Package 'FloodR'

January 22, 2021

Type Package

Title Separation of Flood Event and Event Precipitation

Version 0.6.0

Maintainer Philipp Buehler <philipp.buehler@rub.de>

Description
Flood event separation on the basis of daily and hourly mean discharges using a variance based threshold. The monthly maximum discharges are used as peak values. To define the event separation, first an interpolation algorithm based on Thiessen polygons is used to transform station precipitation into areal precipitation. With the areal precipitation, the event precipitation for each event can be defined using a change-point based approach applied to slope of the cummulative sums of the precipitation.

License GPL-3

Encoding UTF-8

Imports shiny, data.table, Rcpp, rgenoud, spatstat, partitions

Suggests ggplot2

Depends R (¿= 3.5)

LinkingTo Rcpp

LazyData true

RoxygenNote 7.1.1

R topics documented:

calc_stats *Calculating flood statistics for output in GUI and table*

Description

Internal function. Calculating flood statistics for output in GUI and table

Usage

```
calc_stats(Floods, Q)
```

Arguments

`Floods`	Flood table
`Q`	Discharge table

Author(s)

Philipp Bühler

create_voronoi *Creation of Voronoi polygons based on DWD-stations*

Description

For the region in question, Voronoi polygons are created based on coordinates of DWD-stations in and surrounding the region. For every day, they are created anew to consider every change in data availability.

Usage

```
create_voronoi(
  stations,
  z,
  Days,
  Station_list,
  limits_voronoi,
  coordinates,
  YY,
  All_Days,
  Area
)
```

Arguments

`stations`	A dataframe containing TRUE or FALSE values for every day and every DWD-station. If the station has registered a value, there is a ”TRUE”, otherwise there is a ”FALSE”.
`z`	A number that indicates the timestep of the year considered. If the data is interpolated for the years from 1900 to 2015, z ¡- 1 indicates the year 1900.
`Days`	An array containig the date of every day of the year considered.
`Station_list`	A dataframe containing the relevant DWD-stations' IDs, eastings, northings and heights. The Station-IDs must consist of 5 digits. If the ID is shorter than that it must be filled with zeros as leading digits. The column names must be ”Stations_id”, ”Easting”, ”Northing” and ”Height”.
`limits_voronoi`	A vector containing the minimal and maximal eastings and northings of the DWD-stations and the region in question.
`coordinates`	A dataframe containing every subdomain's coordinates. The column names must be ”Easting_gridcode” and ”Northing_gridcode” (”gridcode” must be replaced with the actual gridcode). Missing values indicated by NA-values are omitted during the procedure.
`YY`	An array containing all considered years.
`All_Days`	A sequence containing the date for every day within the period for which the Voronoi polygons are created.The days' format must be ”Y-m-d”.
`Area`	A dataframe containing all subdomain's names, gridcodes and heights. The column names must be ”names”, ”gridcode” and ”height”.

Value

A list with the following entries:

`names_over_all_days`	A list containing as many dataframes as there are subdomains. Every dataframe contains the relevant DWD-stations for the considered subdomain and the year in question.
`D`	A three-dimensional array containing every DWD-station's share for every day and every subdomain.

Author(s)

Laura Haendel

`eventsep` *Flood event separation based on variance*

Description

Based on daily mean discharges, a window-variance is calculated. If this variance exceeds a certain threshold, a flood event has been detected. To determine the begin of the flood event, the lag 1 differences are considered. The point, where the differences are positive for the last time before the variance exceeds the threshold, is defined as the starting point (including natural variability). The end of the flood event is defined as the point where the sum of the rising limb of the event equals approximately the sum of the falling limb.

