Package 'eva'

October 18, 2018

Title Extreme Value Analysis with Goodness-of-Fit Testing

Date 2018-10-04

Version 0.2.5

Description Goodness-of-fit tests for selection of r in the r-largest order statistics (GEVr) model. Goodness-of-fit tests for threshold selection in the Generalized Pareto distribution (GPD). Random number generation and density functions for the GEVr distribution. Profile likelihood for return level estimation using the GEVr and Generalized Pareto distributions. P-value adjustments for sequential, multiple testing error control. Non-stationary fitting of GEVr and GPD.

Imports Matrix, parallel, stats, graphics, utils, EnvStats

Depends R(>= 2.10.0)

Suggests knitr, SpatialExtremes

License GPL (>= 2)

Repository CRAN

VignetteBuilder knitr

URL https://github.com/geekman1/eva_package

BugReports https://github.com/geekman1/eva_package/issues

LazyData true

RoxygenNote 5.0.1

NeedsCompilation no

Author Brian Bader [aut, cre],
Jun Yan [ctb]

Maintainer Brian Bader <bbader.stat@gmail.com>

Date/Publication 2018-10-18 11:40:03 UTC

R topics documented:

---

eva *eva: Extreme Value Analysis with Goodness-of-Fit Testing*

Description

The focus of this package is to provide much needed automated diagnostic tools (in the form of statistical hypothesis testing) to extreme value models. Other useful functionality is efficient and user-friendly non-stationary model fitting, profile likelihood confidence intervals, data generation in the r-largest order statistics model (GEVr), and ordered p-value multiplicity adjustments. Also, all routines are implemented to efficiently handle the near-zero shape parameter, which may cause numerical issues in other packages. Functions can be roughly assigned to the following topics:

Formal (Automated) Goodness-of-Fit Testing

`gevrSeqTests` is a wrapper function that performs sequential testing for r in the GEVr distribution, with adjusted p-values. It can implement three tests:

`gevrEd` An entropy difference test, which uses an asymptotic normal central limit theorem result.

`gevrPbScore` A score test, implemented using parametric bootstrap and can be run in parallel.

`gevrMultScore` An asymptotic approximation to the score test (computationally efficient).

`gpdSeqTests` is a wrapper function that performs sequential testing for thresholds in the Generalized Pareto distribution (GPD), with adjusted p-values. It can implement the following (six) tests:

`gpdAd` The Anderson-Darling test, with log-linear interpolated p-values. Can also be bootstrapped (with a parallel option).

`gpdCvm` The Cramer-Von Mises test, with log-linear interpolated p-values. Can also be bootstrapped (with a parallel option).

`gpdImAsym` An asymptotic information matrix test, with bootstrapped covariance estimates.

`gpdImPb` A full bootstrap version of information matrix test, with bootstrapped covariance estimates and critical values.

`gpdPbScore` A score test, implemented using parametric bootstrap and can be run in parallel.

`gpdMultScore` An asymptotic approximation to the score test (computationally effciient).

`pSeqStop` A simple function that reads in raw, ordered p-values and returns two sets that adjust for the familywise error rate and false discovery rate.

Data generation and model fitting

All the functions in this section (and package) efficiently handle a near-zero value of the shape parameter, which can cause numerical instability in similar functions from other packages. See the vignette for an example.

Data generation, density, quantile, and distribution functions can handle non-stationarity and vectorized inputs.

`gevr` Data generation and density function for the GEVr distribution, with distribution function and quantile functions available for GEV1 (block maxima).

`gpd` Data generation, distribution, quantile, and density functions for the GPD distribution.

`gevrFit` Non-stationary fitting of the GEVr distribution, with the option of maximum product spacings estimation when r=1. Uses formula statements for user friendliness and automatically centers/scales covariates when appropriate to speed up optimization.

`gpdFit` Non-stationary fitting of the GP distribution, with same options and implementation as 'gevrFit'. Allows non-stationary threshold to be used.

`gevrProfShape` Profile likelihood estimation for the shape parameter of the stationary GEVr distribution.

`gpdProfShape` Profile likelihood estimation for the shape parameter of the stationary GP distribution.

`gevrRl` Profile likelihood estimation for return levels of the stationary GEVr distribution.

`gpdRl` Profile likelihood estimation for return levels of the stationary GP distribution.

Visual Diagnostics

gevrDiag, gpdDiag Diagnostic plots for a fit to the GEVr (GP) distribution. For stationary models, return level, density, quantile, and probability plots are returned. For non-stationary models, residual quantile, residual probability, and residuals versus covariate plots are returned.

mrlPlot Plots the empirical mean residual life, with confidence intervals. Visual diagnostic tool to choose a threshold for exceedances.

Data

fortmax Top ten annual precipitation events (inches) for one rain gauge in Fort Collins, Colorado from 1900 through 1999.

lowestoft Top ten annual sea levels at the LoweStoft station tide gauge from 1964 - 2014.

---

fortmax *Top Ten Annual Precipitation: Fort Collins, Colorado*

Description

Top ten annual precipitation events (inches) for one rain gauge in Fort Collins, Colorado from 1900 through 1999. See Katz et al. (2002) Sec. 2.3.1 for more information and analyses.

Usage

```
data(fortmax)
```

Format

A data frame with 100 observations. Each year is considered an observation, with the top ten annual precipitation events.

Source

Colorado Climate Center, Colorado State University. This is the original data source containing the daily precipitation data.

References

Katz, R. W., Parlange, M. B. and Naveau, P. (2002) Statistics of extremes in hydrology. Advances in Water Resources, 25, 1287-1304.

Examples

```
data(fortmax)
y <- fortmax[, -1]
gevrSeqTests(y, method = "ed")
```

gevr *The GEVr Distribution*

Description

Random number generation (rgevr) and density (dgevr) functions for the GEVr distribution with parameters loc, scale, and shape. Also, quantile function (qgev) and cumulative distribution function (pgev) for the GEV1 distribution.

