(360, 14.83, 3.86), dim = dims)
exp_prlr <- array(rnorm(360, 21.19, 25.64), dim = dims)
end_year <- 2012
dates_exp <- as.POSIXct(c(paste0(2010:end_year, "-08-16"),
                          paste0(2010:end_year, "-09-15"),
                          paste0(2010:end_year, "-10-16")), "UTC")
dim(dates_exp) <- c(syear = 3, time = 3)
lat <- c(40)
exp1 <- list('tmax' = exp_tasmax, 'tmin' = exp_tasmin, 'pr' = exp_prlr)
res <- PeriodPET(data = exp1, lat = lat, dates = dates_exp)
```

PeriodStandardization *Compute the Standardization of Precipitation-Evapotranspiration Index*

Description

The Standardization of the data is the last step of computing the SPEI indicator. With this function the data is fit to a probability distribution to transform the original values to standardized units that are comparable in space and time and at different SPEI time scales.

Usage

```
PeriodStandardization(
  data,
  data_cor = NULL,
  dates = NULL,
  time_dim = "syear",
  leadtime_dim = "time",
  memb_dim = "ensemble",
  ref_period = NULL,
  handle_infinity = FALSE,
  method = "parametric",
  distribution = "log-Logistic",
  params = NULL,
  return_params = FALSE,
  na.rm = FALSE,
  ncores = NULL
)
```

Arguments

data	A multidimensional array containing the data to be standardized.
data_cor	A multidimensional array containing the data in which the standardization should be applied using the fitting parameters from 'data'.
dates	An array containing the dates of the data with the same time dimensions as the data. It is optional and only necessary for using the parameter 'ref_period' to select a reference period directly from dates.
time_dim	A character string indicating the name of the temporal dimension. By default, it is set to 'syear'.
leadtime_dim	A character string indicating the name of the temporal dimension. By default, it is set to 'time'.
memb_dim	A character string indicating the name of the dimension in which the ensemble members are stored. When set it to NULL, threshold is computed for individual members.
ref_period	A list with two numeric values with the starting and end points of the reference period used for computing the index. The default value is NULL indicating that the first and end values in data will be used as starting and end points.
handle_infinity	A logical value wether to return infinite values (TRUE) or not (FALSE). When it is TRUE, the positive infinite values (negative infinite) are substituted by the maximum (minimum) values of each computation step, a subset of the array of dimensions time_dim, leadtime_dim and memb_dim.
method	A character string indicating the standardization method used. If can be: 'parametric' or 'non-parametric'. It is set to 'parametric' by default.
distribution	A character string indicating the name of the distribution function to be used for computing the SPEI. The accepted names are: 'log-Logistic' and 'Gamma'. It is set to 'log-Logistic' by default. The 'Gamma' method only works when only precipitation is provided and other variables are 0 because it is positive defined (SPI indicator).
params	An optional parameter that needs to be a multidimensional array with named dimensions. This option overrides computation of fitting parameters. It needs to be of same time dimensions (specified in 'time_dim' and 'leadtime_dim') of 'data' and a dimension named 'coef' with the length of the coefficients needed for the used distribution (for 'Gamma' coef dimension is of lenght 2, for 'log-Logistic' is of length 3). It also needs to have a leadtime dimension (specified in 'leadtime_dim') of length 1. It will only be used if 'data_cor' is not provided.
return_params	A logical value indicating wether to return parameters array (TRUE) or not (FALSE). It is FALSE by default.
na.rm	A logical value indicating whether NA values should be removed from data. It is FALSE by default. If it is FALSE and there are NA values, standardization cannot be carried out for those coordinates and therefore, the result will be filled with NA for the specific coordinates. If it is TRUE, if the data from other dimensions except time_dim and leadtime_dim is not reaching 4 values, it is not enough values to estimate the parameters and the result will include NA.
ncores	An integer value indicating the number of cores to use in parallel computation.

Details

Next, some specifications for the calculation of the standardization will be discussed. If there are NAs in the data and they are not removed with the parameter 'na.rm', the standardization cannot be carried out for those coordinates and therefore, the result will be filled with NA for the specific coordinates. When NAs are not removed, if the length of the data for a computational step is smaller than 4, there will not be enough data for standarize and the result will be also filled with NAs for that coordinates. About the distribution used to fit the data, there are only two possibilities: 'log-logistic' and 'Gamma'. The 'Gamma' method only works when only precipitation is provided and other variables are 0 because it is positive defined (SPI indicator). When only 'data' is provided ('data_cor' is NULL) the standardization is computed with cross validation. For more information about SPEI, see functions PeriodPET and PeriodAccumulation.

Value

A multidimensional array containing the standardized data. If 'data_cor' is provided the array will be of the same dimensions as 'data_cor'. If 'data_cor' is not provided, the array will be of the same dimensions as 'data'. The parameters of the standardization will only be returned if 'return_params' is TRUE, in this case, the output will be a list of two objects one for the standardized data and one for the parameters.

Examples