Usage

```
eventsep(
  dailyMQ,
  monthlyHQ = NULL,
  dvar = 3,
  gamma = 1,
  theta = 0.25,
  ddur = 40,
  omega = 2,
  Kappa = 0.4,
  eta = 0.1,
  delta = 0.2,
  usemed = FALSE,
  medbf = 0.5,
  NA_mode = NULL,
  Messages = TRUE
)
```

Arguments

`dailyMQ`	A dataframe where the date is given as R-date or string (d.m.Y)) in the first and the daily mean discharges are given in the second column.
`monthlyHQ`	A dataframe where the date is given as R-date or string (d.m.Y) in the first and the monthly maximum discharge peaks are given in the second column.
`dvar`	integer: The window length of the moving variance in days. By default, it is set to three days.
`gamma`	numeric: Parameter for finding the start.
`theta`	numeric: The proportion of the sample variance of the dvar-day-window variance on the threshold. By default it is set to theta=0.25.
`ddur`	integer: The approximate minimum length of an overlaid event in days. By default it is set to ddur=40.
`omega`	integer: Threshold parameter for defining the end of a flood event. By default it is set to omega=2.
`Kappa`	numeric: Threshold parameter for defining the begin of a flood event. By default it is set to Kappa=0.4.
`eta`	numeric: Threshold parameter for defining the begin of a flood event. By default it is set to eta=0.1.
`delta`	numeric: Threshold parameter for defining the end of a flood event. By default it is set to delta=0.2.
`usemed`	logical: Should the median of relative difference between the baseflow at the end and the begin be considered to determine the end of the flood event.
`medbf`	numeric: The median of relative difference between the baseflow at the end and the begin of all separated flood events. Used to define the end of the event, if usemed=TRUE.
`NA_mode`	integer: If the timeseries does contain any NA values, NA_mode can be used to interpolate all gaps of continious NA-values by linear interpolation. Only gaps smaller or equal to NA_mode (¿=0) will be interpolated. Additionally, if the timeseries contains larger gaps, the timeseries

	is splited into subsamples with each being seperated intependend. The resulting separation table will still be append together.
`Messages`	logical: Should messages be thrown out

Details

To characterize the flood regime, it is necessary to estimate single events and to separate the amount of direct runoff from the total volume. For this purposes, a procedure is developed, which is based on the following three assumptions:

• A flood event is an exceedance of the normal discharge within a time span. For every flood event beginning and end has to be specified.

• A flood event is characterized by significantly increased discharge dynamics. Especially around the flood peaks, the sample variance of daily discharges within a moving time window will be higher than for other time periods with the same length.

• For all flood events, the volume of the rising limb of the hydrograph has to be equal the volume of the falling limb.

An assumed increase of base flow during the event has to be considered. The automatic event separation is based on the Lag one (1 day) differences of the daily mean discharges $QD_i = Q_i - Q_{i-1}$, with $QD_1 = 0$, and the sample variance of daily discharges within a moving window of dvar days:

$$Var_{d_{var}}(i) = Var(Q_{i-(d_{var}-1)}, \dots, Q_i)$$

$$Var_{d_{var}}(j) = 0, forj = 1, \dots, dvar.$$

The appropriated window length depends on the reaction time of the watershed and in this way from the catchment size. In our studies we modified it between 3 and 7 days, but 3 days were appropriated for nearly all catchments. To indicate the time span of the flood event, the sample variance has to exceed a certain threshold. We defined this threshold with the mean plus 0.25σ variance of all sample variances which could be derived within the long discharge series:

$$TH_{var} = \overline{V_{d_{var}}} + 0.25 \cdot \sqrt{Var(V_{d_{var}})}.$$

The peak is located within the time span, where the sample variance exceeds this threshold. The starting point of the flood event was defined as the time step in front of the peak for which the lag one-difference QD is negative for the last time:

$$t_{start} = \max\{1 \leq i < t_{peak} | QD_i < 0\}.$$

From this point on, discharges increase until the peak is reached. However, this increase does not always indicate the true start of the flood event. Instead, the natural variability in discharge can lead to an increase. To consider this, a temporal decrease between two discharges in succession of 10 percent was accepted. In such cases, the starting point was shifted further in the future ($Q_{t_{start}} = Q_{t_{start}+1}$) until the following condition was fulfilled:

$$|(Q_{t_{start}} - Q_{t_{start}+1})/Q_{t_{start}+1}| < 0.1.$$

The rising limb of a flood event is not steady, often pre-floods occured. We handled this by setting $t_{start} = t_{start} - 2$, if $(Q_{t_{start}-1} - Q_{t_{start}-2}) > 0.4(Q_{t_{peak}} - Q_{t_{start}})$. This condition ensures that the rising limb is included in total in the flood event. For the case where the rising limb of the two days before the peak has been larger than 40% of the actual limb we included these days in the flood event since they define a pre-flood. To define the end of the flood event we needed to find the time for which the differences between the volumes of the rising limb (SI) and the falling limb is approximately zero. Since also a possible increase of the baseflow during the flood event has to be considered, we define the end by

$$t_{end} = \min \left\{ t_{peak} + 1 \leq t \leq n | \left(\beta_{B;k} \leq median(\beta_B)\right) \vee \left(\left(\sum_{i=t_{peak}+1}^{t} QD_i \leq Q_{t_{start}} \right) \wedge \left(0.2 \left(SI - \sum_{i=t_{peak}+1}^{t} \right.\right.\right.\right.$$

where $\beta_{B;k}$ is the slope of the baseflow of the event. This includes two criteria: the volume of the falling limb is smaller than the baseflow at the beginning of the flood and the following two days the difference between the volume of the rising limb and the falling limb is not reduced by more than 20%. Alternatively, also the slope of the baseflow between start and end of the hydrograph is smaller than the median of the slope of the baseflow for all other flood events of the considered gauge. The number of peaks of the flood event then can be estimated by the number of time periods of exceedance of the variance threshold.

A special case that cannot be addresses directly by the procedure described above are flood events with multiple peaks. Here, we want to divide into two main cases: the superposition of two flood events and superimposed floods. In the first case, a second flood event begins before the recession of the first event has ended. This leads to an enhancement of the second event. The second case corresponds to a time period of an already increased baseflow, e.g. due to snowmelt. If flood events occur during this period, they are superimposed on this increased baseflow. Of course, not every flood event with multiple peaks belong to either of this cases. We define that a flood event consists potentially of superpositioned events of the originally separated event has a duration of less than 40 days. This parameter can be chosen according to the catchment size. Since the Mulde basin only consist of mesoscale catchments, this threshold is sufficient for our purpose. We then apply the independence criterion by Klein (2009) to validate if there are independent peaks within this event, i.e. two peaks - That deviate by less than five times the smaller peak - For which the larger peak is more than 2.5 times as large as the smallest discharge between both peaks - For which 70% of the smaller peak are larger than the smalles discharge between two peaks.

This criterion ensures that both peak are large enough to be separate flood event and that the recession of the first event is large enough to state them as independent. For all peaks that fulfill this criterion we choose the two peaks with the largest recession in between. According to this, two sub-events are defined, the first one with starting point equal to the original event begin. The end of this event is estimated by the lowest discharge between both peaks to which the recession of the original event is added (that are all discharge values smaller than the values of the valley) to reconstruct the overlaid recession of the first event. The second events begins with the smallest discharge value between both peaks and end with the original ending. Superimposed events can occur if the originally separated event has a duration larger than 40 days. Again, all peaks are checked for independence and are separated again. Since the separation of multiple peak events, especially of superimposed events is rather difficult, a manual check afterwards is recommended.

Value

A dataframe is returned with the following columns, where each row is a flood event:

`Begin`	date (d.m.Y) of the begin of the flood event
`End`	date (d.m.Y) of the end of the flood event
`Peak_date`	date (d.m.Y) of the maximum daily discharge during the flood event
`DailyMQ`	maximum daily mean discharge [m^3/s] during the flood event
`Volume`	volume [Mio. m^3] of the flood event calculated by the sum of all daily mean discharges during the flood event

