Package ‘MultiHazard’

February 26, 2020

Title Tools for modeling compound events

Version 0.0.0.9000

Description This package provides tools for stationary multivariate statistical modeling. For instance to model the joint distribution of co-occurring hazards. The package contains functions for pre-processing data including imputing missing values, detrending and declustering time series as well as analyzing pairwise correlations over a range of lags. Functionality is also built in to conditionally sample a bivariate dataset (given one of the variables is above a predetermined threshold) and select the best fitting amongst an array of parametric (extreme and non-extreme, truncated and non-truncated) marginal distributions or copulas. Estimation of joint probability contours using the method of overlaying (conditional) contours given in Bender et al. (2016) and subsequently for a given return period extracting design events assuming full dependence, as well as the ``most likely'' or an ensemble of possible design events once accounting for dependence is possible. The package also provides the capability of fitting and simulating synthetic records from three higher dimensional approaches - standard (elliptic/Archimedean) copulas, Pair Copula Constructions (PCCs) and the conditional threshold exceedance approach of Heffernan and Tawn (2004). The package provides the code and data used in Jane et al. (2020), consequently applications in this vignette center around assessing the potential for compound flooding in South Florida.

License

Encoding UTF-8

LazyData true

Imports texmex,
fitdistrplus,
tweedie,
MASS,
VGAM,
copula,
GeneralizedHyperbolic,
statmod,
RColorBrewer,
VineCopula,
ks,
truncnorm,
dplyr

RoxygenNote 6.1.1

R topics documented:

`Con_Sampling_2D` *Conditionally sampling a two dimensional dataset*

Description

Creates a dataframe where the declustered excesses of a (conditioning) variable are paired with co-occurances of another varable.

Usage

```
Con_Sampling_2D(Data_Detrend, Data_Declust, Con_Variable, Thres = 0.97)
```

Arguments

`Data_Detrend` Dataframe containing two at least partially concurrent time series, detrended if necessary. Time steps must be equally spaced, with missing values assigned `NA`. First object may be a `"Date"` object. Can be `Dataframe_Combine` output.

`Data_Declust` Dataframe containing two (independently) declustered at least partially concurrent time series. Time steps must be equally spaced, with missing values assigned `NA`. Columns must be in the same order as in `Data_Detrend`. First object may be a `"Date"` object. Can be `Dataframe_Combine` output.

`Con_Variable` Column number (1 or 2) or the column name of the conditioning variable. Default is `1`.

`Thres` Threshold, as a quantile of the observations of the conditioning variable. Default is `0.97`.

Value

List comprising the specified `Threshold` as the quanitle of the conditioning variable above which declustered excesses are paired with co-occurances of the other varable, the resulting two dimensional sample `data` and `name` of the conditioning variable.

Examples

```
S20.Rainfall<-Con_Sampling_2D(Data_Detrend=S20.Detrend.df[,-c(1,4)],
                              Data_Declust=S20.Detrend.Declustered.df[,-c(1,4)],
                              Con_Variable="Rainfall",Thres=0.97)
```

`Copula_Threshold_2D` *Copula Selection With Threshold 2D - Fit*

Description

Declustered excesses of a (conditioning) variable are paired with co-occurances of the other varable before the best fitting bivariate copula is selected, using `BiCopSelect` function in the `VineCopula` package, for a single or range of thresholds. The procedure is automatically repeated with the variables switched.

Usage

```
Copula_Threshold_2D(Data_Detrend, Data_Declust, Thres = seq(0.9, 0.99,
  0.01), x_lim_min = min(Thres), x_lim_max = max(Thres),
  y_lim_min = -1, y_lim_max = 1, Upper = 0, Lower = 0,
  GAP = 0.05, Legend = TRUE)
```

Arguments

`Data_Detrend`	Dataframe containing two at least partially concurrent time series, detrended if necessary. Time steps must be equally spaced, with missing values assigned `NA`.
`Data_Declust`	Dataframe containing two (independently) declustered at least partially concurrent time series. Time steps must be equally spaced, with missing values assigned `NA`.
`Thres`	A single or sequence of thresholds, given as a quantile of the observations of the conditioning variable. Default, sequence from `0.9` to `0.99` at intervals of `0.01`.
`x_lim_min`	Numeric vector of length one specifying x-axis minimum. Default is the maximum argument in `Thres`.
`x_lim_max`	Numeric vector of length one specifying x-axis maximum. Default is the minimum argument in `Thres`.
`y_lim_min`	Numeric vector of length one specifying y-axis minimum. Default `-1.0`.
`y_lim_max`	Numeric vector of length one specifying y-axis maximum. Default `1.0`.
`Upper`	Numeric vector specifying the element number of the `Thres` argument for which the copula family name label to appear above the correponding point on the Kendall's tau coefficent vs threshold plot, when conditioin on the variable in column 1. Default is `0`.

Lower	Numeric vector specifying the element number of the Thres argument for which the copula family name label to appear below the correponding point on the Kendall's tau coefficent vs threshold plot, when conditioin on the variable in column 2. Default is 0.
GAP	Numeric vector of length one specifying the distance above or below the copula family name label appears the correponding point on the Kendall's tau coefficent vs threshold plot. Default is 0.05.
Legend	Logic vector of length one specifying whether a legend should be plotted. Default is TRUE.

Value

List comprising:

- Kendalls_Tau_Var1 Kendalls tau of a sample
- p_value_Var1 p-value when testing the null hypiothesis H_0 i.e. that there is no correlation between the variables
- N_Var1 size of the dataset
- Copula_Family_Var1 best fitting copula for the specified thresholds

when the dataset is conditioned on the variable in column 1. Anologous vector Kendalls_Tau_Var2,p_value_Var2, N_Var2 and Copula_Family_Var2 for the specified thresholds when the dataset is conditioned on the variable in column 2.