```
dims <-  c(syear = 6, time = 2, latitude = 2, ensemble = 25)
dimscor <-  c(syear = 1, time = 2, latitude = 2, ensemble = 25)
data <- array(rnorm(600, -194.5, 64.8), dim = dims)
datacor <- array(rnorm(100, -217.8, 68.29), dim = dimscor)

SPEI <- PeriodStandardization(data = data)
SPEIcor <- PeriodStandardization(data = data, data_cor = datacor)
```

PeriodVariance *Period Variance on multidimensional array objects*

Description

Period Variance computes the average (var) of a given variable in a period. Two bioclimatic indicators can be obtained by using this function:

- 'BIO4', (Providing temperature data) Temperature Seasonality (Standard Deviation). The amount of temperature variation over a given year (or averaged years) based on the standard deviation (variation) of monthly temperature averages.
- 'BIO15', (Providing precipitation data) Precipitation Seasonality (CV). This is a measure of the variation in monthly precipitation totals over the course of the year. This index is the ratio of the standard deviation of the monthly total precipitation to the mean monthly total precipitation (also known as the coefficient of variation) and is expressed as a percentage.

Usage

```
PeriodVariance(
  data,
  dates = NULL,
  start = NULL,
  end = NULL,
  time_dim = "time",
  na.rm = FALSE,
  ncores = NULL
)
```

Arguments

data	A multidimensional array with named dimensions.
dates	A multidimensional array of dates with named dimensions matching the temporal dimensions on parameter 'data'. By default it is NULL, to select aperiod this parameter must be provided.
start	An optional parameter to defined the initial date of the period to select from the data by providing a list of two elements: the initial date of the period and the initial month of the period. By default it is set to NULL and the indicator is computed using all the data provided in data.
end	An optional parameter to defined the final date of the period to select from the data by providing a list of two elements: the final day of the period and the final month of the period. By default it is set to NULL and the indicator is computed using all the data provided in data.
time_dim	A character string indicating the name of the dimension to compute the indicator. By default, it is set to 'time'. More than one dimension name matching the dimensions provided in the object data$data can be specified.
na.rm	A logical value indicating whether to ignore NA values (TRUE) or not (FALSE).
ncores	An integer indicating the number of cores to use in parallel computation.

Value

A multidimensional array with named dimensions containing the indicator in the element data.

Examples