dir_Volume	direct volume [Mio. m^3] calculated by the difference of the volume and the baseflow. The baseflow is estimated by a straight line between the discharge at the beginning and the end of the flood event.
baseflow_peak	baseflow [m^3/s] at the day of the flood peak, calculated by the straight-line method.
baseflow_begin	baseflow [m^3/s] at the beginning of the flood event, equal to the daily mean discharge of the beginning of the flood event.
baseflow_end	baseflow [m^3/s] at the end of the flood event, equal to the daily mean discharge of the end of the flood event.
No_peaks	number of peaks during the flood event
HQ	peak discharge [m^3/s] of the flood event taken from the monthly maximum discharges. If no monthly maximum discharge is available for the flood event, it is set to NA.
HQ_dir	direct peak discharge, claculated by the difference of HQ and baseflow_peak
comments	a short note if the event is overlaid, or first respectively second part of a double-peaked flood event.

Note

Important note: If a double-peaked or an overlaid event occurs, the whole event is listed in the output first, followed by the splitted partial events.

Author(s)

Svenja Fischer

References

Klein, B. (2009): Ermittlung von Ganglinien für die risikoorientierte Hochwasserbemessung von Talsperren. Schriftenreihe des Lehrstuhls Hydrologie, Wasserwirtschaft und Umwelttechnik, Ruhr-University Bochum.

Examples

```
## Not run:

dailyMQ<-data.frame(Date=seq(from=as.Date("01.01.2000", format="%d.%m.%Y"),
to=as.Date("01.01.2004", format="%d.%m.%Y"), by="days"),
discharge=rbeta(1462,2,20)*100)

monthlyHQ<-data.frame(Date=seq(from=as.Date("01.01.2000", format="%d.%m.%Y"),
to=as.Date("01.01.2004", format="%d.%m.%Y"), by="months"),
discharge=dailyMQ$discharge[(0:48)*12+1]+rnorm(49,5,1))

eventsep(dailyMQ, monthlyHQ)

## End(Not run)
```

fncols *Adds Column to dataframe if nonexcistend*

Description

Internal function. Adds Column to dataframe if nonexcistend

Usage

```
fncols(data, cname)
```

Arguments

`data`	data
`cname`	columnname to add

Author(s)

Philipp Bühler

make_typing_of_floods *Make a typology for a set of flood events*

Description

This functions uses already defined flood events with relevant characteristics (like precipitation, snow-melt and peak-value) for creating a typology for the whole set of flood events

Usage

```
make_typing_of_floods(
  Floods,
  n_G = 3L,
  fast_composition = FALSE,
  Type_3_min_samplesize = NULL,
  R_Seed = NULL
)
```

Arguments

`Floods`	data.frame or data.table: Table with Floods, including the columns
`n_G`	integer: Number of rain flood-types, defaults to 3
`fast_composition`	logical: use the partition package for fast combination of events
`Type_3_min_samplesize`	integer: NULL or interger specifying the minimum number of R3 events
`R_Seed`	integer: NULL or a R-Seed for reproductable flood typing

Author(s)

Philipp Bühler

min_max_coordinates — *Determine the minimal and maximal eastings and northings for a region and the surrounding DWD-stations*

Description

The coordinates of the subdomains of the specified region and the coordinates of the relevant DWD-stations are read. The minimal and maximal eastings and northings of all these coordinates are determined.

Usage

```
min_max_coordinates(Station_list, Area, coordinates)
```

Arguments

`Station_list`	A dataframe containing the relevant DWD-stations' IDs, eastings, northings and heights. The Station-IDs must consist of 5 digits. If the ID is shorter than that it must be filled with zeros as leading digits. The column names must be "Stations_id", "Easting", "Northing" and "Height".
`Area`	A dataframe containing all subdomain's names, gridcodes and heights. The column names must be "names", "gridcode" and "height".
`coordinates`	A dataframe containing every subdomain's coordinates. The column names must be "Easting_gridcode" and "Northing_gridcode" ("gridcode" must be replaced with the actual gridcode). Missing values indicated by NA-values are omitted during the procedure.

Value

`results`	A vector containing the minimal easting, the maximal easting, the minimal northing and the maximal northing in this order.

