Package 'evir'

July 26, 2012

Version 1.7-3

Date 2011-07-22

Title Extreme Values in R

Depends stats

Description Functions for extreme value theory, which may be divided
into the following groups; exploratory data analysis, block
maxima, peaks over thresholds (univariate and bivariate), point processes, gev/gpd distributions.

License GPL (>= 2)

Author Bernhard Pfaff [aut, cre], Alexander McNeil [aut] (S original
(EVIS)), Alec Stephenson [trl] (R port of EVIS)

Maintainer Bernhard Pfaff <bernhard@pfaffikus.de>

Repository CRAN

Repository/R-Forge/Project evir

Repository/R-Forge/Revision 9

Date/Publication 2012-07-26 21:32:20

R topics documented:

---

bmw *Daily Log Returns on BMW Share Price*

Description

These data are the daily log returns on BMW share price from Tuesday 2nd January 1973 until Tuesday 23rd July 1996. The data are contained in a numeric vector. The dates of each observation are contained in a `times` attribute, which is an object of class `"POSIXct"` (see `DateTimeClasses`). Note that these data form an irregular time series because no trading takes place at the weekend.

Usage

```
data(bmw)
```

Format

A numeric vector containing 6146 observations, with a `times` attribute which is a `POSIXct` object of the same length.

---

danish *Danish Fire Insurance Claims*

Description

These data describe large fire insurance claims in Denmark from Thursday 3rd January 1980 until Monday 31st December 1990. The data are contained in a numeric vector. The dates of each observation are contained in a `times` attribute, which is an object of class `"POSIXct"` (see `DateTimeClasses`). They were supplied by Mette Rytgaard of Copenhagen Re. Note that these data form an irregular time series.

Usage

```
data(danish)
```

Format

A numeric vector containing 2167 observations, with a `times` attribute which is a `POSIXct` object of the same length.

---

decluster *Decluster Point Process*

Description

Declusters clustered point process data so that Poisson assumption is more tenable over a high threshold.

Usage

```
decluster(series, run = NA, picture = TRUE)
```

Arguments

`series`	a numeric vector of threshold exceedances with a `times` attribute which should be a numeric vector containing either the indices or the times/dates of each exceedance (if times/dates, the attribute should be an object of class `"POSIXct"` or an object that can be converted to that class; see `as.POSIXct`)
`run`	parameter to be used in the runs method; any two consecutive threshold exceedances separated by more than this number of observations/days are considered to belong to different clusters
`picture`	whether or not a picture of declustering should be drawn

Value

The declustered object.

References

Embrechts, P., Klueppelberg, C., Mikosch, T. (1997). *Modelling Extremal Events*. Springer. Chapter 8, 413–429.

See Also

pot, exindex, as.POSIXct

Examples

```
# decluster the 200 exceedances of a particular threshold in
# the negative BMW data
data(bmw)
out <- pot(-bmw, ne = 200)
decluster(out$data, 30)
```

---

dgev *Generalized Extreme Value Distribution*

Description

Cumulative probability, quantiles, density and random generation from the generalized extreme value distribution.

Usage

```
pgev(q, xi = 1, mu = 0, sigma = 1)
qgev(p, xi = 1, mu = 0, sigma = 1)
dgev(x, xi = 1, mu = 0, sigma = 1)
rgev(n, xi = 1, mu = 0, sigma = 1)
```

Arguments

q	vector of quantiles
p	vector of probabilities
x	vector of values at which to evaluate density
n	sample size
xi	shape parameter
mu	location parameter
sigma	scale parameter

Value

Probability (pgev), quantile (qgev), density (dgev) or random sample (rgev) for the GEV distribution with shape xi.

See Also

dgpd, gev

---

dgpd *Generalized Pareto Distribution*

Description

Cumulative probability, quantiles, density and random generation from the generalized Pareto distribution.