```
data <- array(rnorm(45), dim = c(member = 7, sdate = 4, time = 3))
Dates <- c(seq(as.Date("2000-11-01", "%Y-%m-%d", tz = "UTC"),
              as.Date("2001-01-01", "%Y-%m-%d", tz = "UTC"), by = "month"),
           seq(as.Date("2001-11-01", "%Y-%m-%d", tz = "UTC"),
              as.Date("2002-01-01", "%Y-%m-%d", tz = "UTC"), by = "month"),
           seq(as.Date("2002-11-01", "%Y-%m-%d", tz = "UTC"),
              as.Date("2003-01-01", "%Y-%m-%d", tz = "UTC"), by = "month"),
           seq(as.Date("2003-11-01", "%Y-%m-%d", tz = "UTC"),
              as.Date("2004-01-01", "%Y-%m-%d", tz = "UTC"), by = "month"))
dim(Dates) <- c(sdate = 4, time = 3)
res <- PeriodVariance(data, dates = Dates, start = list(01, 12), end = list(01, 01))
```

QThreshold *Transform an absolute threshold into probabilities*

Description

From the user's perspective, an absolute threshold can be very useful for a specific needs (e.g.: grape variety). However, this absolute threshold could be transformed to a relative threshold in order to get its frequency in a given dataset. Therefore, the function `QThreshold` returns the probability of an absolute threshold. This is done by computing the Cumulative Distribution Function of a sample and leaving-one-ot. The sample used will depend on the dimensions of the data provided and the dimension names provided in sdate_dim and memb_dim parameters:

- If a forecast (hindcast) has dimensions member and start date, and both must be used in the sample, their names should be passed in sdate_dim and memb_dim.
- If a forecast (hindcast) has dimensions member and start date, and only start date must be used in the sample (the calculation is done in each separate member), memb_dim can be set to NULL.
- If a reference (observations) has start date dimension, the sample used is the start date dimension.
- If a reference (observations) doesn't have start date dimension, the sample used must be especified in sdate_dim parameter.

Usage

```
QThreshold(
  data,
  threshold,
  dates = NULL,
  start = NULL,
  end = NULL,
  time_dim = "time",
  memb_dim = "member",
  sdate_dim = "sdate",
  ncores = NULL
)
```

Arguments

`data`	A multidimensional array with named dimensions.
`threshold`	A multidimensional array with named dimensions in the same units as parameter 'data' and with the common dimensions of the element 'data' of the same length.
`dates`	A multidimensional array of dates with named dimensions matching the temporal dimensions on parameter 'data'. By default it is NULL, to select aperiod this parameter must be provided.

start	An optional parameter to defined the initial date of the period to select from the data by providing a list of two elements: the initial date of the period and the initial month of the period. By default it is set to NULL and the indicator is computed using all the data provided in data.
end	An optional parameter to defined the final date of the period to select from the data by providing a list of two elements: the final day of the period and the final month of the period. By default it is set to NULL and the indicator is computed using all the data provided in data.
time_dim	A character string indicating the name of the temporal dimension. By default, it is set to 'time'. More than one dimension name matching the dimensions provided in the object data$data can be specified. This dimension is required to subset the data in a requested period.
memb_dim	A character string indicating the name of the dimension in which the ensemble members are stored.
sdate_dim	A character string indicating the name of the dimension in which the initialization dates are stored.
ncores	An integer indicating the number of cores to use in parallel computation.

Value

A multidimensional array with named dimensions containing the probability of an absolute threshold in the element data.

Examples

```
threshold = 25
data <- array(rnorm(5 * 3 * 20 * 2, mean = 26),
              c(member = 5, sdate = 3, time = 214, lon = 2))

Dates <- c(seq(as.Date("01-05-2000", format = "%d-%m-%Y"),
               as.Date("30-11-2000", format = "%d-%m-%Y"), by = 'day'),
           seq(as.Date("01-05-2001", format = "%d-%m-%Y"),
               as.Date("30-11-2001", format = "%d-%m-%Y"), by = 'day'),
           seq(as.Date("01-05-2002", format = "%d-%m-%Y"),
               as.Date("30-11-2002", format = "%d-%m-%Y"), by = 'day'))
dim(Dates) <- c(sdate = 3, time = 214)

thres_q <- QThreshold(data, threshold, dates = Dates, time_dim = 'time',
                      start = list(21, 4), end = list(21, 6))
```

SelectPeriodOnData *Select a period on Data on multidimensional array objects*

Description

Auxiliary function to subset data for a specific period.

Usage