Author(s)

Laura Haendel

my.read.table — *reads table and covnerts string to date*

Description

Internal function. reads table and covnerts string to date

Usage

```
my.read.table(..., date.formats = c("%d.%m.%Y", "%Y-%m-%d"))
```

Arguments

`...`	as in read.table
`date.formats`	date.formats are preformatted data formats

Author(s)

Philipp Bühler

optimize_floodsep_parameters

Optimize the parameters of eventsep

Description

Optimize the parameters of eventsep by maximising flood events during high quantiles and minimizing flood events during low quantiles

Usage

```
optimize_floodsep_parameters(
  params,
  Discharge,
  upper_TH = 0.95,
  lower_TH = 0.5,
  lower_tolerance = 0.01,
  NA_mode = NULL
)
```

Arguments

params	Vector: Named vector of the stating parameters and their name, e.g. c("gamma"=1, "eta"=0.1)
Discharge	Data.frame: Discharge timeseries with Date amd Discharge column
upper_TH	numeric: Upper Threshold
lower_TH	numeric: Lower Threshold
lower_tolerance	numeric: Lower tolerance
NA_mode	integer: NULL or the length of the gap in which NA-values are interpolated

Author(s)

Philipp Bühler

`PKGENVIR` *Internal Environment. Environment for Shiny*

Description

Environment for Shiny

Environment for Shiny

Usage

```
PKGENVIR
```

Format

An object of class `environment` of length 0.

An object of class `environment` of length 0.

`PreconeCP` *Determine the begin of event precipitation*

Description

The cumulative precipitation is used to estimate the begin of the event precipitation using a change point estimation. For this, the time series is divided into two subsamples and the slope is estimated for each subsample using linear regression and least squares estimation. This is done for all possible combinations of two consecutive subsamples, such that the change point can be estimated as the point where the maximum difference between the slopes of the two subsamples occurs.

Usage

```
PreconeCP(X, indT, min_step = 3)
```

Arguments

`X`	A dataframe with the first column equal to the date (d.m.Y) and the second column consisting of the daily precipitation sums [mm].
`indT`	An integer vector with three entries: first the index of date in X of the begin of the flood event, second the index of date in X of the end of the flood event and third the index of the date in X of the peak of the flood event.
`min_step`	The minimum required number of data points used for the linear regression to determine the slope. At default it is set to 3 and it is recommended to use no shorter length.

Details

We want to estimated the event precipitation by comparing the rising limb of the discharge with the rising limb of precipitation. Here, we assumed that the begin of the increase of the hydrograph indicates the begin of the event precipitation, whereas the end of the flood event automatically defines the end of the event precipitation. Due to catchment reaction times, a possible delay of the begin of event precipitation and the begin of the flood event has to be considered. For this purpose a buffer time period b between the begin of the precipitation and the flood event was used. Here, b was defined as $b = \max(rt, 7)$, where rt is the duration of the rising limb of the flood event. Then the begin of event precipitation $hatk$ was defined by the change-point of the slope of the cumulative precipitation sum in this period:

$$\hat{k} = \arg\max_{t_{start}-b \leq k \leq t_{peak}+b} \{\beta_{k:t_{peak}+b} - \beta_{t_{start}-b:k}\},$$

where $\beta_{i:j}$ is the slope of the linear regression for the sum of all precipitation values $P_i, \ldots, P_j$ derived by least-squares method. This means, we divided the cumulative sums of the precipitation beginning b days before the begin of the flood event and ending b days after the day of the flood peak into two parts and used the time step where the difference of the slope of these two parts was maximal to define the begin of the precipitation. Each part had to consist of at least three days. The end of the event precipitation was defined equally to one day before the end of flood event.

Value

A date that indicates the estimated begin of the event precipitation.