Usage

```
pgpd(q, xi, mu = 0, beta = 1)
qgpd(p, xi, mu = 0, beta = 1)
dgpd(x, xi, mu = 0, beta = 1)
rgpd(n, xi, mu = 0, beta = 1)
```

Arguments

q	vector of quantiles.
p	vector of probabilities.
x	vector of values at which to evaluate density
n	sample size
xi	shape parameter.
mu	location parameter.
beta	scale parameter

Value

Probability (pgpd), quantile (qgpd), density (dgpd) or random sample (rgpd) for the GPD distribution with shape xi.

See Also

dgev, gpd

emplot — *Plot of Empirical Distribution Function*

Description

Plots empirical distribution function of a sample.

Usage

```
emplot(data, alog = "x", labels = TRUE, ...)
```

Arguments

`data`	data vector
`alog`	whether axes are to be logged: `"x"` x-axis only; `"y"` y-axis only; `"xy"` both axes; `""` neither axis.
`labels`	whether or not axes should be labelled
`...`	other graphics parameters

Details

This is a simple explanatory function. A straight line on the double log scale indicates Pareto tail behaviour.

See Also

`qplot`, `meplot`

Examples

```
## Not run: data(danish)
## Not run: emplot(danish)
# Danish fire insurance data show Pareto tail behaviour
```

exindex — *Estimate Extremal Index*

Description

Plot estimates of extremal index using the blocks method.

Usage

```
exindex(data, block, start = 5, end = NA, reverse = FALSE,
    auto.scale = TRUE, labels = TRUE, ...)
```

Arguments

`data`	data vector (raw values not block maxima).
`block`	the block size. A numeric value is interpreted as the number of data values in each successive block. All the data is used, so the last block may not contain `block` observations. If the `data` has a `times` attribute containing (in an object of class `"POSIXct"`, or an object that can be converted to that class; see `as.POSIXct`) the times/dates of each observation, then `block` may instead take the character values `"month"`, `"quarter"`, `"semester"` or `"year"`.
`start`	lowest value of K at which to plot a point; K is the number of blocks in which a specified threshold is exceeded
`end`	highest value of K at which to plot a point
`reverse`	whether plot is to be by increasing threshold (`TRUE`) or increasing K value (`FALSE`)
`auto.scale`	whether or not plot should be automatically scaled; if not, `xlim` and `ylim` graphical parameters may be entered
`labels`	whether or not axes should be labelled
`...`	other graphics parameters

Value

A table of results is returned invisibly.

References

Embrechts, P., Klueppelberg, C., Mikosch, T. (1997). *Modelling Extremal Events*. Springer. Chapter 8, 413-429.

See Also

`gev`, `hill`, `as.POSIXct`

Examples

```
## Not run: data(bmw)
## Not run: exindex(bmw, 100)
## Not run: exindex(-bmw, 100)
# calculate extremal index for the right and left tails of the BMW
# log returns
```

findthresh *Find Threshold*

Description

Finds a threshold so that a given number of extremes lie above.

Usage

```
findthresh(data, ne)
```

Arguments

data	data vector
ne	vector giving number of extremes above the threshold

Details

When the data are tied a threshold is found so that at least the specified number of extremes lie above.

Value

Vector of suitable thresholds.

See Also

hill, gpd, pot

Examples

```
# Find threshold giving (at least) fifty exceedances for Danish data
data(danish)
findthresh(danish, 50)
```

gev *Fit Generalized Extreme Value Distribution*

Description

Fits generalized extreme value distribution (GEV) to block maxima data.

Usage

```
gev(data, block = NA, ...)
```

Arguments

`data`	data vector. Interpretation depends on value of block: if no block size is specified then data are interpreted as block maxima; if block size is set, then data are interpreted as raw data and block maxima are calculated.
`block`	the block size. A numeric value is interpreted as the number of data values in each successive block. All the data is used, so the last block may not contain `block` observations. If the `data` has a `times` attribute containing (in an object of class `"POSIXct"`, or an object that can be converted to that class; see `as.POSIXct`) the times/dates of each observation, then `block` may instead take the character values `"month"`, `"quarter"`, `"semester"` or `"year"`.
`...`	arguments passed to `optim`

Value

An object of class gev describing the fit and including parameter estimates and standard errors. Fitting is carried out using maximum likelihood.