```
SelectPeriodOnData(data, dates, start, end, time_dim = "time", ncores = NULL)
```

Arguments

data	A multidimensional array with named dimensions with at least the time dimension specified in parameter 'time_dim'. All common dimensions with 'dates' parameter need to have the same length.
dates	An array of dates with named dimensions with at least the time dimension specified in parameter 'time_dim'. All common dimensions with 'data' parameter need to have the same length.
start	A list with two elements to define the initial date of the period to select from the data. The first element is the initial day of the period and the second element is the initial month of the period.
end	A list with two elements to define the final date of the period to select from the data. The first element is the final day of the period and the second element is the final month of the period.
time_dim	A character string indicating the name of the dimension to compute select the dates. By default, it is set to 'time'. Parameters 'data' and 'dates'
ncores	An integer indicating the number of cores to use in parallel computation.

Value

A multidimensional array with named dimensions containing the subset of the object data during the period requested from start to end.

Examples

```
data <- array(rnorm(5 * 3 * 214 * 2),
              c(memb = 5, sdate = 3, time = 214, lon = 2))
Dates <- c(seq(as.Date("01-05-2000", format = "%d-%m-%Y"),
               as.Date("30-11-2000", format = "%d-%m-%Y"), by = 'day'),
           seq(as.Date("01-05-2001", format = "%d-%m-%Y"),
               as.Date("30-11-2001", format = "%d-%m-%Y"), by = 'day'),
           seq(as.Date("01-05-2002", format = "%d-%m-%Y"),
               as.Date("30-11-2002", format = "%d-%m-%Y"), by = 'day'))
dim(Dates) <- c(time = 214, sdate = 3)
Period <- SelectPeriodOnData(data, Dates, start = list(21, 6), end = list(21, 9))
```

SelectPeriodOnDates *Select a period on Dates*

Description

Auxiliary function to subset dates for a specific period.

Usage

```
SelectPeriodOnDates(dates, start, end, time_dim = "time", ncores = NULL)
```

Arguments

dates	An array of dates with named dimensions.
start	An optional parameter to defined the initial date of the period to select from the data by providing a list of two elements: the initial date of the period and the initial month of the period.
end	An optional parameter to defined the final date of the period to select from the data by providing a list of two elements: the final day of the period and the final month of the period.
time_dim	A character string indicating the name of the dimension to compute select the dates. By default, it is set to 'time'. More than one dimension name matching the dimensions provided in the object data$data can be specified.
ncores	An integer indicating the number of cores to use in parallel computation.

Value

A multidimensional array with named dimensions containing the subset of the vector dates during the period requested from start to end.

Examples

```
Dates <- c(seq(as.Date("01-05-2000", format = "%d-%m-%Y"),
               as.Date("30-11-2000", format = "%d-%m-%Y"), by = 'day'),
           seq(as.Date("01-05-2001", format = "%d-%m-%Y"),
               as.Date("30-11-2001", format = "%d-%m-%Y"), by = 'day'),
           seq(as.Date("01-05-2002", format = "%d-%m-%Y"),
               as.Date("30-11-2002", format = "%d-%m-%Y"), by = 'day'))
dim(Dates) <- c(time = 214, sdate = 3)
Period <- SelectPeriodOnDates(Dates, start = list(21, 6), end = list(21, 9))
```

Threshold *Absolute value of a relative threshold (percentile)*

Description

Frequently, thresholds are defined by a percentile that may correspond to a different absolute value depending on the variable, gridpoint and also julian day (time). This function calculates the corresponding value of a percentile given a dataset.

Usage