Author(s)

Philipp Bühler, Svenja Fischer

Examples

```
## Not run:

dailyprec<-data.frame(Date=seq(from=as.Date("01.01.2000", format="%d.%m.%Y"),
to=as.Date("30.04.2000", format="%d.%m.%Y"), by="days"),
discharge=rbeta(121,2,20)*100)
indT<-c(15,30,14+which.max(dailyprec[15:30,2]))

PreconeCP(X=dailyprec,indT)

## End(Not run)
```

`PrectwoCP` *Determine the begin of event precipitation*

Description

The cumulative precipitation is used to estimate the begin of the event precipitation using a change point estimation. For this, the time series is divided into three subsamples and the slope is estimated for each subsample using linear regression and least squares estimation. This is done for all possible combinations of three consecutive subsamples, such that the change point can be estimated as the point where the maximum difference between the slopes of the first two subsamples occurs.

Usage

```
PrectwoCP(X, indT, s_p = 4, min_step = 3)
```

Arguments

`X`	A dataframe with the first column equal to the date (d.m.Y) and the second column consisting of the daily precipitation sums [mm].
`indT`	An integer vector with three entries: first the index of date in X of the begin of the flood event, second the index of date in X of the end of the flood event and third the index of the date in X of the peak of the flood event.
`s_p`	Integer that gives the number of days after the peak that are considered in the estimation of the slope, by default set to 4 days.
`min_step`	The minimum required number of data points used for the linear regression to determine the slope. By default it is set to 3 days and it is recommended to use no shorter length.

Details

Since pre-floods and pre-event precipitation often causes falsified starting point, the estimated begin with the one-change-point-method (function PreconeCP) is extended with a begin estimated by a twofold change-point consideration. For this, the precipitation time series is splitted into three sub-sequences. The first point where the cumulative precipitation is splitted has to be located before the flood peak and the second point after the peak. Then the points, for which the differences between the slopes of the first and the second as well as the second and the third part are maximal, are chosen as change points. The begin of the event precipitation then is defined as the first of these two change points.

Value

A date that indicates the estimated begin of the event precipitation.

Author(s)

Philipp Bühler

Examples

```
## Not run:

dailyprec<-data.frame(Date=seq(from=as.Date("01.01.2000", format="%d.%m.%Y"),
to=as.Date("30.04.2000", format="%d.%m.%Y"), by="days"),
discharge=rbeta(121,2,20)*100)
indT<-c(15,30,14+which.max(dailyprec[15:30,2]))

PrectwoCP(X=dailyprec,indT)

## End(Not run)
```

Run_WebFlood — *Graphical web interface for browsing trough floodevents in daily resolution and correcting them*

Description

This shiny app opens a graphical web interface, in which floodevents are shown with discharge and Precipitation. With the buttons you can browse trough all the floodevents supplemented in the flood file. The floods can be changed (begin and end) and the current properties are calculated. (see Details)

Usage

```
Run_WebFlood(
  Discharge = NULL,
  Precipitation = NULL,
  Catchment_Properties = NULL,
  language = "en"
)
```

Arguments

`Discharge`	[OPTIONAL]: A list of dataframes with the discharge data. Each dataframe must contain the Dischare data for a specific catchment with two columns: Column 1: Date in a continiuous daily timeseries (R "date" format) Column 2: Meassured Data of Discharge The list entries need to be named after the catchment-identifier (eg. name)
`Precipitation`	[OPTIONAL]: A list of dataframes with the precipitation data. Each dataframe must contain the precipitation data for a specific catchment with two columns: Column 1: Date in a continiuous daily timeseries (R "date" format) Column 2: Meassured Data of Precipitation The list entries need to be named after the catchment-identifier (eg. name)
`Catchment_Properties`	[OPTIONAL]: A dataframe with the properties for each catchment you want to use. This dataframe is only used to show information about the catchment. The dataframe must contain the following columns: Column "Name": Unique identifier (eg. Catchment Name) connecting the columns to the names of the list of discharge and precipitation. optional: Column "Area": Area size of the Catchments
`language`	The language for the graphical interface. Default is English ("en"), the other option is German: ("de"). Important: The table headers don't change and use english naming.