See Also

Dataframe_Combine

Examples

```
Copula_Threshold_2D(Data_Detrend=S20.Detrend.df[,-c(1,4)],
                    Data_Declust=S20.Detrend.Declustered.df[,-c(1,4)],
                    y_lim_min=-0.075, y_lim_max =0.25,
                    Upper=c(6,8), Lower=c(6,8),GAP=0.1)
```

Dataframe_Combine *Creates a dataframe containing up to five time series*

Description

Combines up to five time series, detrended where necessary, into a single dataframe.

Usage

```
Dataframe_Combine(data.1, data.2, data.3, data.4 = 0, data.5 = 0,
  n = 3, names)
```

Arguments

`n`	Integer `1-5` specifying the number of time series. Default is `3`.
`data.1:5`	Dataframes with two columns containing in column • 1 - Continuous sequence of times spaning from the first to the final recorded observations. • 2 - Corresponding values detrended where necessary.

Value

A dataframe containing all times from the first to the most up to date reading of any of the variables.

See Also

`Detrend`

Examples

```
#Formatting data
S20.Rainfall.df<-Perrine_df
S20.Rainfall.df$Date<-as.Date(S20.Rainfall.df$Date)
S20.OsWL.df<-S20_T_MAX_Daily_Completed_Detrend_Declustered[,c(2,4)]
S20.OsWL.df$Date<-as.Date(S20.OsWL.df$Date)
#Detrending O-sWL series at Site S20
S20.OsWL.Detrend<-Detrend(Data=S20.OsWL.df,Method = "window",PLOT=FALSE,
                          x_lab="Date",y_lab="O-sWL (ft NGVD 29)")
#Creating a dataframe with the date alongside the detrended OsWL series
S20.OsWL.Detrend.df<-data.frame(as.Date(S20.OsWL.df$Date),S20.OsWL.Detrend)
colnames(S20.OsWL.Detrend.df)<-c("Date","OsWL")
#Combining the two datasets by “Date” argument
S20.Detrend.df<-Dataframe_Combine(data.1<-S20.Rainfall.df,
                                  data.2<-S20.OsWL.Detrend.df,
                                  data.3=0,
                                  names=c("Rainfall","OsWL"))
```

`Decluster` *Declusters a time series*

Description

Identify cluster maxima above a threshold, using the runs method of Smith and Weissman (1994).

Usage

```
Decluster(Data, u = 0.95, SepCrit = 3, mu = 365.25)
```

Arguments

`Data`	Numeric vector of the time series.
`u`	Numeric vector of length one specifying the declustering threshold; as a quantile `[0,1]` of `Data` vector. Default is `0.95`.

SepCrit	Integer; specifying the separation criterian under which events are declustered. Default is 3 corresponding to a storm window of three days in the case of daily data.
mu	(average) Number of events per year. Numeric vector of length one. Default is 365.25, daily data.

Value

List comprising the Threshold above which cluster maxima are identifed, average number of declustered excesses per year EventsPerYear, a vector containing the origional time series Detrended and the Declustered series.

See Also

Detrend

Examples

```
Decluster(data=S20_T_MAX_Daily_Completed_Detrend$Detrend)
```

Design_Event_2D *Derives a single or ensemble of bivariate design events*

Description

Calculates the single design event under the assumption of full dependence, or once accounting for dependence between variables the single "most-likely" or an ensemble of possible design events.

Usage

```
Design_Event_2D(Data, Data_Con1, Data_Con2, Thres1, Thres2, Copula_Family1,
  Copula_Family2, Marginal_Dist1, Marginal_Dist2, Con1 = "Rainfall",
  Con2 = "OsWL", mu = 365.25, RP, x_lab = "Rainfall (mm)",
  y_lab = "O-sWL (mNGVD 29)", N, N_Ensemble)
```

Arguments

Data	Dataframe of dimension nx2 containing two co-occurring time series of length n.
Data_Con1	Dataframe containing the conditional sample (declustered excesses paired with concurrent values of other variable), conditioned on the variable in the first column.
Data_Con2	Dataframe containing the conditional sample (declustered excesses paired with concurrent values of other variable), conditioned on the variable in the second column. Can be obtained using the Con_Sampling_2D function.
Thres1	Numeric vector of length one specifying the threshold above which the variable in the first column was sampled in Data_Con1.
Thres2	Numeric vector of length one specifying the threshold above which the variable in the second column was sampled in Data_Con2.

- `Copula_Family1` Numeric vector of length one specifying the copula family used to model the `Data_Con1` dataset.
- `Copula_Family2` Numeric vector of length one specifying the copula family used to model the `Data_Con2` dataset. Best fitting of 40 copulas can be found using the `Copula_Threshold_2D` function.
- `Marginal_Dist1` Character vector of length one specifying (non-extreme) distribution used to model the marginal distribution of the non-conditioned variable.
- `Marginal_Dist2` Character vector of length one specifying (non-extreme) distribution used to model the marginal distribution of the non-conditioned variable. Best fitting among two truncted distributions or eight truncated distributions can be found using the functions.
- `Con2` Character vector of length one specifying the name of variable in the first column of `Data`.
- `mu` Numeric vector of length one specifying the (average) number of events per year. Default is `365.25`, daily data.
- `RP` Numeric vector of length one specifying the return period of interest.
- `x_lab` Charactor vector specifying the x-axis label.
- `y_lab` Charactor vector specifying the y-axis label.
- `N` Numeric vector of length one specifying the size of the sample from the fitted joint distributions used to estimate the density along an isoline. Samples are collected from the two joint distribution with proportions consistent with the total number of exteme events conditioned on each variable.
- `N_Ensemble` Numeric vector of length one specifying the number of possible design events sampled along the isoline of interest.
- `Con2` Character vector of length one specifying the name of variable in the second column of `Data`.