Usage

```
dgevr(x, loc = 0, scale = 1, shape = 0, log.d = FALSE)

rgevr(n, r, loc = 0, scale = 1, shape = 0)

qgev(p, loc = 0, scale = 1, shape = 0, lower.tail = TRUE,
  log.p = FALSE)

pgev(q, loc = 0, scale = 1, shape = 0, lower.tail = TRUE,
  log.p = FALSE)
```

Arguments

x	Vector or matrix of observations. If x is a matrix, each row is taken to be a new observation.
loc, scale, shape	Location, scale, and shape parameters. Can be vectors, but the lengths must be appropriate.
log.d	Logical: Whether or not to return the log density. (FALSE by default)
n	Number of observations
r	Number of order statistics for each observation.
p	Vector of probabilities.
lower.tail	Logical: If TRUE (default), probabilities are P[X <= x] otherwise, P[X > x].
log.p	Logical: If TRUE, probabilities p are given as log(p). (FALSE by default)
q	Vector of quantiles.

Details

GEVr data (in matrix x) should be of the form $x[i,1] > x[i,2] > \cdots > x[i,r]$ for each observation $i = 1, \ldots, n$. Note that currently the quantile and cdf functions are only for the GEV1 distribution. The GEVr distribution is also known as the r-largest order statistics model and is a generalization of the block maxima model (GEV1). The density function is given by

$$f_r(x_1, x_2, ..., x_r|\mu, \sigma, \xi) = \sigma^{-r} \exp\Big\{ -(1+\xi z_r)^{-\frac{1}{\xi}} - \left(\frac{1}{\xi}+1\right) \sum_{j=1}^{r} \log(1+\xi z_j) \Big\}$$

for some location parameter μ, scale parameter $\sigma > 0$, and shape parameter ξ, where $x_1 > \cdots > x_r$, $z_j = (x_j - \mu)/\sigma$, and $1 + \xi z_j > 0$ for $j = 1, \ldots, r$. When $r = 1$, this distribution is exactly the GEV distribution.

References

Coles, S. (2001). An introduction to statistical modeling of extreme values (Vol. 208). London: Springer.

Examples

```
## Plot the densities of the heavy and bounded upper tail forms of GEVr
set.seed(7)
dat1 <- rgevr(1000, 1, loc = 0, scale = 1, shape = 0.25)
dat2 <- rgevr(1000, 1, loc = 0, scale = 1, shape = -0.25)
hist(dat1, col = rgb(1, 0, 0, 0.5), xlim = c(-5, 10), ylim = c(0, 0.4),
     main = "Histogram of GEVr Densities", xlab = "Value", freq = FALSE)
hist(dat2, col = rgb(0, 0,1, 0.5), add = TRUE, freq = FALSE)
box()

## Generate sample with decreasing trend in location parameter
x <- rgevr(10, 2, loc = 10:1, scale = 1, shape = 0.1)
dgevr(x, loc = 10:1, scale = 10:1, shape = 0.1)

## Incorrect parameter specifications
# rgevr(10, 2, loc = 5:8, scale = 1, shape = 0.1)
# rgevr(1, 2, loc = 5:8, scale = 1:2, shape = 0.1)
```

---

gevrDiag *Diagnostic plots for a fit to the GEVr distribution.*

Description

Diagnostic plots for a fit to the GEVr distribution.

Usage

```
gevrDiag(z, conf = 0.95, method = c("delta", "profile"))
```

Arguments

z	A class object returned from 'gevrFit'.
conf	Confidence level used in the return level plot.
method	The method to compute the return level confidence interval - either delta method (default) or profile likelihood. Choosing profile likelihood may be quite slow.

Details

In certain cases the quantile plot may fail, because it requires solving a root equation. See the references for details.

Value

For stationary models, provides return level plot and density, probability, and quantile plots for each marginal order statistic. The overlaid density is the 'true' marginal density for the estimated parameters. For nonstationary models, provides residual probability and quantile plots. In addition, nonstationary models provide plots of the residuals vs. the parameter covariates.

References

Tawn, J. A. (1988). An extreme-value theory model for dependent observations. Journal of Hydrology, 101(1), 227-250.

Smith, R. L. (1986). Extreme value theory based on the r largest annual events. Journal of Hydrology, 86(1), 27-43.

Examples

```
## Not run
# x <- rgevr(500, 2, loc = 0.5, scale = 1, shape = 0.1)
# z <- gevrFit(x)
# plot(z)
```

---

gevrEd *GEVr Entropy Difference Test*

Description

Goodness-of-fit test for GEVr using the difference in likelihood between GEVr and GEV(r-1). This can be used sequentially to test for the choice of r.

Usage

```
gevrEd(data, theta = NULL)
```

Arguments

data	Data should be contain n rows, each a GEVr observation.
theta	Estimate for theta in the vector form (loc, scale, shape). If NULL, uses the MLE from the full data.

Details

GEVr data (in matrix x) should be of the form $x[i,1] > x[i,2] > \cdots > x[i,r]$ for each observation $i = 1, \ldots, n$. The test uses an asymptotic normality result based on the expected entropy between the GEVr and GEV(r-1) likelihoods. See reference for detailed information. This test can be used to sequentially test for the choice of r, implemented in the function 'gevrSeqTests'.

Value

statistic	Test statistic.
p.value	P-value for the test.
theta	Estimate of theta using the top r order statistics.

References

Bader B., Yan J., & Zhang X. (2015). Automated Selection of r for the r Largest Order Statistics Approach with Adjustment for Sequential Testing. Department of Statistics, University of Connecticut.

