Package 'FloodMetrics'

August 23, 2023

Title Flood Metrics Analysis Tools

Version 0.0.0.9000

Description Provides tools and functions for analyzing flood metrics, including frequency analysis, statistical tests, stochastical analysis and visualizations. Aims to bridge the gap between theoretical research and practical application by providing a one-stop solution for flood analysis.

Imports Kendall,
actuar,
extRemes,
fitdistrplus,
forecast,
trend,
tseries,
zoo,
stats,
lfstat,
nortest,
dplyr,
eva,
lubridate,
VineCopula,
graphics

License MIT + file LICENSE

Encoding UTF-8

Roxygen list(markdown = TRUE)

RoxygenNote 7.2.3

R topics documented:

`analyzeFloodMetrics` *Analyze Flood Metrics*

Description

This function analyzes flood metrics based on the provided flood flow data.

Usage

```
analyzeFloodMetrics(FldFl, i, Min_AP, percentile)
```

Arguments

`FldFl`	A dataframe containing the flood flow data.
`i`	An identifier or index.
`Min_AP`	Minimum Annual Precipitation for Annual Maximum Series.
`percentile`	The ratio used in the percentile calculation.

Value

A dataframe with analyzed flood metrics.

Examples

```
## Not run:
# Assuming you have a dataframe 'FldFl' with flood flow data, an identifier 'i', Min_AP, and percentile:
result_df <- analyzeFloodMetrics(FldFl, i, Min_AP, percentile)

## End(Not run)
```

bivariateCopulaWithBestFit *Bivariate Copula Analysis with Best Fit Distributions*

Description

This function fits two datasets to multiple distributions, selects the best distribution based on AIC, and then performs a bivariate copula analysis.

Usage

```
bivariateCopulaWithBestFit(X1, X2)
```

Arguments

X1	A numeric vector representing the first data set.
X2	A numeric vector representing the second data set.

Value

A list containing the results of the copula analysis.

Examples

```
## Not run:
# Assuming you have two vectors 'X1' and 'X2' and a function 'fitMultipleDistributions':
result <- bivariateCopulaWithBestFit(X1, X2, fitMultipleDistributions)

## End(Not run)
```

compareDistributions *Compare Best-Fit and GEV Distributions*

Description

This function compares the best-fit and GEV distributions using the KS test and plots their CDFs.

Usage

```
compareDistributions(
  sample1_properties,
  sample2_properties,
  p = c(0.05, seq(0.1, 0.9, 0.005), 0.95),
  output_filename = "CDF_comparison.svg"
)
```

Arguments

sample1_properties	A named list containing the properties of the first sample (distribution and its parameters).
sample2_properties	A named list containing the properties of the second sample (distribution and its parameters).
p	A numeric vector of percentiles.
output_filename	The filename for the SVG output plot.

Value

A dataframe containing the KS test statistic and p-value.

---

cumulativeDensityFunction

Cumulative Density of Specified Distribution

Description

This function computes the cumulative density value for a given distribution and parameters.

Usage

```
cumulativeDensityFunction(dbn, P, shp, sc)
```

Arguments

dbn	A character string specifying the distribution. Valid options are "norm", "lnorm", "gamma", "weibull", and "llogis".
P	A numeric value or vector for which the cumulative density is to be computed.
shp	Shape parameter for the distribution.
sc	Scale parameter for the distribution.

Value

A numeric value or vector representing the cumulative density for the specified distribution and parameters.

Examples

```
## Not run:
cumulativeDensityFunction("norm", 0, 0, 1)
cumulativeDensityFunction("gamma", 1, 2, 1)

## End(Not run)
```

extractFitStatistics *Fit Statistics Extraction*

Description

Function used to obtain coefficients and Metrics for each distribution

Takes the object fitted using fitdistrplus

Extracts - Shape, Scale, SDShape, SDscale, Loglikelihood and AIC

Usage

```
extractFitStatistics(obj)
```

Arguments

obj	Object that is obtained after fitting a set of data to fitdist function offitdistplus library

Value

Shape, Scale, SDShape, SDscale, Loglikelihood and AIC of the distribution

Examples

```
## Not run:
# Generate some sample data
data <- rnorm(100)

# Fit the data using fitdist from the fitdistrplus package
fit <- fitdistrplus::fitdist(data, "norm")

# Extract fit statistics
extractFitStatistics(fit)

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

extractPeakFlow *Extract Peak Flow and Time to Peak*

Description

This function extracts the peak flow and the corresponding time to peak from a given dataframe.

Usage

```
extractPeakFlow(df)
```

Arguments

df	A dataframe containing the streamflow data with columns 'Floodidx', 'FloodQ', and 'Date'.

Value

A dataframe with peak flows and their corresponding dates.

Examples

```
## Not run:
# Assuming you have a dataframe 'df' with streamflow data:
result_df <- extractPeakFlow(df)