Value

Plot of all the observations (grey circles) as well as the declustered excesses above Thres1 (blue circles) or Thres2 (blue circles), observations may belong to both conditional samples. Also shown is the isoline associated with `RP` contoured according to their relative probability of occurrence on the basis of the sample from the two joint distributions, the "most likely" design event (black diamond), and design event under the assumption of full dependence (black triangle) are also shown in the plot. The function also returns a list comprising the design events assuming full dependence `"FullDependence"`, as well as once the dependence between the variables is accounted for the "Most likley" "MostLikelyEvent" as well as an "Ensemble" of possible design events.

See Also

`Dataframe_Combine` `Copula_Threshold_2D` `Diag_Non_Con` `Diag_Non_Con_Trunc`

Examples

```
S22.Rainfall<-Con_Sampling_2D(Data_Detrend=S22.Detrend.df[,-c(1,4)],
                              Data_Declust=S22.Detrend.Declustered.df[,-c(1,4)],
                              Con_Variable="Rainfall",Thres=0.97)
S22.OsWL<-Con_Sampling_2D(Data_Detrend=S22.Detrend.df[,-c(1,4)],
                          Data_Declust=S22.Detrend.Declustered.df[,-c(1,4)],
                          Con_Variable="OsWL",Thres=0.97)
```

```
S22.Copula.Rainfall<-Copula_Threshold_2D(Data_Detrend=S22.Detrend.df[,-c(1,4)],
                                 Data_Declust=S22.Detrend.Declustered.df[,-c(1,4)],Thres =0.97,
                                               y_lim_min=-0.075,y_lim_max=0.25,
                                     Upper=c(2,9),Lower=c(2,10),GAP=0.15)$Copula_Family_Var1
S22.Copula.OsWL<-Copula_Threshold_2D(Data_Detrend=S22.Detrend.df[,-c(1,4)],
                                 Data_Declust=S22.Detrend.Declustered.df[,-c(1,4)],Thres =0.97,
                                            y_lim_min=-0.075, y_lim_max =0.25,
                                     Upper=c(2,9),Lower=c(2,10),GAP=0.15)$Copula_Family_Var2
Design_Event_2D(Data=S22.Detrend.df[,-c(1,4)], Data_Con1=S22.Rainfall$Data,
                Data_Con2=S22.OsWL$Data, Thres1=0.97, Thres2=0.97,
                Copula_Family1=S22.Copula.Rainfall, Copula_Family2=S22.Copula.OsWL,
                Marginal_Dist1="Logis", Marginal_Dist2="Twe",RP=100,N=10,N_Ensemble=10)
```

`Detrend` *Detrends a time series.*

Description

Detrends a time series using either a linear fit covering the entire dataset or moving average trend correction with a user-sepcified window width.

Usage

```
Detrend(Data, Method = "window", Window_Width = 89,
  End_Length = 1826, PLOT = FALSE, x_lab = "Data", y_lab = "Data")
```

Arguments

`Data`	Dataframe containing two columns. In column: • 1 A `"Date"` object of equally spaced discrete time steps. • 2 Numeric vector containing corresponding time series values. No `NA`s allowed.
`Method`	Character vector of length one specifying approach used to detrend the data. Options are moving average `"window"` (default) and `"linear"`.
`Window_Width`	Numeric vector of length one specifying length of the moving average window. Default is `89`, window comprises the observation plus 44 days either side, which for daily data corresponds to an approximate 3 month window.
`End_Length`	Numeric vector of length one specifying number of observations at the end of the time series used to calculate the present day average. Default is `1826`, which for daily data corresponds to the final five years of observations.
`PLOT`	Logical; whether to plot origional and detrended series. Default is `"FALSE"`.
`x_lab`	Character vector of length one specifying x-axis label. Default is `"Date"`.
`y_lab`	Character vector of length one specifying y-axis label. Default is `"Data"`.

Value

Numeric vector of the detrended time series.

Examples

```
#Detrending ocean-side water level at site S22 using a 3 month moving average window and the last
#five years of observations to calculate the present day average.
Detrend(S22_T_MAX_Daily_Completed_Detrend,Method = "window",Window_Width= 89,
        End_Length = 1826, PLOT=FALSE,x_lab="Data",y_lab="Data")
```

Diag_Non_Con *Goodness of fit of non-extreme marginal distributions*

Description

Fits two (unbounded) non-extreme marginal distributions to a dataset and returns three plots demonstrating their relative goodness of fit.

Usage

```
Diag_Non_Con(Data, x_lab, y_lim_min = 0, y_lim_max = 1)
```

Arguments

Data	Numeric vector containing realizations of the variable of interest.
x_lab	Character vector of length one specifying the label on the x-axis of histogram and cummulative distribution plot.
y_lim_min	Numeric vector of length one specifying the lower y-axis limit of the histogram. Default is 0.
y_lim_max	Numericr vector of length one specifying the upper y-axis limit of the histogram. Default is 1.

Value

Panel consisting of three plots. Upper plot: Plot depicting the AIC of the two fitted distributions. Middle plot: Probabilty Density Functions (PDFs) of the fitted distribtions superimposed on a histgram of the data. Lower plot: Cummulaibre Distribution Functions (CDFs) of the fitted distributions overlaid on a plot of the empirical CDF.