```
Threshold(
  data,
  threshold,
  dates = NULL,
  start = NULL,
  end = NULL,
  time_dim = "time",
  memb_dim = "member",
  sdate_dim = "sdate",
  na.rm = FALSE,
  ncores = NULL
)
```

Arguments

data	A multidimensional array with named dimensions.
threshold	A single scalar or vector indicating the relative threshold(s). It must contain values between 0 and 1.
dates	A multidimensional array of dates with named dimensions matching the temporal dimensions on parameter 'data'. By default it is NULL, to select aperiod this parameter must be provided.
start	An optional parameter to defined the initial date of the period to select from the data by providing a list of two elements: the initial date of the period and the initial month of the period. By default it is set to NULL and the indicator is computed using all the data provided in data.
end	An optional parameter to defined the final date of the period to select from the data by providing a list of two elements: the final day of the period and the final month of the period. By default it is set to NULL and the indicator is computed using all the data provided in data.
time_dim	A character string indicating the name of the temporal dimension. By default, it is set to 'time'. More than one dimension name matching the dimensions provided in the object data$data can be specified. This dimension is required to subset the data in a requested period.
memb_dim	A character string indicating the name of the dimension in which the ensemble members are stored. When set it to NULL, threshold is computed for individual members.
sdate_dim	A character string indicating the name of the dimension in which the initialization dates are stored.
na.rm	A logical value indicating whether to ignore NA values (TRUE) or not (FALSE).
ncores	An integer indicating the number of cores to use in parallel computation.

Value

A multidimensional array with named dimensions containing the corresponding values of a percentile in the element data.

Examples

```
threshold <- 0.9
data <- array(rnorm(25 * 3 * 214 * 2, mean = 26),
              c(member = 25, sdate = 3, time = 214, lon = 2))
thres_q <- Threshold(data, threshold)
data <- array(rnorm(1 * 3 * 214 * 2), c(member = 1, sdate = 3, time = 214, lon = 2))
res <- Threshold(data, threshold)
```

TotalSpellTimeExceedingThreshold

Total Spell Time Exceeding Threshold

Description

The number of days (when daily data is provided) that are part of a spell (defined by its minimum length e.g. 6 consecutive days) that exceed (or not exceed) a threshold are calculated with `TotalSpellTimeExceedingThreshold`. This function allows to compute indicators widely used in Climate Services, such as:

- 'WSDI', Warm Spell Duration Index that count the total number of days with at least 6 consecutive days when the daily temperature maximum exceeds its 90th percentile.

This function requires the data and the threshold to be in the same units. The 90th percentile can be translate into absolute values given a reference dataset using function `Threshold` or the data can be transform into probabilites by using function `AbsToProbs`. See section @examples.

Usage

```
TotalSpellTimeExceedingThreshold(
  data,
  threshold,
  spell,
  op = ">",
  dates = NULL,
  start = NULL,
  end = NULL,
  time_dim = "time",
  ncores = NULL
)
```

Arguments

`data`	A multidimensional array with named dimensions.
`threshold`	If only one threshold is used: it can be a multidimensional array with named dimensions. It must be in the same units and with the common dimensions of the same length as parameter 'data'. It can also be a vector with the same

	length of 'time_dim' from 'data' or a scalar. If we want to use two thresholds: it can be a vector of two scalars, a list of two vectors with the same length of 'time_dim' from 'data' or a list of two multidimensional arrays with the common dimensions of the same length as parameter 'data'. If two thresholds are used, parameter 'op' must be also a vector of two elements.
spell	A scalar indicating the minimum length of the spell.
op	An operator '>' (by default), '<', '>=' or '<='. If two thresholds are used it has to be a vector of a pair of two logical operators: c('<', '>'), c('<', '>='), c('<=', '>'), c('<=', '>='), c('>', '<'), c('>', '<='), c('>=', '<'),c('>=', '<=')).
dates	A multidimensional array of dates with named dimensions matching the temporal dimensions on parameter 'data'. By default it is NULL, to select aperiod this parameter must be provided.
start	An optional parameter to defined the initial date of the period to select from the data by providing a list of two elements: the initial date of the period and the initial month of the period. By default it is set to NULL and the indicator is computed using all the data provided in data.
end	An optional parameter to define the final date of the period to select from the data by providing a list of two elements: the final day of the period and the final month of the period. By default it is set to NULL and the indicator is computed using all the data provided in data.
time_dim	A character string indicating the name of the dimension to compute the indicator. By default, it is set to 'time'. It can only indicate one time dimension.
ncores	An integer indicating the number of cores to use in parallel computation.

Details

This function considers NA values as the end of the spell. For a different behaviour consider to modify the 'data' input by substituting NA values by values exceeding the threshold.

Value

A multidimensional array with named dimensions containing the number of days that are part of a spell within a threshold with dimensions of the input parameter 'data' except the dimension where the indicator has been computed.

See Also

[Threshold()] and [AbsToProbs()].

Examples