## End(Not run)
```

`fitAndTestDistributions` *Fit Multiple Distributions and Test Goodness of Fit*

Description

This function fits multiple distributions to peak flow data, computes empirical probabilities, and tests the goodness of fit using the KS and AD tests.

Usage

```
fitAndTestDistributions(Stn, StnID)
```

Arguments

`Stn`	A dataframe containing the peak flow data in a column named 'peak_va'.
`StnID`	A numeric or character value representing the station ID.

Value

A dataframe containing fit parameters and test statistics for each distribution.

Examples

```
## Not run:
# Assuming you have a dataframe 'Stn' and a station ID '08041500':
fitAndTestDistributions(Stn, '08041500')

## End(Not run)
```

`fitDistTestGOF` *Fit Distributions and Test Goodness of Fit*

Description

This function fits multiple distributions to peak flow data, computes empirical probabilities, and tests the goodness of fit using the KS and AD tests.

Usage

```
fitDistTestGOF(Stn, StnID)
```

Arguments

`Stn`	A dataframe containing the peak flow data in a column named 'peak_va'.
`StnID`	A numeric or character value representing the station ID.

Value

A dataframe containing fit parameters and test statistics for each distribution.

Examples

```
## Not run:
# Assuming you have a dataframe 'Stn' and a station ID '08041500':
fit_results <- fitDistTestGOF(Stn, '08041500')

## End(Not run)
```

`fitExpAndGammaDistributions` *Fit Exponential and Gamma Distributions to Data*

Description

This function fits both exponential and gamma distributions to the provided data using Maximum Likelihood Estimation (MLE).

Usage

```
fitExpAndGammaDistributions(X)
```

Arguments

`X`	A numeric vector of data to be fitted into the distributions.

Value

A data frame containing the fit statistics for each distribution, including their shape, scale, standard errors for shape and scale, Log-likelihood, and AIC value.

Examples

```
## Not run:
fitExpAndGammaDistributions(data)

## End(Not run)
```

`fitExtremeDistributions` *Fit Extreme Distributions (GEV and Gumbel) to Peak Flow Data*

Description

This function fits the Generalized Extreme Value (GEV) and Gumbel distributions to peak flow data from a given station.

Usage

```
fitExtremeDistributions(Stn)
```

Arguments

`Stn`	A dataframe containing the peak flow data in a column named 'peak_va'.

Value

A dataframe with fitted parameters for the GEV and Gumbel distributions.

Examples

```
## Not run:
# Assuming you have a dataframe 'Stn' with peak flow data:
results <- fitExtremeDistributions(Stn)

## End(Not run)
```

`fitMultipleDistributions` *Fit Multiple Distributions to Data*

Description

This function fits five candidate distributions to the provided data using Maximum Likelihood Estimation (MLE). The distributions fitted are: normal, lognormal, weibull, gamma, and logistic.

Usage

```
fitMultipleDistributions(X)
```

Arguments

X	A numeric vector of data to be fitted into various distributions.

Value

A data frame containing the fit statistics for each distribution, including their shape, scale, Log-likelihood, and AIC value.

Examples

```
## Not run:
fitMultipleDistributions(data)

## End(Not run)
```

---

floodCharacteristics *Compute Flood Characteristics*

Description

This function computes flood characteristics from a given flood data dataframe.

Usage

```
floodCharacteristics(df, StnID, volume_threshold = 4355990)
```

Arguments

df	Dataframe with flood data.
StnID	ID of the Station
volume_threshold	The volume threshold to filter out floods (default is set to 4355990).

Value

A dataframe with flood characteristics.

Examples

```
## Not run:
# Assuming you have a dataframe 'df' with flood data:
result_df <- floodCharacteristics(df, '08041500')

## End(Not run)
```

flowDataDiagnostics *Analyze Flow Data for Trends, Change Points, and Stationarity*

Description

This function conducts a series of tests on flow data to detect trends, change points, and assess stationarity. The tests performed include the Mann-Kendall Test, Pettitt Test, Augmented Dickey-Fuller Test, KPSS Test, and the Phillips-Perron Unit Root Test.

Usage

```
flowDataDiagnostics(pf)
```

Arguments

pf	A numeric vector or time series representing flow data.

Value

A list containing the results of each of the tests performed.

References

McLeod, A. I. (2005). Kendall rank correlation and Mann-Kendall trend test. R Package Kendall, 602, 1-10.

Examples

```
## Not run:
  data <- rnorm(100)  # Example data
  results <- flowDataDiagnostics(data)
  print(results)

## End(Not run)
```

GenParPois *Generalized Pareto Distribution for Flood Metrics*

Description

This function fits a Generalized Pareto Distribution to flood flow data.

Usage

```
GenParPois(i, FldFl, Min_AP, thres)
```

Arguments

i	An identifier or index.
FldFl	A dataframe containing the flood flow data.
Min_AP	Minimum Annual Precipitation for Annual Maximum Series.
thres	The threshold for the Generalized Pareto Distribution.

Value

A dataframe with parameters and statistics related to the fitted Generalized Pareto Distribution.

Examples