See Also

Copula_Threshold_2D

Examples

```
S20.Rainfall<-Con_Sampling_2D(Data_Detrend=S20.Detrend.df[,-c(1,4)],
                              Data_Declust=S20.Detrend.Declustered.df[,-c(1,4)],
                              Con_Variable="Rainfall",Thres=0.97)
Diag_Non_Con(Data=S20.Rainfall$Data$OsWL,x_lab="O-sWL (ft NGVD 29)",
             y_lim_min=0,y_lim_max=1.5)
```

`Diag_Non_Con_Sel` *Demonstrate the goodness of fit of the slected non-extreme marginal distribution*

Description

Plots demonstrating the goodness of fit of a selected non-extreme marginal distribution to a dataset.

Usage

```
Diag_Non_Con_Sel(Data, x_lab = "Data", y_lim_min = 0, y_lim_max = 1,
  Selected)
```

Arguments

`Data`	Numeric vector containing realizations of the variable of interest.
`x_lab`	Numeric vector of length one specifyingLabel on the x-axis of histogram and cummulative distribution plot.
`y_lim_min`	Numeric vector of length one specifying the lower y-axis limit of the histogram.
`y_lim_max`	Numeric vector of length one specifying the upper y-axis limit of the histogram.
`Selected`	Charactor vector of length one specifying the chosen distribution, options are the Gaussian `"Gaus"` and logistic `"Logis"`.

Value

Panel consisting of three plots. Upper plot: Plots depicting the AIC of the two fitted distributions. Middle plot: Probabilty Density Functions (PDFs) of the `selected` distribtions superimposed on a histgram of the data. Lower plot: Cummulative distribution function (CDFs) of the `selected` distribution overlaid on a plot of the empirical CDF.

See Also

`Diag_Non_Con`

Examples

```
S20.Rainfall<-Con_Sampling_2D(Data_Detrend=S20.Detrend.df[,-c(1,4)],
                              Data_Declust=S20.Detrend.Declustered.df[,-c(1,4)],
                              Con_Variable="Rainfall",Thres=0.97)
Diag_Non_Con(Data=S20.Rainfall$Data$OsWL,x_lab="O-sWL (ft NGVD 29)",
             y_lim_min=0,y_lim_max=1.5)
Diag_Non_Con_Sel(Data=S20.Rainfall$Data$OsWL,x_lab="O-sWL (ft NGVD 29)",
                 y_lim_min=0,y_lim_max=1.5,Selected="Twe")
```

Diag_Non_Con_Trunc *Goodness of fit of non-extreme marginal distributions*

Description

Fits seven (tuncated) non-extreme marginal distributions to a dataset and returns three plots demonstrating their relative goodness of fit.

Usage

```
Diag_Non_Con_Trunc(Data, x_lab, y_lim_min = 0, y_lim_max = 1)
```

Arguments

Data	Numeric vector containing realizations of the variable of interest.
x_lab	Character vector of length one specifying the label on the x-axis of histogram and cummulative distribution plot.
y_lim_min	Numeric vector of length one specifying the lower y-axis limit of the histogram. Default is `0`.
y_lim_max	Numericr vector of length one specifying the upper y-axis limit of the histogram. Default is `1`.

Value

Panel consisting of three plots. Upper plot: Plot depicting the AIC of the eight fitted distributions. Middle plot: Probabilty Density Functions (PDFs) of the fitted distribtions superimposed on a histgram of the data. Lower plot: Cummulaibre Distribution Functions (CDFs) of the fitted distributions overlaid on a plot of the empirical CDF.

See Also

Copula_Threshold_2D

Examples

```
S20.OsWL<-Con_Sampling_2D(Data_Detrend=S20.Detrend.df[,-c(1,4)],
                          Data_Declust=S20.Detrend.Declustered.df[,-c(1,4)],
                          Con_Variable="OsWL",Thres=0.97)
Diag_Non_Con_Trunc(Data=S20.OsWL$Data$Rainfall,x_lab="Rainfall (Inches)",
                   y_lim_min=0,y_lim_max=2)
```

Diag_Non_Con_Trunc_Sel
Goodness of fit of non-extreme marginal distributions

Description

Fits eight non-extreme marginal distributions to a dataset and returns three plots demonstrating their relative goodness of fit.

Usage

```
Diag_Non_Con_Trunc_Sel(Data, x_lab, y_lim_min = 0, y_lim_max = 1,
  Selected)
```

Arguments

Data	Numeric vector containing realizations of the variable of interest.
x_lab	Character vector of length one specifying the label on the x-axis of histogram and cummulative distribution plot.
y_lim_min	Numeric vector of length one specifying the lower y-axis limit of the histogram. Default is `0`.
y_lim_max	Numericr vector of length one specifying the upper y-axis limit of the histogram. Default is `1`.
Selected	Charactor vector of length one specifying the chosen distribution, options are the Birnbaum-Saunders `"BS"`, exponential `"Exp"`, gamma `"Gam"`, lognormal `"LogN"`, Tweedie `"Twe"` and Weibull `"Weib"`.

Value

Panel consisting of three plots. Upper plot: Plot depicting the AIC of the eight fitted distributions. Middle plot: Probabilty Density Functions (PDFs) of the fitted distribtions superimposed on a histgram of the data. Lower plot: Cummulaibre Distribution Functions (CDFs) of the fitted distributions overlaid on a plot of the empirical CDF.