See Also

`plot.gev`, `gumbel`, `optim`, `as.POSIXct`

Examples

```
# Fit GEV to monthly maxima
data(bmw)
out <- gev(bmw, "month")
# Fit GEV to maxima of blocks of 100 observations
out <- gev(bmw, 100)
# Fit GEV to the data in nidd.annual, the annual maximum water
# levels of the River Nidd, using the "BFGS" optimization method
data(nidd.annual)
out <- gev(nidd.annual, method = "BFGS", control = list(maxit = 500))
```

---

gpd *Fit Generalized Pareto Model*

Description

Returns an object of class "gpd" representing the fit of a generalized Pareto model to excesses over a high threshold.

Usage

```
gpd(data, threshold = NA, nextremes = NA, method = c("ml", "pwm"),
    information = c("observed", "expected"), ...)
```

Arguments

`data`	data vector
`threshold`	a threshold value (either this or `nextremes` must be given but not both)
`nextremes`	the number of upper extremes to be used (either this or `threshold` must be given but not both)
`method`	whether parameters should be estimated by the maximum likelihood method `"ml"` or the probability-weighted moments method `"pwm"`
`information`	whether standard errors should be calculated with `"observed"` or `"expected"` information. This only applies to the maximum likelihood method; for the probability-weighted moments method `"expected"` information is used if possible
`...`	arguments passed to `optim`

Details

The function uses the general purpose optimization function `optim` when `method = "ml"` is chosen.

Value

An object of class `"gpd"` describing the fit and including parameter estimates and standard errors.

References

Parameter and quantile estimation for the generalized Pareto distribution, JRM Hosking and JR Wallis, *Technometrics* **29**(3), pages 339-349, 1987.

See Also

`plot.gpd`, `shape`, `quant`, `optim`

Examples

```
data(danish)
out <- gpd(danish, 10)
# Fits GPD to excess losses over 10 for the Danish
# fire insurance data
```

`gpd.q` *Add Quantile Estimates to plot.gpd*

Description

Calculates quantile estimates and confidence intervals for high quantiles above the threshold in a GPD analysis, and adds a graphical representation to an existing plot.

Usage

```
gpd.q(x, pp, ci.type = c("likelihood", "wald"), ci.p = 0.95,
    like.num = 50)
```

Arguments

`x`	a list object returned by `plot.gpd` or `tailplot`
`pp`	the desired probability for quantile estimate (e.g. 0.99 for the 99th percentile)
`ci.type`	method for calculating a confidence interval: `"likelihood"` or `"wald"`
`ci.p`	probability for confidence interval (must be less than 0.999)
`like.num`	number of times to evaluate profile likelihood

Details

The GPD approximation in the tail is used to estimate quantile. The `"wald"` method uses the observed Fisher information matrix to calculate confidence interval. The `"likelihood"` method reparametrizes the likelihood in terms of the unknown quantile and uses profile likelihood arguments to construct a confidence interval.

See Also

`gpd`, `plot.gpd`, `gpd.sfall`, `tailplot`

Examples

```
## Not run: data(danish)
## Not run: out <- gpd(danish, 10)
## Not run: tp <- tailplot(out)
## Not run: gpd.q(tp, 0.999)
# Estimates 99.9th percentile of Danish fire losses
```

`gpd.sfall` *Add Expected Shortfall Estimates to a GPD Plot*

Description

Calculates expected shortfall (tail conditional expectation) estimates and confidence intervals for high quantiles above the threshold in a GPD analysis, and adds a graphical representation to an existing plot.