```
data <- array(1:100, c(member = 5, sdate = 3, time = 214, lon = 2))
Dates <- c(seq(as.Date("01-05-2000", format = "%d-%m-%Y"),
               as.Date("30-11-2000", format = "%d-%m-%Y"), by = 'day'),
           seq(as.Date("01-05-2001", format = "%d-%m-%Y"),
               as.Date("30-11-2001", format = "%d-%m-%Y"), by = 'day'),
           seq(as.Date("01-05-2002", format = "%d-%m-%Y"),
               as.Date("30-11-2002", format = "%d-%m-%Y"), by = 'day'))
```

```
dim(Dates) <- c(sdate = 3, time = 214)

threshold <- array(1:4, c(lat = 4))
total <- TotalSpellTimeExceedingThreshold(data, threshold, dates = Dates,
                                          spell = 6, start = list(21, 4),
                                          end = list(21, 6))
```

`TotalTimeExceedingThreshold` *Total Time of a variable Exceeding (not exceeding) a Threshold*

Description

The Total Time of a variable exceeding (or not) a Threshold. It returns the total number of days (if the data provided is daily, or the corresponding units of the data frequency) that a variable is exceeding a threshold during a period. The threshold provided must be in the same units as the variable units, i.e. to use a percentile as a scalar, the function `AbsToProbs` or `QThreshold` may be needed (see examples). Providing maximum temperature daily data, the following agriculture indices for heat stress can be obtained by using this function:

- 'SU35', Total count of days when daily maximum temperatures exceed 35°C in the seven months from the start month given (e.g. from April to October for start month of April).
- 'SU36', Total count of days when daily maximum temperatures exceed 36 between June 21st and September 21st.
- 'SU40', Total count of days when daily maximum temperatures exceed 40 between June 21st and September 21st.
- 'Spr32', Total count of days when daily maximum temperatures exceed 32 between April 21st and June 21st.

Usage

```
TotalTimeExceedingThreshold(
  data,
  threshold,
  op = ">",
  dates = NULL,
  start = NULL,
  end = NULL,
  time_dim = "time",
  na.rm = FALSE,
  ncores = NULL
)
```

Arguments

data	A multidimensional array with named dimensions.
threshold	If only one threshold is used: it can be a multidimensional array with named dimensions. It must be in the same units and with the common dimensions of the same length as parameter 'data'. It can also be a vector with the same length of 'time_dim' from 'data' or a scalar. If we want to use two thresholds: it can be a vector of two scalars, a list of two vectors with the same length of 'time_dim' from 'data' or a list of two multidimensional arrays with the common dimensions of the same length as parameter 'data'. If two thresholds are used, parameter 'op' must be also a vector of two elements.
op	An operator '>' (by default), '<', '>=' or '<='. If two thresholds are used it has to be a vector of a pair of two logical operators: c('<', '>'), c('<', '>='), c('<=', '>'), c('<=', '>='), c('>', '<'), c('>', '<='), c('>=', '<'),c('>=', '<=')).
dates	A multidimensional array of dates with named dimensions matching the temporal dimensions on parameter 'data'. By default it is NULL, to select aperiod this parameter must be provided.
start	An optional parameter to define the initial date of the period to select from the data by providing a list of two elements: the initial date of the period and the initial month of the period. By default it is set to NULL and the indicator is computed using all the data provided in data.
end	An optional parameter to define the final date of the period to select from the data by providing a list of two elements: the final day of the period and the final month of the period. By default it is set to NULL and the indicator is computed using all the data provided in data.
time_dim	A character string indicating the name of the dimension to compute the indicator. By default, it is set to 'time'. It can only indicate one time dimension.
na.rm	A logical value indicating whether to ignore NA values (TRUE) or not (FALSE).
ncores	An integer indicating the number of cores to use in parallel computation.

Value

A multidimensional array with named dimensions containing the total number of the corresponding units of the data frequency that a variable is exceeding a threshold during a period with dimensions of the input parameter 'data' except the dimension where the indicator has been computed.

Examples