Examples

```
## This will test if the GEV2 distribution fits the data.
x <- rgevr(100, 2, loc = 0.5, scale = 1, shape = 0.5)
result <- gevrEd(x)
```

---

gevrFit *Parameter estimation for the GEVr distribution model*

Description

This function provides maximum likelihood estimation for the GEVr model, with the option of probability weighted moment and maximum product spacing estimation for block maxima (GEV1) data. It also allows generalized linear modeling of the parameters.

Usage

```
gevrFit(data, method = c("mle", "mps", "pwm"), information = c("expected",
  "observed"), locvars = NULL, scalevars = NULL, shapevars = NULL,
  locform = ~1, scaleform = ~1, shapeform = ~1, loclink = identity,
  scalelink = identity, shapelink = identity, gumbel = FALSE,
  start = NULL, opt = "Nelder-Mead", maxit = 10000, ...)
```

Arguments

data	Data should be a matrix from the GEVr distribution.
method	Method of estimation - maximum likelihood (mle), maximum product spacings (mps), and probability weighted moments (pwm). Uses mle by default. For $r > 1$, only mle can be used.
information	Whether standard errors should be calculated via observed or expected (default) information. For probability weighted moments, only expected information will be used if possible. In the case with covariates, only observed information is available.

`locvars, scalevars, shapevars`
: A dataframe of covariates to use for modeling of the each parameter. Parameter intercepts are automatically handled by the function. Defaults to NULL for the stationary model.

`locform, scaleform, shapeform`
: An object of class 'formula' (or one that can be coerced into that class), specifying the model of each parameter. By default, assumes stationary (intercept only) model. See details.

`loclink, scalelink, shapelink`
: A link function specifying the relationship between the covariates and each parameter. Defaults to the identity function. For the stationary model, only the identity link should be used.

`gumbel`
: Whether to fit the Gumbel (type I) extreme value distribution (i.e. shape parameter equals zero). Defaults to FALSE.

`start`
: Option to provide a set of starting parameters to optim; a vector of location, scale, and shape, in that order. Otherwise, the routine attempts to find good starting parameters. See details.

`opt`
: Optimization method to use with optim.

`maxit`
: Number of iterations to use in optimization, passed to optim. Defaults to 10,000.

`...`
: Additional arguments to pass to optim.

Details

In the stationary case (no covariates), starting parameters for mle and mps estimation are the probability weighted moment estimates. In the case where covariates are used, the starting intercept parameters are the probability weighted moment estimates from the stationary case and the parameters based on covariates are initially set to zero. For non-stationary parameters, the first reported estimate refers to the intercept term. Covariates are centered and scaled automatically to speed up optimization, and then transformed back to original scale.

Formulas for generalized linear modeling of the parameters should be given in the form '~ var1 + var2 + $\cdots$'. Essentially, specification here is the same as would be if using function 'lm' for only the right hand side of the equation. Interactions, polynomials, etc. can be handled as in the 'formula' class.

Intercept terms are automatically handled by the function. By default, the link functions are the identity function and the covariate dependent scale parameter estimates are forced to be positive. For some link function $f(\cdot)$ and for example, scale parameter σ, the link is written as $\sigma = f(\sigma_1 x_1 + \sigma_2 x_2 + \cdots + \sigma_k x_k)$.

Maximum likelihood estimation can be used in all cases. Probability weighted moment estimation can only be used if $r = 1$ and data is assumed to be stationary. Maximum product spacings estimation can be used in the non-stationary case, but only if $r = 1$.

Value

A list describing the fit, including parameter estimates and standard errors for the mle and mps methods. Returns as a class object 'gevrFit' to be used with diagnostic plots.

Examples

```
set.seed(7)
x1 <- rgevr(500, 1, loc = 0.5, scale = 1, shape = 0.3)
result1 <- gevrFit(x1, method = "mps")

## A linear trend in the location and scale parameter
n <- 100
r <- 10
x2 <- rgevr(n, r, loc = 100 + 1:n / 50,  scale = 1 + 1:n / 300, shape = 0)

covs <- as.data.frame(seq(1, n, 1))
names(covs) <- c("Trend1")
## Create some unrelated covariates
covs$Trend2 <- rnorm(n)
covs$Trend3 <- 30 * runif(n)
result2 <- gevrFit(data = x2, method = "mle", locvars = covs, locform = ~ Trend1 + Trend2*Trend3,
scalevars = covs, scaleform = ~ Trend1)

## Show summary of estimates
result2
```

---

gevrMultScore *GEVr Multiplier Score Test*

Description

Fast weighted bootstrap alternative to the parametric bootstrap procedure for the GEVr score test.

Usage

```
gevrMultScore(data, bootnum, information = c("expected", "observed"))
```

Arguments

`data`	Data should be contain n rows, each a GEVr observation.
`bootnum`	Number of bootstrap replicates.
`information`	To use expected (default) or observed information in the test.

Details

GEVr data (in matrix x) should be of the form $x[i,1] > x[i,2] > \cdots > x[i,r]$ for each observation $i = 1, \ldots, n$.

Value

`statistic`	Test statistic.
`p.value`	P-value for the test.
`theta`	Value of theta used in the test.

References

Bader B., Yan J., & Zhang X. (2015). Automated Selection of r for the r Largest Order Statistics Approach with Adjustment for Sequential Testing. Department of Statistics, University of Connecticut.

Examples

```
x <- rgevr(500, 5, loc = 0.5, scale = 1, shape = 0.3)
result <- gevrMultScore(x, bootnum = 1000)
```

`gevrPbScore` *GEVr Parametric Bootstrap Score Test*

Description

Parametric bootstrap score test procedure to assess goodness-of-fit to the GEVr distribution.

Usage

```
gevrPbScore(data, bootnum, information = c("expected", "observed"),
  allowParallel = FALSE, numCores = 1)
```

Arguments

`data`	Data should be contain n rows, each a GEVr observation.
`bootnum`	Number of bootstrap replicates.
`information`	To use expected (default) or observed information in the test.
`allowParallel`	Should the bootstrap procedure be run in parallel or not. Defaults to false.
`numCores`	If allowParallel is true, specify the number of cores to use.