See Also

Diag_Non_Con_Trunc

Examples

```
S20.OsWL<-Con_Sampling_2D(Data_Detrend=S20.Detrend.df[,-c(1,4)],
                          Data_Declust=S20.Detrend.Declustered.df[,-c(1,4)],
                          Con_Variable="OsWL",Thres=0.97)
Diag_Non_Con_Trunc(Data=S20.OsWL$Data$Rainfall,x_lab="Rainfall (Inches)",
                   y_lim_min=0,y_lim_max=2)
Diag_Non_Con_Sel_Trunc(Data=S20.OsWL$Data$Rainfall,x_lab="Rainfall (Inches)",
                       y_lim_min=0,y_lim_max=2,Selected="Twe")
```

GPD_Fit *Fits a single generalised Pareto distribution - Fit*

Description

Fit a Generalized Pareto Distribution (GPD) to a declstered dataset.

Usage

```
GPD_Fit(Data, Data_Full, u = 0.95, PLOT = FALSE, xlab_hist = "Data",
  y_lab = "Data")
```

Arguments

Data	Numeric vector containing the declusted data.
Data_Full	Numeric vector containing the non-declustered data.
u	GPD threshold; as a quantile `[0,1]` of `Data` vector. Default is `0.95`.
xlab_hist	Character vector of length one. Histogram x-axis label. Default is `"Data"`.
y_lab	Character vector of length one. Histogram x-axis label. Default is `"Data"`.
Plot	Logical; indicating whether to plot diagnostics. Default is `FALSE`.

Value

List comprising the GPD `Threshold`, shape parameter `xi` and scale parameters `sigma` along with their standard errors `sigma.SE` and `xi.SE`.

Details

The fitted GPD model, is following parameterised as follows: $P(X > x|X > u)$

Examples

```
Decluster(Data=S20_T_MAX_Daily_Completed_Detrend$Detrend)
```

GPD_Parameter_Stability_Plot
GPD parameter stability plots

Description

Plots showing the stability of the GPD scale and shape parameter estimates across a specified range of thresholds.

Usage

```
GPD_Parameter_Stability_Plot(Data, Data_Full, u = 0.95, PLOT = FALSE,
  xlab_hist = "Data", y_lab = "Data")
```

Arguments

`Data`	Numeric vector containing the declusted data.
`Data_Full`	Numeric vector containing the non-declustered data.
`u`	Numeric vector of GPD thresholds; given as a quantiles `[0,1]` of `Data` vector. Default is `0.9` to `0.999` in intervals of `0.001`.
`Plot`	Logical; indicating whether to plot diagnostics. Default is `FALSE`.

Value

Plot of the shape and modified scale parameter estimates along with their errors bars over the range of specified thresholds.

See Also

`Decluster`

Examples

```
GPD_Parameter_Stability_Plot(Data = S20.Detrend.Declustered.df$Rainfall,
                             Data_Full= na.omit(S20.Detrend.df$Rainfall),
                             u=seq(0.9,0.999,0.001))
```

HT04 — *Fits and simulates from the conditional multivariate approach of Heffernan and Tawn (2004)*

Description

Fitting and simulating the conditional multivariate approach of Heffernan and Tawn (2004) to a dataset comprising 3 variables. Function utilizes the `mexDependence` and functions from the `texmex` package.

Usage

```
HT04(data_Detrend_Dependence_df, data_Detrend_Declustered_df, u_Dependence,
  Migpd, mu = 365.25, N = 100, Margins = "gumbel", V = 10,
  Maxit = 10000)
```

Arguments

`data_Detrend_Dependence_df`
: A dataframe with (n+1) columns, containing in column
  - 1 - Continuous sequence of dates spanning the first to the final time of any of the variables are recorded.
  - 2:(n+1) - Values, detrended where necessary, of the variables to be modelled.

`data_Detrend_Declustered_df`
: A dataframe with (n+1) columns, containing in column
  - 1 - Continuous sequence of dates spanning the first to the final time of any of the variables are recorded.

- 2:(n+1) - Declustered and if necessary detrended values of the variables to be modelled.

u_Dependence
: Dependence quantile. Specifies the (sub-sample of) data to which the dependence model is fitted, that for which the conditioning variable exceeds the threshold associated with the presecribed quantile. Default is `0.7`, thus the dependence parameters are estimated using the data with the highest `30%` of values of the conditioning variables.

Migpd
: An `Migpd` object, containing the generalised Pareto models fitted (independently) to each of the variables.

Margins
: Character vector specifying the form of margins to which the data are transformed for carrying out dependence estimation. Default is `"gumbel"`, alternative is `"laplace"`. Under Gumbel margins, the estimated parameters `a` and `b` describe only positive dependence, while `c` and `d` describe negative dependence in this case. For Laplace margins, only parameters `a` and `b` are estimated as these capture both positive and negative dependence.

V
: See documentation for mexDependence.

Maxit
: See documentation for mexDependence.

Value

List comprising the fitted HT04 models `Models`, proportion of the time each variable is most extreme, given at least one variable is extreme `Prop`, as well as the simulated values on the transformed `u.sim` and original x.sim scales.