```
data <- array(rnorm(5 * 3 * 214 * 2)*23,
              c(member = 5, sdate = 3, time = 214, lon = 2))
Dates <- c(seq(as.Date("01-05-2000", format = "%d-%m-%Y"),
               as.Date("30-11-2000", format = "%d-%m-%Y"), by = 'day'),
           seq(as.Date("01-05-2001", format = "%d-%m-%Y"),
               as.Date("30-11-2001", format = "%d-%m-%Y"), by = 'day'),
           seq(as.Date("01-05-2002", format = "%d-%m-%Y"),
               as.Date("30-11-2002", format = "%d-%m-%Y"), by = 'day'))
dim(Dates) <- c(sdate = 3, time = 214)
DOT <- TotalTimeExceedingThreshold(data, threshold = 23, dates = Dates,
```

```
                                start = list(21, 4), end = list(21, 6))
```

`WindCapacityFactor` *Wind capacity factor*

Description

Wind capacity factor computes the wind power generated by a specific wind turbine model under specific wind speed conditions, and expresses it as a fraction of the rated capacity (i.e. maximum power) of the turbine.

It is computed by means of a tabular power curve that relates wind speed to power output. The tabular values are interpolated with a linear piecewise approximating function to obtain a smooth power curve. Five different power curves that span different IEC classes can be selected (see below).

Usage

```
WindCapacityFactor(
  wind,
  IEC_class = c("I", "I/II", "II", "II/III", "III"),
  dates = NULL,
  start = NULL,
  end = NULL,
  time_dim = "time",
  ncores = NULL
)
```

Arguments

`wind`	A multidimensional array, vector or scalar with instantaneous wind speeds expressed in m/s.
`IEC_class`	A string indicating the IEC wind class (see IEC 61400-1) of the turbine to be selected. Classes `'I'`, `'II'` and `'III'` are suitable for sites with an annual mean wind speed of 10, 8.5 and 7.5 m/s respectively. Classes `'I/II'` and `'II/III'` indicate intermediate turbines that fit both classes. More details of the five turbines and a plot of its power curves can be found in Lledó et al. (2019).
`dates`	A multidimensional array of dates with named dimensions matching the temporal dimensions on parameter 'data'. By default it is NULL, to select aperiod this parameter must be provided.
`start`	An optional parameter to defined the initial date of the period to select from the data by providing a list of two elements: the initial date of the period and the initial month of the period. By default it is set to NULL and the indicator is computed using all the data provided in `data`.
`end`	An optional parameter to defined the final date of the period to select from the data by providing a list of two elements: the final day of the period and the final month of the period. By default it is set to NULL and the indicator is computed using all the data provided in `data`.

time_dim	A character string indicating the name of the dimension to compute the indicator. By default, it is set to 'time'. More than one dimension name matching the dimensions provided in the object data$data can be specified.
ncores	An integer indicating the number of cores to use in parallel computation for temporal subsetting.

Value

An array with the same dimensions as wind, containing the Wind Capacity Factor (unitless).

Author(s)

Llorenç Lledó, <llledo@bsc.es>

References

Lledó, Ll., Torralba, V., Soret, A., Ramon, J., & Doblas-Reyes, F. J. (2019). Seasonal forecasts of wind power generation. Renewable Energy, 143, 91–100. https://doi.org/10.1016/j.renene.2019.04.135

International Standard IEC 61400-1 (third ed.) (2005)

Examples