```
## Not run:
# Assuming you have a dataframe 'FldFl' with flood flow data, an identifier 'i', Min_AP, and threshold 'thres':
result_df <- GenParPois(i, FldFl, Min_AP, thres)

## End(Not run)
```

GEVPars *Compute GEV Parameters*

Description

This function calculates the GEV parameters based on the provided dataframe.

Usage

```
GEVPars(a)
```

Arguments

a	A dataframe containing the columns 'GP_Location', 'GP_scale', 'GP_shape', and 'Pois_Lam'.

Value

A dataframe with added columns for 'GEV_loc', 'GEV_scale', and 'GEV_shp'.

Examples

```
## Not run:
# Assuming you have a dataframe 'FldFl' with flood flow data, an identifier 'i', Min_AP, and percentile:
result_df <- GEVPars(df)

## End(Not run)
```

gringorten2DBivariateEmpirical

Bivariate Empirical Distribution Using Gringorten's Formula

Description

This function computes the bivariate empirical distribution value for a given data point using Gringorten's plotting formula.

Usage

```
gringorten2DBivariateEmpirical(idx, X)
```

Arguments

idx	An integer index indicating which row of the data matrix X should be used as the reference point.
X	A numeric matrix or data frame where each row represents a bivariate data point.

Value

A numeric value representing the bivariate empirical distribution value for the specified data point.

Examples

```
## Not run:
data <- matrix(rnorm(100), ncol = 2)
gringorten2DBivariateEmpirical(10, data)

## End(Not run)
```

independenceCriteriaCheck

Independence Criteria Check on Flood

Description

This function applies independence criteria checks on flood data and filters out flows below a given threshold.

Usage

```
independenceCriteriaCheck(QT, Ar, daily_flow, flow_threshold = 10)
```

Arguments

QT	A dataframe with peak flows and corresponding time obtained after baseflow separation.
Ar	Area of the watershed in square kilometers.
daily_flow	Dataframe with daily flow data.
flow_threshold	The value to filter out flows (default is set to 10).

Value

A dataframe with peak flows and their corresponding dates after applying the independence criteria checks.

Examples

```
## Not run:
#Assuming you have dataframes 'QT' with peak flows, 'daily_flow' with daily flow data, and area 'Ar':
result_df <- independenceCriteriaCheck(QT, Ar, daily_flow)

## End(Not run)
```

---

plotAnnualPeakFlow *Time Series Plot of Annual Peak Flow*

Description

This function takes annual peak flow data and plots it as a time series for a specified range of years.

Usage

```
plotAnnualPeakFlow(df, st_yr, end_yr)
```

Arguments

df	A dataframe with annual peak flow in a column labeled peak_va.
st_yr	Start Year.
end_yr	End Year.

Value

Time Series Plot

Examples

```
## Not run:
plotAnnualPeakFlow(df,1980,2022)

## End(Not run)
```

plotAutocorrelations *Plot Autocorrelation and Partial Autocorrelation*

Description

This function generates the Autocorrelation Function (ACF) and 'Partial Autocorrelation Function (PACF) plots for the given data.

Usage

```
plotAutocorrelations(df)
```

Arguments

df	A numeric vector or time series.

Value

Autocorrelation Function (ACF) and Partial Autocorrelation Function (PACF) plots.

Examples

```
## Not run:
plotAutocorrelations(data)

## End(Not run)
```

RP_thres *Calculate Flood Metrics for Various Thresholds*

Description

This function calculates flood metrics for a sequence of thresholds.

Usage

```
RP_thres(i, FldFl, Min_AP)
```

Arguments

i	An identifier or index.
FldFl	A dataframe containing the flood flow data.
Min_AP	Minimum Annual Precipitation for Annual Maximum Series.

Value

A dataframe with flood metrics calculated for various thresholds.

Examples

```
## Not run:
# Assuming you have a dataframe 'FldFl' with flood flow data, an identifier 'i', and Min_AP:
result_df <- RP_thres(i, FldFl, Min_AP)

## End(Not run)
```

`separateBaseflowAndFloodflow` *Separate Streamflow into Baseflow and Floodflow*

Description

This function infills missing data using linear interpolation and separates baseflow from daily streamflow data.

Usage

```
separateBaseflowAndFloodflow(a, start, end)
```

Arguments

`a`	A dataframe containing the streamflow data.
`start`	The start date for the station in the format "mm/dd/yyyy".
`end`	The end date for the station in the format "mm/dd/yyyy".

Details

The BFI function is sourced from an external URL: https://raw.github.com/TonyLadson/BaseflowSeparation_LyneHollick/master/BFI.R

Value

A dataframe with infilled data and separated baseflow and floodflow.

Examples

```
## Not run:
# Assuming you have a dataframe 'a' with streamflow data and start and end dates:
result_df <- separateBaseflowAndFloodflow(a, start = "10/1/1939", end = "9/30/2022")

## End(Not run)
```

Index