Details

GEVr data (in matrix x) should be of the form $x[i,1] > x[i,2] > \cdots > x[i,r]$ for each observation $i = 1, \ldots, n$.

Value

`statistic`	Test statistic.
`p.value`	P-value for the test.
`theta`	Initial value of theta used in the test.
`effective_bootnum`	Effective number of bootstrap replicates (only those that converged are used).

References

Bader B., Yan J., & Zhang X. (2015). Automated Selection of r for the r Largest Order Statistics Approach with Adjustment for Sequential Testing. Department of Statistics, University of Connecticut.

Examples

```
## Not run
## Generate some data from GEVr
# x <- rgevr(200, 5, loc = 0.5, scale = 1, shape = 0.25)
# gevrPbScore(x, bootnum = 99)
```

`gevrProfShape` *GEVr Shape Parameter Profile Likelihood Estimation for Stationary Models*

Description

Computes the profile likelihood based confidence interval for the shape parameter of the stationary GEVr model.

Usage

```
gevrProfShape(z, conf = 0.95, plot = TRUE, opt = c("Nelder-Mead"))
```

Arguments

`z`	A class object returned from gevrFit.
`conf`	Confidence level to use. Defaults to 95 percent.
`plot`	Plot the profile likelihood and estimate (vertical line)?
`opt`	Optimization method to maximize the profile likelihood, passed to optim. The default method is Nelder-Mead.

Value

`Estimate`	Estimated shape parameter.
`CI`	Profile likelihood based confidence interval for the shape parameter.
`ConfLevel`	The confidence level used.

Examples

```
## Compare the length of the shape confidence intervals using GEV1 vs. GEV10
set.seed(7)
x <- rgevr(200, 10, loc = 0.5, scale = 1, shape = -0.3)
z1 <- gevrFit(x[, 1])
z2 <- gevrFit(x)
gevrProfShape(z1)
gevrProfShape(z2)
```

gevrRl *GEVr Return Level Estimate and Confidence Interval for Stationary Models*

Description

Computes stationary m-period return level estimate and interval, using either the delta method or profile likelihood.

Usage

```
gevrRl(z, period, conf = 0.95, method = c("delta", "profile"),
  plot = TRUE, opt = c("Nelder-Mead"))
```

Arguments

z	A class object returned from gevrFit. Must be a stationary fit.
period	The number of periods to use for the return level.
conf	Confidence level. Defaults to 95 percent.
method	The method to compute the confidence interval - either delta method (default) or profile likelihood.
plot	Plot the profile likelihood and estimate (vertical line)?
opt	Optimization method to maximize the profile likelihood if that is selected. The default method is Nelder-Mead.

Details

It is generally accepted that profile likelihood confidence intervals provide greater accuracy than the delta method, in particular for large return level periods. Also, by their nature, delta method confidence intervals must be symmetric which may be undesirable for return level estimation. If the original fit was Gumbel, then return levels will be for the Gumbel distribution.

Caution: The profile likelihood optimization may be slow (on the order of minutes).

Value

Estimate	Estimated m-period return level.
CI	Confidence interval for the m-period return level.
Period	The period length used.
ConfLevel	The confidence level used.

References

http://www.mas.ncl.ac.uk/~nlf8/teaching/mas8391/background/chapter2.pdf

Coles, S. (2001). An introduction to statistical modeling of extreme values (Vol. 208). London: Springer.

Examples

```
x <- rgevr(100, 2, loc = 0.5, scale = 1, shape = -0.3)
z <- gevrFit(x)
## Compute 250-period return level.
gevrRl(z, 250, method = "delta")
```

---

gevrSeqTests *Sequential Tests for the GEVr Model*

Description

Sequentially performs the entropy difference (ED) test or the multiplier or parametric bootstrap score tests for the GEVr model.

Usage

```
gevrSeqTests(data, bootnum = NULL, method = c("ed", "pbscore", "multscore"),
  information = c("expected", "observed"), allowParallel = FALSE,
  numCores = 1)
```

Arguments

data	Data should be contain n rows, each a GEVr observation.
bootnum	If method equals 'pbscore' or 'multscore', the number of bootstrap simulations to use.
method	Which test to run: ED test (ed), multiplier (multscore) or parametric bootstrap (pbscore) score test.
information	To use expected (default) or observed information in the score tests.
allowParallel	If method equals 'pbscore', should the parametric boostrap procedure be run in parallel or not. Defaults to false.
numCores	If allowParallel is true, specify the number of cores to use.

Details

GEVr data (in matrix x) should be of the form $x[i,1] > x[i,2] > \cdots > x[i,r]$ for each observation $i = 1, \ldots, n$. See function 'pSeqStop' for details on transformed p-values.

Value

Function returns a dataframe containing the test statistics, estimates, and p-value results of the sequential tests.

r	Value of r to be tested.
p.values	Raw p-values from the individual tests at each value of r.
ForwardStop	Transformed p-values according to the ForwardStop stopping rule.

StrongStop	Transformed p-values according to the StrongStop stopping rule.
statistic	Returned test statistics of each individual test.
est.loc	Estimated location parameter for the given r.
est.scale	Estimated scale parameter for the given r.
est.shape	Estimated shape parameter for the given r.

Examples

```
x <- rgevr(200, 5, loc = 0.5, scale = 1, shape = 0.25)
gevrSeqTests(x, method = "ed")
```

---

gpd *The Generalized Pareto Distribution (GPD)*

Description

Density, distribution function, quantile function and random number generation for the Generalized Pareto distribution with location, scale, and shape parameters.