See Also

`Dataframe_Combine` `Migpd_Fit`

Examples

```
#Fitting and simulating from the Heffernan and Tawn (2004) model
S20.HT04<-HT04(data_Detrend_Dependence_df=S20.Detrend.df,
               data_Detrend_Declustered_df=S20.Detrend.Declustered.df,
               u_Dependence=0.995,Migpd=S20.Migpd,mu=365.25,N=1000)
#View model conditioning on rainfall
S20.HT04$Model$Rainfall
#Assigning simulations (transfomed back to the origional scale) a name
S20.HT04.Sim<-S20.HT04$x.sim
#Plotting observed (black) and simulated (red) values
S20.Pairs.Plot.Data<-data.frame(rbind(na.omit(S20.Detrend.df[,-1]),S20.HT04.Sim),
                                c(rep("Observation",nrow(na.omit(S20.Detrend.df))),
                                  rep("Simulation",nrow(S20.HT04.Sim))))
colnames(S20.Pairs.Plot.Data)<-c(names(S20.Detrend.df)[-1],"Type")
pairs(S20.Pairs.Plot.Data[,1:3],
      col=ifelse(S20.Pairs.Plot.Data$Type=="Observation","Black","Red"),
      upper.panel=NULL,pch=16)
```

Imputation *Imputing missing values through linear regression*

Description

Fits a simple linear regression model, impute missing values of the dependent variable.

Usage

```
Imputation(Data, Variable, x_lab, y_lab)
```

Arguments

`Data`	Dataframe containing two at least partially concurrent time series. First column may be a `"Date"` object. Can be `Dataframe_Combine` output.
`Variable`	Character vector of length one specifying the (column) name of the varible to be imputed i.e. dependent variable in the fitted regression.
`x_lab`	Character vector of length one specifying the name of the independent variable to appear as the x-axis label on a plot showing the data, imputed values and the linear regression model.
`y_lab`	Character vector of length one specifying the name of the dependent variable to appear as the y-axis label on plot showing the data, imputed values and the linear regression model.

Value

List comprising

- `Data` dataframe containing the origional data plus an additional column named `Value` where the `NA` values of the `Variable` of interest have been imputed where possible.
- `Model` linear regression model paramters including its coefficient of determination

and a scatter plot of the data (black points), linear regression model (red line) and fitted (imputed) values (blue points).

Examples

```
####Objective: Fill in missing values at groundwater well G_3356 using record at G_3355
##Viewing first few rows of G_3356
head(G_3356)
#Converting date column to a "Date" object
G_3356$Date<-seq(as.Date("1985-10-23"), as.Date("2019-05-29"), by="day")
#Converting readings to numeric object
G_3356$Value<-as.numeric(as.character(G_3356$Value))

##Viewing first few rows of G_3355
head(G_3355)
#Converting date column to a "Date" object
G_3355$Date<-seq(as.Date("1985-08-20"), as.Date("2019-06-02"), by="day")
#Converting readings to numeric object
G_3355$Value<-as.numeric(as.character(G_3355$Value))
```

```
##Merge the two dataframes by date
library('dplyr')
GW_S20<-merge(G_3356,G_3355,by="Date")
colnames(GW_S20)<-c("Date","G3356","G3355")
#Carrying out imputation
Imputation(Data=GW_S20,Variable="G3356",
           x_lab="Groundwater level (ft NGVD 29)",
           y_lab="Groundwater level (ft NGVD 29)")
```

Kendall_Lag *Kendall's tau correlation coefficient between pairs of variables over a range of lags*

Description

Kendall's tau correlation coefficient between pairs of variables over a range of lags

Usage

```
Kendall_Lag(Data, Lags = seq(-6, 6, 1), PLOT = TRUE, GAP = 0.1)
```

Arguments

Data	A data frame with 3 columns, containing concurrent observations of three time series.
Lags	Integer vector giving the lags over which to calculate coefficient. Default is a vector from -6 to 6.
GAP	Numeric vector of length one. Length of y-axis above and below max and min Kendall's tau values.
Plot	Logical; whether to show plot of Kendall's coefficient vs lag. Default is TRUE.

Value

List comprising Kendall's tau coefficients between the variables pairs composing columns of Data with the specified lags applied to the second named variable `Values` and the p-values `Test` when testing the null hypothesis H_0: tau=0 i.e. there is no correlation between a pair of variables. Plot of the coefficient with a filled point of hypothesis test (p-value<0.05). Lag applied to variable named second in the legend.

See Also

Dataframe_Combine

Examples

```
Kendall_Lag(Data=S20.Detrend.df,GAP=0.1)
```

Mean_Excess_Plot *Mean excess plot - GPD threshold selection*

Description

The empirical mean excess function is linear in the case of a GPD.

Usage

```
Mean_Excess_Plot(Data)
```

Arguments

data	A vector comprising a declustered and if necessary detrended time series to be modelled.

Value

Plot of the empirical mean excess function (black line), average of all observations exceeding a threshold decreased by the threshold, for thresholds spanning the range of the observations. Also provided are 95% confidence intervals (blue dotted lines) and the observations (black dots).

See Also

Decluster Detrend

Examples

```
Mean_Excess_Plot(Data=S20_Detrend_Declustered_df$Rainfall)
```

Migpd_Fit *Fits Multiple independent generalized Pareto models - Fit*

Description

Fit multiple independent generalized Pareto models to each column of a dataframe. Edited version of the migpd function in texmex, to alllow for NAs in a time series.

Usage

```
Migpd_Fit(Data, mth, mqu, penalty = "gaussian", maxit = 10000,
  trace = 0, verbose = FALSE, priorParameters = NULL)
```

Arguments

`Data`	A dataframe with n columns, each comprising a declustered and if necessary detrended time series to be modelled.
`mth`	Marginal thresholds, above which generalized Pareto models are fitted. Numeric vector of length n.
`mqu`	Marginal quantiles, above which generalized Pareto models are fitted. **Only one of `mth` and `mqu` should be supplied.** Numeric vector of length n.
`penalty`	See ggplot.migpd.
`maxit`	See ggplot.migpd.
`trace`	See ggplot.migpd.
`verbose`	See ggplot.migpd.
`priorParameters`	See ggplot.migpd.