Details

This shiny app opens a graphical web interface, in which floodevents are shown with discharge and Precipitation. With the buttons you can browse trough all the floodevents supplemented in the flood file. The floods can be changed (begin and end) and the current properties are calculated. (see Details) The plotting output can be changed with a buffer. Uppon opening the interface, you need to open a ".csv"-file (";"-separator) which contains the flood information (begin,end and peak date). The naming convention of the file is "CATCHMENTNAME_your_comments.csv". It is important to have the exact catchment name before the underscore. After the underscore, anything can be written, followed by the .csv extension. The file needs the following columns:

Column "Begin": The date of the begin of the floodevent as sting (either in "

Column "End": The date of the end of the floodevent as sting (either in "

Column "Peak_date": The date of the peak of the floodevent as sting (either in "

optional:

Column "HQ": A value for the "true" peak of the flood, as if it was meassured in continious small timesteps. If the value is given, the direct HQ and the TQ value are computed

Author(s)

Philipp Buehler

Examples

```
## Not run:

#Run the dummy data:
Run_WebFlood()


#Run with own Data:
Discharge <-list(A=data.frame(Date=seq(from=as.Date("01.01.2000", format="%d.%m.%Y"),
                              to=as.Date("01.01.2004", format="%d.%m.%Y"), by="days"),
                      discharge=rbeta(1462,2,20)*100),
                 B=data.frame(Date=seq(from=as.Date("01.01.2000", format="%d.%m.%Y"),
                               to=as.Date("01.01.2004", format="%d.%m.%Y"), by="days"),
                             discharge=rbeta(1462,2,20)*100))

Precipitation <-list(A=data.frame(Date=seq(from=as.Date("01.01.2000", format="%d.%m.%Y"),
                               to=as.Date("01.01.2004", format="%d.%m.%Y"), by="days"),
                                 prec=rbeta(1462,2,20)*100),
                 B=data.frame(Date=seq(from=as.Date("01.01.2000", format="%d.%m.%Y"),
                               to=as.Date("01.01.2004", format="%d.%m.%Y"), by="days"),
                             prec=rbeta(1462,2,20)*100))


Catchment_Properties <-data.frame(Name=c("A","B"),Area=c(10,100),
Height=c(100,1000),stringsAsFactors = FALSE)
  language<-"en"

#Run_WebFlood(Discharge,Precipitation,Catchment_Properties)
#Use the provided file "A_Floodevents_example.csv" to test

## End(Not run)
```

Sample_Flood_events *Sample dataset of flood events*

Description

Sample dataset of flood events

Usage

```
data(Sample_Flood_events)
```

Format

An object of class `data.frame` with 149 rows and 10 columns.

Examples

```
data(Sample_Flood_events)
```

tr *Translates text in Shiny (German ¡-¿ English)*

Description

Internal Function. Translates text in Shiny (German ¡-¿ English)

Usage

```
tr(text)
```

Arguments

`text`	which text to translate

Author(s)

Philipp Bühler

`voronoiinterpolation` *Interpolation of precipitation or temperature data for a specified region using Voronoi-polygons*

Description

Precipitation or temperature data from surrounding DWD-stations are interpolated for the specified region to obtain areal values. Voronoi polygons are created, every polygon's share from the region is calculated and the precipitation or temperature data for the region is interpolated.

Usage

```
voronoiinterpolation(
  Area,
  All_Days,
  YY,
  parameter,
  Station_list,
  coordinates,
  stations,
  station_values
)
```

Arguments

`Area`	A dataframe containing all subdomain's names, gridcodes and heights. The column names must be "names", "gridcode" and "height".
`All_Days`	A sequence containing the date for every day within the period for which the data is interpolated. The days' format must be "Y-m-d".
`YY`	An array containing every year considered in the interpolation.
`parameter`	Character (either "prec" or "temp"), determines whether precipitation or temperature data is interpolated.
`Station_list`	A dataframe containing the relevant DWD-stations' IDs, eastings, northings and heights. The Station-IDs must consist of 5 digits. If the ID is shorter than that it must be filled with zeros as leading digits. The column names must be "Stations_id", "Easting", "Northing" and "Height".
`coordinates`	A dataframe containing every subdomain's coordinates. The column names must be "Easting_gridcode" and "Northing_gridcode" ("gridcode" must be replaced with the actual gridcode). Missing values indicated by NA-values are omitted during the procedure.
`stations`	A dataframe containing TRUE or FALSE values for every day and every DWD-station. If a precipitation or temperature value has been registered there is a "TRUE", otherwise there is a "FALSE".
`station_values`	A dataframe containing the precipitation or temperature data for every day and every relevant DWD-station. If a DWD-station did not register data for a period, the values must be set to zero.