Usage

```
dgpd(x, loc = 0, scale = 1, shape = 0, log.d = FALSE)

rgpd(n, loc = 0, scale = 1, shape = 0)

qgpd(p, loc = 0, scale = 1, shape = 0, lower.tail = TRUE,
  log.p = FALSE)

pgpd(q, loc = 0, scale = 1, shape = 0, lower.tail = TRUE,
  log.p = FALSE)
```

Arguments

x	Vector of observations.
loc, scale, shape	Location, scale, and shape parameters. Can be vectors, but the lengths must be appropriate.
log.d	Logical; if TRUE, the log density is returned.
n	Number of observations.
p	Vector of probabilities.
lower.tail	Logical; if TRUE (default), probabilities are P[X <= x], otherwise, P[X > x].
log.p	Logical; if TRUE, probabilities p are given as log(p).
q	Vector of quantiles.

Details

The Generalized Pareto distribution function is given (Pickands, 1975) by

$$H(y) = 1 - \left[1 + \frac{\xi(y-\mu)}{\sigma}\right]^{-1/\xi}$$

defined on $\{y : y > 0, (1 + \xi(y - \mu)/\sigma) > 0\}$, with location μ, scale $\sigma > 0$, and shape parameter ξ.

References

Pickands III, J. (1975). Statistical inference using extreme order statistics. Annals of Statistics, 119-131.

Examples

```
dgpd(2:4, 1, 0.5, 0.01)
dgpd(2, -2:1, 0.5, 0.01)
pgpd(2:4, 1, 0.5, 0.01)
qgpd(seq(0.9, 0.6, -0.1), 2, 0.5, 0.01)
rgpd(6, 1, 0.5, 0.01)

## Generate sample with linear trend in location parameter
rgpd(6, 1:6, 0.5, 0.01)

## Generate sample with linear trend in location and scale parameter
rgpd(6, 1:6, seq(0.5, 3, 0.5), 0.01)

p <- (1:9)/10
pgpd(qgpd(p, 1, 2, 0.8), 1, 2, 0.8)
## [1] 0.1 0.2 0.3 0.4 0.5 0.6 0.7 0.8 0.9

## Incorrect syntax (parameter vectors are of different lengths other than 1)
# rgpd(1, 1:8, 1:5, 0)

## Also incorrect syntax
# rgpd(10, 1:8, 1, 0.01)
```

---

gpdAd *Generalized Pareto Distribution Anderson-Darling Test*

Description

Anderson-Darling goodness-of-fit test for the Generalized Pareto (GPD) distribution.

Usage

```
gpdAd(data, bootstrap = FALSE, bootnum = NULL, allowParallel = FALSE,
  numCores = 1)
```

Arguments

data	Data should be in vector form, assumed to be from the GPD.
bootstrap	Should bootstrap be used to obtain p-values for the test? By default, a table of critical values is used via interpolation. See details.
bootnum	Number of replicates if bootstrap is used.
allowParallel	Should the bootstrap procedure be run in parallel or not. Defaults to false.
numCores	If allowParallel is true, specify the number of cores to use.

Details

A table of critical values were generated via Monte Carlo simulation for shape parameters -0.5 to 1.0 by 0.1, which provides p-values via log-linear interpolation from .001 to .999. For p-values below .001, a linear equation exists by regressing -log(p-value) on the critical values for the tail of the distribution (.950 to .999 upper percentiles). This regression provides a method to extrapolate to arbitrarily small p-values.

Value

statistic	Test statistic.
p.value	P-value for the test.
theta	Estimated value of theta for the initial data.
effective_bootnum	Effective number of bootstrap replicates if bootstrap based p-value is used (only those that converged are used).

References

Choulakian, V., & Stephens, M. A. (2001). Goodness-of-fit tests for the Generalized Pareto distribution. Technometrics, 43(4), 478-484.

Examples

```
## Generate some data from GPD
x <- rgpd(200, loc = 0, scale = 1, shape = 0.2)
gpdAd(x)
```

---

gpdCvm *Generalized Pareto Distribution Cramer-von Mises Test*

Description

Cramer-von Mises goodness-of-fit test for the Generalized Pareto (GPD) distribution.

Usage

```
gpdCvm(data, bootstrap = FALSE, bootnum = NULL, allowParallel = FALSE,
  numCores = 1)
```

Arguments

data	Data should be in vector form, assumed to be from the GPD.
bootstrap	Should bootstrap be used to obtain p-values for the test? By default, a table of critical values is used via interpolation. See details.
bootnum	Number of bootstrap replicates.
allowParallel	Should the bootstrap procedure be run in parallel or not. Defaults to false.
numCores	If allowParallel is true, specify the number of cores to use.

Details

A table of critical values were generated via Monte Carlo simulation for shape parameters -0.5 to 1.0 by 0.1, which provides p-values via log-linear interpolation from .001 to .999. For p-values below .001, a linear equation exists by regressing -log(p-value) on the critical values for the tail of the distribution (.950 to .999 upper percentiles). This regression provides a method to extrapolate to arbitrarily small p-values.

Value

statistic	Test statistic.
p.value	P-value for the test.
theta	Estimated value of theta for the initial data.
effective_bootnum	Effective number of bootstrap replicates if bootstrap based p-value is used (only those that converged are used).

References

Choulakian, V., & Stephens, M. A. (2001). Goodness-of-fit tests for the Generalized Pareto distribution. Technometrics, 43(4), 478-484.

Examples

```
## Generate some data from GPD
x <- rgpd(200, loc = 0, scale = 1, shape = 0.2)
gpdCvm(x)
```

---

gpdDiag *Diagnostic plots for a fit to the Generalized Pareto distribution*

Description

Diagnostic plots for a fit to the Generalized Pareto distribution

Usage

```
gpdDiag(z, conf = 0.95, method = c("delta", "profile"))
```

Arguments

z	A class object returned from 'gpdFit'.
conf	Confidence level used in the return level plot.
method	The method to compute the return level confidence interval - either delta method (default) or profile likelihood. Choosing profile likelihood may be quite slow.