Value

An object of class "migpd". There are `coef`, `print`, `plot`, `ggplot` and `summary` functions available.

See Also

Decluster Detrend Dataframe_Combine

Examples

```
#With date as first column
S22.GPD<-Migpd_Fit(Data=S22.Detrend.Declustered.df, mqu =c(0.99,0.99,0.99))
#Without date as first column
S22.GPD<-Migpd_Fit(Data=S22.Detrend.Declustered.df[,-1], mqu =c(0.99,0.99,0.99))
```

NOAA_SLR *NOAA sea-level rise scenarios*

Description

Time (in years) for a specified amount of sea-level rise (SLR) to occur at Maimi Beach according to the five SLR scenarios in NOAA 2017 report titled "Global and Regional Sea Level Rise Scenarios for the United States".

Usage

```
NOAA_SLR(OsWL_req, SLR_scen = c("High", "Intermediate", "Low"),
  Input_unit = "m", Year.Inital = 2020)
```

Arguments

`OsWL_req`	Numeric vector of SLR required.
`SLR_scen`	Character vector specifying which of the NOAA (2017) scenarios to consider. Options include `High`, Intermediate high `Int.High`, `Intermediate`, Intermediate low (`Int.Low`) and `Low`.
`Input_unit`	Character vector of length one; specifying units of SLR. Default is meters "m", other option is feet "ft".
`Year`	Character vector of length one; specifying

Value

List comprising the specified `Threshold` as the quanitle of the conditioning variable above which declustered excesses are paired with co-occurances of the other varable, the resulting two dimensional sample `data` and `name` of the conditioning variable.

Examples

```
NOAA_SLR<-function(OsWL_req=seq(0,1,0.01),SLR_scen = c("High","Intermediate","Low"),Input_unit="m")
```

`SLR_Scenarios` *Sea level rise scenarios in the Southeast Florida Regional Climate Change Compact:*

Description

Calculates and plots time required for sea level rise to reach a specified level according to the three scenarios in the Compact.

Usage

```
SLR_Scenarios(SeaLevelRise, Unit = "m")
```

Arguments

`SeaLevelRise`	Numeric vector of length one, sea level rise required.
`data`	A dataframe with `n` columns, each comprising a declustered and if necessary detrended time series to be modelled.

Value

An object of class `"migpd"`. There are `coef`, `print`, `plot`, `ggplot` and `summary` functions available.

Examples

```
SLRScenarios(0.45)
```

`Standard_Copula_Fit` *Fit an Archimedean/elliptic copula model - Fit*

Description

Fit a n-dimensional Archimedean or elliptic copula model. Function is simply a repackaging of the `fitCopula` function in the `copula` package.

Usage

```
Standard_Copula_Fit(Data, Copula_Type = "Gaussian")
```

Arguments

Data	Dataframe containing n at least partially concurrent time series. First column may be a "Date" object. Can be Dataframe_Combine output.
Copula_Type	Type of elliptical copula to be fitted, options are "Gaussian" (Default), "tcopula", "Gumbel", "Clayton" and "Frank".

Value

List comprising the Copula_Type and the fitted copula Model object.

See Also

Dataframe_Combine Standard_Copula_Sel CDVineCopSelect BiCopSelect

Examples

```
cop<-Standard_Copula_Fit(Data=S20.Detrend.df,Copula_Type="Gaussian")
cop<-Standard_Copula_Fit(Data=S20.Detrend.df,Copula_Type="tcopula")
cop<-Standard_Copula_Fit(Data=S20.Detrend.df,Copula_Type="Gumbel")
cop<-Standard_Copula_Fit(Data=S20.Detrend.df,Copula_Type="Clayton")
cop<-Standard_Copula_Fit(Data=S20.Detrend.df,Copula_Type="Frank")
```

Standard_Copula_Sel *Selecting best fitting standard (elliptical and Archimeadean) copula*

Description

Fits five n-dimensional standard copula to a dataset and returns their corresponding AIC values.

Usage

```
Standard_Copula_Sel(Data)
```

Arguments

Data	Data frame containing n at least partially concurrent time series, detrended if necessary. Time steps must be equally spaced, with missing values assigned NA. First object may be a "Date" object. Can be Dataframe_Combine output.

Value

Data frame containing copula name in column 1 and associated AIC in column 2. Parameters are estimated using the fitCopula() function in copula package using maximum pseudo-likelihood estimator "mpl". See fitCopula for a more thorough explanation.

See Also

Dataframe_Combine Standard_Copula_Fit

Examples

```
Standard_Copula_Sel(Data_Detrend=S20.Detrend.df)
```

`Standard_Copula_Sim` *Archimedean/elliptic copula model - Simulation*

Description

Simulating from a fitted Archimedean or elliptic copula Model.

Usage

```
Standard_Copula_Sim(Data, Marginals, Copula, mu = 365.25, N = 10000)
```

Arguments

`Data`	Dataframe containing `n` at least partially concurrent time series. First column may be a `"Date"` object. Can be `Dataframe_Combine` output.
`Marginals`	An `migpd` object containing the n-independent generalized Pareto models.
`Copula`	An Archimedean or elliptic copula model. Can be specified as an `Standard_Copula_Fit` object.
`mu`	(average) Number of events per year. Numeric vector of length one. Default is 365.25, daily data.
`N`	Number of years worth of extremes to be simulated. Numeric vector of length one. Default 10,000 (years).

Value

Each n-dimensional realisation is given on the transformed `[0,1]^n` scale (first n columns) in the first dataframe `u.Sim` and on the origional scale in the second dataframe `x.Sim`.