Value

`interpolated_params`
: A matrix containing the interpolated precipitation or temperature data for every day for every subdomain.

Author(s)

Laura Haendel

Examples

```
## Not run:
Area <- data.frame("name"=c("Aue", "Niederschlema", "Golzern"),
"gridcode"=c(563790,562040,560020),"height" = c(2210, 685, 452))
coordinates <- data.frame("Easting_563790" = c(759830.4946,764830.4946,
769830.4946,759830.4946,764830.4946,769830.4946,759830.946,764830.4946,
769830.4946,759830.4946,764830.4946,769830.4946),
"Northing_563790" = c(5608061.8,5608061.8,5608061.8,5603061.8,5603061.8,
5603061.8,5598061.8,5598061.8,5598061.8,5593061.8,5593061.8,5593061.8),
"Easting_562040" = c(744830.4946,749830.4946,754830.4946,749830.4946,
754830.4946,744830.4946,749830.4946,754830.4946,744830.4946,749830.4946,
754830.4946,744830.4946),
"Northing_562040" = c(5608061.8,5608061.8,5608061.8,5603061.8,5603061.8,
5598061.8,5598061.8,5598061.8,5593061.8,5593061.8,5593061.8,5603061.8),
"Easting_560020" = c(754830.4946,754830.4946,754830.4946, 754830.4946,
759830.4946,764830.4946,759830.4946,764830.4946,759830.4946,764830.4946,
759830.4946,764830.4946),
"Northing_560020" = c(5608061.8,5603061.8,5598061.8,5593061.8,5608061.8,
5608061.8,5603061.8,5603061.8,5598061.8,5598061.8,5593061.8,5593061.8))
All_Days <- seq(from = as.Date("1900-01-01"), to = as.Date("1900-12-31"), by = 1)
YY <- unique(format(All_Days,"%Y"))
Station_list <- data.frame("Stations_id" = c("00438","00559","00840",
"01153","02038","02372","04506","04767"), "Easting" = c(767617.4963,
761110.605,756463.7461,755137.788,753573.4476,752950.0651,757558.097,
758533.0727),
"Northing" = c(5603716.394,5605149.476,5592820.687,5603864.269,
5607286.876,5605496.48,5610110.483,5599517.766),
"Height" = c(2451,586, 620,500,653,4764,663,658))
stations <- data.frame("00438" = unlist(list(rep(TRUE,365))),
"00559" = unlist(list(rep(TRUE,365))),
"00840" = unlist(list(rep(TRUE,365))),"01153" = unlist(list(rep(TRUE,365))),
"02038" = unlist(list(rep(TRUE,365))),
"02372" = unlist(list(rep(TRUE,365))),"04506" = unlist(list(rep(TRUE,365))),
"04767" = unlist(list(rep(TRUE,365))))
station_values <- data.frame("00438" = runif(365,min=0,max=10),
"00559" = runif(365,min=0,max=10), "00840" = runif(365,min=0,max=10),
"01153" = runif(365,min=0,max=10),"02038" = runif(365,min=0,max=10),
"02372" = runif(365,min=0,max=10),"04506" = runif(365,min=0,max=10),
"04767" = runif(365,min=0,max=10))

names(stations) <- c("00438", "00559", "00840", "01153",
"02038", "02372", "04506", "04767")
names(station_values)<- c("00438", "00559", "00840",
"01153", "02038", "02372", "04506", "04767")

parameter = "prec"
```

```
interpolated_values <- voronoiinterpolation(Area,All_Days,YY,parameter,Station_list,
coordinates,stations,station_values)

## End(Not run)
```

Index