Details

See the reference for details on how return levels are calculated.

Value

For stationary models, provides return level, density, probability, and quantile plots for the GPD exceedances. The overlaid density is the 'true' density for the estimated parameters. For nonstationary models, provides residual probability and quantile plots. In addition, nonstationary models provide plots of the residuals vs. the parameter covariates.

References

Coles, S. (2001). An introduction to statistical modeling of extreme values (Vol. 208). London: Springer.

Examples

```
## Not run
# x <- rgpd(10000, loc = 0.5, scale = 1, shape = 0.1)
# z <- gpdFit(x, nextremes = 500)
# plot(z)
```

---

gpdFit *Parameter estimation for the Generalized Pareto Distribution (GPD)*

Description

Fits exceedances above a chosen threshold to the Generalized Pareto model. Various estimation procedures can be used, including maximum likelihood, probability weighted moments, and maximum product spacing. It also allows generalized linear modeling of the parameters.

Usage

```
gpdFit(data, threshold = NA, nextremes = NA, npp = 365,
  method = c("mle", "mps", "pwm"), information = c("expected", "observed"),
  scalevars = NULL, shapevars = NULL, scaleform = ~1, shapeform = ~1,
  scalelink = identity, shapelink = identity, start = NULL,
  opt = "Nelder-Mead", maxit = 10000, ...)
```

Arguments

`data`	Data should be a numeric vector from the GPD.
`threshold`	A threshold value or vector of the same length as the data.
`nextremes`	Number of upper extremes to be used (either this or the threshold must be given, but not both).
`npp`	Length of each period (typically year). Is used in return level estimation. Defaults to 365.
`method`	Method of estimation - maximum likelihood (mle), maximum product spacing (mps), and probability weighted moments (pwm). Uses mle by default. For pwm, only the stationary model can be fit.
`information`	Whether standard errors should be calculated via observed or expected (default) information. For probability weighted moments, only expected information will be used if possible. For non-stationary models, only observed information is used.
`scalevars, shapevars`	A dataframe of covariates to use for modeling of the each parameter. Parameter intercepts are automatically handled by the function. Defaults to NULL for the stationary model.
`scaleform, shapeform`	An object of class 'formula' (or one that can be coerced into that class), specifying the model of each parameter. By default, assumes stationary (intercept only) model. See details.
`scalelink, shapelink`	A link function specifying the relationship between the covariates and each parameter. Defaults to the identity function. For the stationary model, only the identity link should be used.
`start`	Option to provide a set of starting parameters to optim; a vector of scale and shape, in that order. Otherwise, the routine attempts to find good starting parameters. See details.
`opt`	Optimization method to use with optim.
`maxit`	Number of iterations to use in optimization, passed to optim. Defaults to 10,000.
`...`	Additional arguments to pass to optim.

Details

The base code for finding probability weighted moments is taken from the R package evir. See citation. In the stationary case (no covariates), starting parameters for mle and mps estimation are the probability weighted moment estimates. In the case where covariates are used, the starting intercept parameters are the probability weighted moment estimates from the stationary case and the parameters based on covariates are initially set to zero. For non-stationary parameters, the first reported estimate refers to the intercept term. Covariates are centered and scaled automatically to speed up optimization, and then transformed back to original scale.

Formulas for generalized linear modeling of the parameters should be given in the form '~ var1 + var2 + $\cdots$'. Essentially, specification here is the same as would be if using function 'lm' for only the right hand side of the equation. Interactions, polynomials, etc. can be handled as in the 'formula'

class.
Intercept terms are automatically handled by the function. By default, the link functions are the identity function and the covariate dependent scale parameter estimates are forced to be positive. For some link function $f(\cdot)$ and for example, scale parameter σ, the link is written as $\sigma = f(\sigma_1 x_1 + \sigma_2 x_2 + \cdots + \sigma_k x_k)$.
Maximum likelihood estimation and maximum product spacing estimation can be used in all cases. Probability weighted moments can only be used for stationary models.

Value

A class object 'gpdFit' describing the fit, including parameter estimates and standard errors.

References

Pfaff, Bernhard, Alexander McNeil, and A. Stephenson. "evir: Extreme Values in R." R package version (2012): 1-7.

Examples

```
## Fit data using the three different estimation procedures
set.seed(7)
x <- rgpd(2000, loc = 0, scale = 2, shape = 0.2)
## Set threshold at 4
mle_fit <- gpdFit(x, threshold = 4, method = "mle")
pwm_fit <- gpdFit(x, threshold = 4, method = "pwm")
mps_fit <- gpdFit(x, threshold = 4, method = "mps")
## Look at the difference in parameter estimates and errors
mle_fit$par.ests
pwm_fit$par.ests
mps_fit$par.ests

mle_fit$par.ses
pwm_fit$par.ses
mps_fit$par.ses

## A linear trend in the scale parameter
set.seed(7)
n <- 300
x2 <- rgpd(n, loc = 0, scale = 1 + 1:n / 200, shape = 0)

covs <- as.data.frame(seq(1, n, 1))
names(covs) <- c("Trend1")

result1 <- gpdFit(x2, threshold = 0, scalevars = covs, scaleform = ~ Trend1)

## Show summary of estimates
result1
```

gpdImAsym *GPD Asymptotic Adjusted Information Matrix (IM) Test*

Description

Runs the IM Test using bootstrap estimated covariance matrix. Asymptotically (in sample size) follows the F(3, bootnum - 3) distribution (see reference for details).

Usage

```
gpdImAsym(data, bootnum, theta = NULL)
```

Arguments

data	Data should be in vector form.
bootnum	Number of bootstrap replicates for the covariance estimate.
theta	Estimate for theta in the vector form (scale, shape). If NULL, uses the MLE.