See Also

`Standard_Copula_Sel` `Standard_Copula_Fit`

Examples

```
#Fitting multiple independent GPDs to the data
#(required to transform realisation back to origional scale)
S20.Migpd<-Migpd_Fit(Data=S20.Detrend.Declustered.df[,-1],mqu=c(0.975,0.975,0.9676))
#Fitting Gaussian copula
Standard_Copula_Sim(Data=S20.Detrend.df,Marginals=S20.Migpd,Copula=S20.Gaussian,
                    mu=365.25,N=10000)
```

Vine_Copula_Fit *C and D-vine Copula - Fitting*

Description

Fit either a C- or D-vine copula model. Function is a repackaging of the `CDVineCopSelect` function in the `CDVine` package.

Usage

```
Vine_Copula_Fit(Data, FamilySet = NA, Type = "DVine",
  SelCrit = "AIC", Indeptest = FALSE, Level = 0.05)
```

Arguments

`Data`	Dataframe containing `n` at least partially concurrent time series. First column may be a `"Date"` object. Can be `Dataframe_Combine` output.
`FamilySet`	Integer vector which must include at least one pair-copula family that allows for positive and one that allows for negative dependence. If `familyset = NA` (default), selection among all possible families is performed. The coding of pair-copula families is shown below. See help file of the `CDVineSim` function to find out copula represented by integers `0-40`.
`Type`	Type of the vine model: • 1 or "CVine" = C-vine • 2 or "DVine" = D-vine
`SelCrit`	Character vector specifying the criterion for choosing among the competing pair-copula. Possible choices: `"AIC"` (default) or `"BIC"`.
`Indeptest`	Logical; whether a hypothesis test for the independence of `u1` and `u2` is performed before bivariate copula selection (default: `Indeptest = FALSE`; cp. BiCopIndTest). The independence copula is chosen for a (conditional) pair if the null hypothesis of independence cannot be rejected.
`level`	Numeric; significance level of the independence test (default: level = 0.05).

Value

List comprising the pair-copula families composing the C- or D-vine copula `Family`, its parameters `Par` and `Par2` as well as whether it is a C or D-vine `Type`.

See Also

`Dataframe_Combine` `Vine_Copula_Sim` `CDVineCopSelect` `BiCopSelect`

Examples

```
S20.Vine<-Vine_Copula_Fit(Data=S20.Detrend.df, FamilySet=NA,
                          Type="DVine", SelCrit="AIC",
                          Indeptest=FALSE, Level=0.05)
```

`Vine_Copula_Sim` *C and D-vine Copula - Simulation*

Description

Simulating from specified C- and D-vine copula models. Builds on the `CDVineSim` in `CDVine`.

Usage

```
Vine_Copula_Sim(Data, Marginals, Vine_family, Vine_par, Vine_par2,
  Vine_Type = "DVine", mu = 365.25, N = 10000)
```

Arguments

`Data`	Dataframe containing `n` at least partially concurrent time series. First column may be a `"Date"` object. Can be `Dataframe_Combine` output.
`Marginals`	An `migpd` object containing the d-independent generalized Pareto models.
`Vine_family`	A n*(n-1)/2 integer vector specifying the pair-copula families defining the fitting C- or a D-vine copula models. Can be specified as the `Family` agument of a `Vine_Copula_Fit` object. See help file of the `CDVineSim` function to find out copula represented by integers 0-40.
`Vine_par`	A n*(n-1)/2 vector of pair-copula parameters.
`Vine_par2`	A n*(n-1)/2 vector of second parameters for pair-copula families with two parameters.
`Vine_Type`	Type of the vine model: • 1 or "CVine" = C-vine • 2 or "DVine" = D-vine Can be specified as the `Type` argument of a `Vine_Copula_Fit` object.
`mu`	(average) Number of events per year. Numeric vector of length one. Default is 365.25, daily data.
`N`	Number of years worth of extremes to be simulated. Numeric vector of length one. Default 10,000 (years).

Value

List comprising an integer vector specifing the pair-copula families composing the C- or D-vine copula `Vine_family`, its paraeters `Vine_par` and `Vine_par2` and type of regular vine `Vine_Type`. In addition, dataframes of the simulated observations: `u.Sim` on the transformed `$[0,1]^n$` and `x.Sim` the origional scales.

See Also

`Vine_Copula_Fit`

Examples

```
#Fitting vine copula
S20.Vine<-Vine_Copula_Fit(Data=S20.Detrend.df, FamilySet=NA,
                          Type="DVine", SelCrit="AIC",Indeptest=FALSE,
                          Level=0.05)
#Simulating from fitted copula
S20.Vine.Sim<-Vine_Copula_Sim(Data=S20.Detrend.df,Marginals=S20.Migpd,
                              Vine_family=S20.Vine$Family, Vine_par=S20.Vine$Par,
                              Vine_par2=S20.Vine$Par2, Vine_Type="DVine",N=10)
#Plotting observed (black) and simulated (red) values
S20.Pairs.Plot.Data<-data.frame(rbind(na.omit(S20.Detrend.df[,-1]),S22.Vine.Sim$x.Sim),
                                c(rep("Observation",nrow(na.omit(S20.Detrend.df))),
                                rep("Simulation",nrow(S20.Vine.Sim$x.Sim))))
colnames(S20.Pairs.Plot.Data)<-c(names(S20.Detrend.df)[-1],"Type")
pairs(S20.Pairs.Plot.Data[,1:3],
      col=ifelse(S20.Pairs.Plot.Data$Type=="Observation","Black","Red"),
      upper.panel=NULL)
```

Index