```
wind <- array(rweibull(n = 32100, shape = 2, scale = 6),
              c(member = 5, sdate = 3, time = 214, lon = 2, lat = 5))

Dates <- c(seq(as.Date("01-05-2000", format = "%d-%m-%Y"),
               as.Date("30-11-2000", format = "%d-%m-%Y"), by = 'day'),
           seq(as.Date("01-05-2001", format = "%d-%m-%Y"),
               as.Date("30-11-2001", format = "%d-%m-%Y"), by = 'day'),
           seq(as.Date("01-05-2002", format = "%d-%m-%Y"),
               as.Date("30-11-2002", format = "%d-%m-%Y"), by = 'day'))
dim(Dates) <- c(sdate = 3, time = 214)

WCF <- WindCapacityFactor(wind, IEC_class = "III", dates = Dates,
                          start = list(21, 4), end = list(21, 6))
```

WindPowerDensity *Wind power density on multidimensional array objects*

Description

Wind Power Density computes the wind power that is available for extraction per square meter of swept area.

It is computed as 0.5*ro*wspd^3. As this function is non-linear, it will give inaccurate results if used with period means.

Usage

```
WindPowerDensity(
  wind,
  ro = 1.225,
  dates = NULL,
  start = NULL,
  end = NULL,
  time_dim = "time",
  ncores = NULL
)
```

Arguments

wind	A multidimensional array, vector or scalar with instantaneous wind speeds expressed in m/s.
ro	A scalar, or alternatively a multidimensional array with the same dimensions as wind, with the air density expressed in kg/m^3. By default it takes the value 1.225, the standard density of air at 15ºC and 1013.25 hPa.
dates	A multidimensional array of dates with named dimensions matching the temporal dimensions on parameter 'data'. By default it is NULL, to select aperiod this parameter must be provided.
start	An optional parameter to defined the initial date of the period to select from the data by providing a list of two elements: the initial date of the period and the initial month of the period. By default it is set to NULL and the indicator is computed using all the data provided in data.
end	An optional parameter to defined the final date of the period to select from the data by providing a list of two elements: the final day of the period and the final month of the period. By default it is set to NULL and the indicator is computed using all the data provided in data.
time_dim	A character string indicating the name of the dimension to compute the indicator. By default, it is set to 'time'. More than one dimension name matching the dimensions provided in the object data$data can be specified.
ncores	An integer indicating the number of cores to use in parallel computation for temporal subsetting.

Value

An array with the same dimensions as wind, containing Wind Power Density expressed in W/m^2.

Author(s)

Llorenç Lledó, <llledo@bsc.es>

Examples

```
wind <- array(rweibull(n = 32100, shape = 2, scale = 6),
              c(member = 5, sdate = 3, time = 214, lon = 2, lat = 5))
```

```
Dates <- c(seq(as.Date("01-05-2000", format = "%d-%m-%Y"),
              as.Date("30-11-2000", format = "%d-%m-%Y"), by = 'day'),
           seq(as.Date("01-05-2001", format = "%d-%m-%Y"),
              as.Date("30-11-2001", format = "%d-%m-%Y"), by = 'day'),
           seq(as.Date("01-05-2002", format = "%d-%m-%Y"),
              as.Date("30-11-2002", format = "%d-%m-%Y"), by = 'day'))
dim(Dates) <- c(sdate = 3, time = 214)
WPD <- WindPowerDensity(wind, dates = Dates, start = list(21, 4),
                        end = list(21, 6))
```

Index