Value

statistic	Test statistic.
p.value	P-value for the test.
theta	Value of theta used in the test.
effective_bootnum	Effective number of bootstrap replicates used for the covariance estimate. If a replicate fails to converge, it will not be used in the estimation.

References

Dhaene, G., & Hoorelbeke, D. (2004). The information matrix test with bootstrap-based covariance matrix estimation. Economics Letters, 82(3), 341-347.

Examples

```
## Generate some data from GPD
x <- rgpd(200, loc = 0, scale = 1, shape = 0.2)
gpdImAsym(x, bootnum = 50)
```

gpdImPb *GPD Bootstrapped Information Matrix (IM) Test*

Description

Runs the IM Test using a two-step iterative procedure, to boostrap the covariance estimate and critical values. See reference for details.

Usage

```
gpdImPb(data, inner, outer, allowParallel = FALSE, numCores = 1)
```

Arguments

data	Data should be in vector form.
inner	Number of bootstrap replicates for the covariance estimate.
outer	Number of bootstrap replicates for critical values.
allowParallel	Should the outer bootstrap procedure be run in parallel or not. Defaults to false.
numCores	If allowParallel is true, specify the number of cores to use.

Details

Warning: This test can be very slow, since the covariance estimation is nested within the outer replicates. It would be recommended to use a small number of replicates for the covariance estimate (at most 50).

Value

statistic	Test statistic.
p.value	P-value for the test.
theta	Estimate of theta for the initial dataset.
effective_bootnum	Effective number of outer bootstrap replicates used (only those that converged are used).

References

Dhaene, G., & Hoorelbeke, D. (2004). The information matrix test with bootstrap-based covariance matrix estimation. Economics Letters, 82(3), 341-347.

Examples

```
## Not run
# x <- rgpd(200, loc = 0, scale = 1, shape = 0.2)
# gpdImPb(x, inner = 20, outer = 99)
```

gpdMultScore *GPD Multiplier Score Test*

Description

Fast weighted bootstrap alternative to the parametric bootstrap procedure for the Generalized Pareto score test.

Usage

```
gpdMultScore(data, bootnum, information = c("expected", "observed"))
```

Arguments

data	Data should be in vector form.
bootnum	Number of bootstrap replicates.
information	To use expected (default) or observed information in the test.

Value

statistic	Test statistic.
p.value	P-value for the test.
theta	Value of theta used in the test.

Examples

```
x <- rgpd(100, loc = 0, scale = 1, shape = 0.25)
gpdMultScore(x, bootnum = 1000)
```

gpdPbScore *GPD Parametric Bootstrap Score Test*

Description

Parametric bootstrap score test procedure to assess goodness-of-fit to the Generalized Pareto distribution.

Usage

```
gpdPbScore(data, bootnum, information = c("expected", "observed"),
  allowParallel = FALSE, numCores = 1)
```

Arguments

data	Data should be in vector form.
bootnum	Number of bootstrap replicates.
information	To use expected (default) or observed information in the test.
allowParallel	Should the bootstrap procedure be run in parallel or not. Defaults to false.
numCores	If allowParallel is true, specify the number of cores to use.

Value

statistic	Test statistic.
p.value	P-value for the test.
theta	Estimated value of theta for the initial data.
effective_bootnum	Effective number of bootstrap replicates (only those that converged are used).

Examples

```
## Generate some data from GPD
x <- rgpd(200, loc = 0, scale = 1, shape = 0.2)
gpdPbScore(x, bootnum = 100)
```

---

gpdProfShape — *GPD Shape Parameter Profile Likelihood Estimation for Stationary Models*

Description

Computes the profile likelihood based confidence interval for the shape parameter of the stationary Generalized Pareto model.

Usage

```
gpdProfShape(z, conf = 0.95, plot = TRUE)
```

Arguments

z	A class object returned from gpdFit.
conf	Confidence level to use. Defaults to 95 percent.
plot	Plot the profile likelihood and estimate (vertical line)?

Value

Estimate	Estimated shape parameter.
CI	Profile likelihood based confidence interval for the shape parameter.
ConfLevel	The confidence level used.

Examples

```
x <- rgpd(200, loc = 0, scale = 1, shape = 0.25)
z <- gpdFit(x, threshold = 0)
gpdProfShape(z)
```

---

gpdRl *GPD Return Level Estimate and Confidence Interval for Stationary Models*

Description

Computes stationary m-period return level estimate and interval for the Generalized Pareto distribution, using either the delta method or profile likelihood.

Usage

```
gpdRl(z, period, conf = 0.95, method = c("delta", "profile"), plot = TRUE,
  opt = c("Nelder-Mead"))
```

Arguments

z	An object of class 'gpdFit'.
period	The number of periods to use for the return level.
conf	Confidence level. Defaults to 95 percent.
method	The method to compute the confidence interval - either delta method (default) or profile likelihood.
plot	Plot the profile likelihood and estimate (vertical line)?
opt	Optimization method to maximize the profile likelihood if that is selected. Argument passed to optim. The default method is Nelder-Mead.

Details

Caution: The profile likelihood optimization may be slow for large datasets.

Value

Estimate	Estimated m-period return level.
CI	Confidence interval for the m-period return level.
Period	The period length used.
ConfLevel	The confidence level used.

References

Coles, S. (2001). An introduction to statistical modeling of extreme values (Vol. 208). London: Springer.

Examples

```
x <- rgpd(5000, loc = 0, scale = 1, shape = -0.1)
## Compute 50-period return level.
z <- gpdFit(x, nextremes = 200)
gpdRl(z, period = 50, method = "delta")
gpdRl(z, period = 50, method = "profile")
```

---

gpdSeqTests *GPD Multiple Threshold Goodness-of-Fit Testing*

Description

Wrapper function to test multiple thresholds for goodness-of-fit to the Generalized Pareto model. Can choose which test to run from the available tests in this package.

Usage

```
gpdSeqTests(data, thresholds = NA, nextremes = NA, method = c("ad", "cvm",
  "pbscore", "multscore", "imasym", "impb"), nsim = NULL, inner = NULL,
  outer = NULL, information = c("expected", "observed"),
  allowParallel = FALSE, numCores = 1)
```

Arguments

data	Original, full dataset in vector form.
thresholds	A set of threshold values (either this or a set of the number of extremes must be given, but not both). Must be provided as a vector.
nextremes	A set of the number of upper extremes to be used, provided as a vector.
method	Which test to run to sequentially test the thresholds. Must be one of 'ad', 'cvm', 'pbscore', 'multscore', 'imasym', or 'impb'.
nsim	Number of boostrap replicates for the 'ad', 'cvm', 'pbscore', 'multscore', and 'imasym' tests.
inner	Number of inner boostrap replicates if 'impb' test is chosen.
outer	Number of outer boostrap replicates if 'impb' test is chosen.
information	To use observed or expected (default) information for the 'pbscore' and 'multscore' tests.
allowParallel	If selected, should the 'cvm', 'ad', 'pbscore', or 'impb' procedure be run in parallel or not. Defaults to false.
numCores	If allowParallel is true, specify the number of cores to use.

Details

Function returns a matrix containing the thresholds used, the number of observations above each threshold, the corresponding test statistics, p-values (raw and transformed), and parameter estimates at each threshold. The user must provide the data, a vector of thresholds or number of upper extremes to be used, and select the test.

Value

threshold	The threshold used for the test.
num.above	The number of observations above the given threshold.
p.values	Raw p-values for the thresholds tested.
ForwardStop	Transformed p-values according to the ForwardStop stopping rule.
StrongStop	Transformed p-values according to the StrongStop stopping rule.
statistic	Returned test statistics of each individual test.
est.scale	Estimated scale parameter for the given threshold.
est.shape	Estimated shape parameter for the given threshold.

Examples

```
set.seed(7)
x <- rgpd(10000, loc = 0, scale = 5, shape = 0.2)
## A vector of thresholds to test
threshes <- c(1.5, 2.5, 3.5, 4.5, 5.5)
gpdSeqTests(x, thresholds = threshes, method = "ad")
```

---

lowestoft *Top Ten Annual Sea Levels: Lowestoft, UK (1964 - 2014)*

Description

Top ten annual sea levels at the LoweStoft Station tide gauge from 1964 - 2014. From 1964 - 1992, raw data is collected in hour intervals; from 1993 - present, raw data is collected in fifteen minute intervals. Data is pre-processed here to account for storm length - see reference for details.

Usage

```
data(lowestoft)
```

Format

A data matrix with 51 observations. Each year is considered an observation, with the top ten annual sea level events.

Source

UK Tide Gauge Network (Lowestoft Station): https://www.bodc.ac.uk/data/online_delivery/ntslf/processed/

References

Bader B., Yan J., & Zhang X. (2015). Automated Selection of r for the r Largest Order Statistics Approach with Adjustment for Sequential Testing. Department of Statistics, University of Connecticut.

Examples

```
data(lowestoft)
gevrSeqTests(lowestoft, method = "ed")
## Not run
## Look at the difference in confidence intervals between r = 1 and r = 10
# z1 <- gevrFit(lowestoft[, 1])
# z2 <- gevrFit(lowestoft)
# gevrRl(z1, 50, method = "profile")
# gevrRl(z2, 50, method = "profile")
```

---

mrlPlot *Mean Residual Life Plot for the Generalized Pareto Distribution*

Description

Plots the empirical mean residual life, with confidence intervals. The mean residual life plot provides a visual diagnostic tool to choose a threshold for exceedances.

Usage

```
mrlPlot(data, thresholds = NULL, conf = 0.95, npoints = 100)
```

Arguments

data	Vector of data.
thresholds	A numeric vector of threshold(s) to plot vertically. Defaults to NULL.
conf	The level of the confidence bounds to use. Defaults to 0.95.
npoints	The number of points to interpolate with. Defaults to 100.

Examples

```
## Not run
## x <- rgpd(500, loc = 0, scale = 1, shape = 0.1)
## mrlPlot(x, thresholds = c(2))
```

---

pSeqStop *P-Value Sequential Adjustment*

Description

Given a set of (ordered) p-values, returns p-values adjusted according to the ForwardStop and StrongStop stopping rules.

Usage

```
pSeqStop(p)
```

Arguments

p	Vector of ordered p-values.

Details

Roughly speaking, under the assumption of independent but ordered p-values, the StrongStop adjusted p-values control for the familywise error rate, while ForwardStop provides control for the false discovery rate.

Value

StrongStop	Vector of ordered p-values adjusted for the familywise error rate.
ForwardStop	Vector of ordered p-values adjusted for the false discovery rate.
UnAdjusted	Vector of non-transformed p-values.

References

G'Sell, M. G., Wager, S., Chouldechova, A., & Tibshirani, R. (2013). Sequential Selection Procedures and False Discovery Rate Control. arXiv preprint arXiv:1309.5352.

Benjamini, Y., & Hochberg, Y. (1995). Controlling the false discovery rate: a practical and powerful approach to multiple testing. Journal of the Royal Statistical Society. Series B (Methodological), 289-300.

Bader B., Yan J., & Zhang X. (2015). Automated Selection of r for the r Largest Order Statistics Approach with Adjustment for Sequential Testing. Department of Statistics, University of Connecticut.

Examples

```
x <- rgevr(500, 10, loc = 0.5, scale = 1, shape = 0.5)
y <- gevrSeqTests(x, method = "ed")
pSeqStop(rev(y$p.values))
```

Index