Usage

```
gpd.sfall(x, pp, ci.p = 0.95, like.num = 50)
```

Arguments

x	a list object returned by plot.gpd or tailplot
pp	the desired probability for expected shortfall estimate (e.g. 0.99 for the 99th percentile)
ci.p	probability for confidence interval (must be less than 0.999)
like.num	number of times to evaluate profile likelihood

Details

Expected shortfall is the expected size of the loss, given that a particular quantile of the loss distribution is exceeded. The GPD approximation in the tail is used to estimate expected shortfall. The likelihood is reparametrised in terms of the unknown expected shortfall and profile likelihood arguments are used to construct a confidence interval.

See Also

gpd, plot.gpd, tailplot, gpd.q

Examples

```
## Not run: data(danish)
## Not run: out <- gpd(danish, 10)
## Not run: tp <- tailplot(out)
## Not run: gpd.q(tp, 0.999)
# Estimates 99.9th percentile of Danish fire losses
## Not run: gpd.sfall(tp, 0.999)
# Estimates associated expected shortfall for Danish fire losses
```

---

gpdbiv *Implements Bivariate POT Method*

Description

Returns an object of class "gpdbiv" representing the fit of a bivariate POT (peaks over thresholds) model for joint excesses over thresholds.

Usage

```
gpdbiv(data1 = NA, data2 = NA, u1 = NA, u2 = NA, ne1 = NA, ne2 = NA,
    global = FALSE, method = "BFGS", ...)
```

Arguments

`data1`	first data vector
`data2`	second data vector
`u1`	threshold for `data1` (either this or `ne1` must be given but not both)
`u2`	threshold for `data2` (either this or `ne2` must be given but not both)
`ne1`	number of upper extremes to be used for `data1` (either this or `u1` must be given but not both)
`ne2`	number of upper extremes to be used for `data2` (either this or `u2` must be given but not both)
`global`	should a global maximisation of the likelihood with respect to marginal and dependence parameters be undertaken. The default alternative is a two-stage local fit where first the marginal parameters are estimated and then the dependence parameter. This is much faster than a global fit.
`method`	the optimization method (see `optim`). The argument has been created (as distinct from `...`) in order to make the `"BFGS"` method the default, as the default used by `optim` is not recommended for the one-dimensional optimizations that occur when `global = FALSE`.
`...`	other arguments passed to `optim`

Details

This function implements a model suggested by Richard Smith (see references below). The marginal excess distributions are GPD distributions, as suggested by univariate EVT and implemented in gpd. The dependence specification is known as the logistic or Gumbel dependence structure, but it would be easy to program alternatives.

Value

An object of class `"gpdbiv"` representing the fit and including parameter estimates and standard errors.

References

Multivariate Threshold Methods, Richard L. Smith, in *Extreme Value Theory and Applications*, ed. J. Galambos, published by Kluwer, pages 225-248, 1994.

Markov Chain Models for Threshold Exceedances, R.L. Smith, J.A. Tawn, S.G. Coles, *Biometrika* **84**, 249-268, 1997.

See Also

`gpd`, `plot.gpdbiv`, `interpret.gpdbiv`

Examples

```
data(bmw) ; data(siemens)
out <- gpdbiv(-bmw, -siemens, ne1 = 100, ne2 = 100)
interpret.gpdbiv(out, 0.05, 0.05)
## Not run: plot(out)
```

`gumbel` *Fit Gumbel Distribution*

Description

Fits gumbel distribution (GEV with `xi = 0`) to block maxima data.

Usage

```
gumbel(data, block = NA, ...)
```

Arguments

`data`	data vector. Interpretation depends on value of block: if no block size is specified then data are interpreted as block maxima; if block size is set, then data are interpreted as raw data and block maxima are calculated.
`block`	the block size. A numeric value is interpreted as the number of data values in each successive block. All the data is used, so the last block may not contain `block` observations. If the `data` has a `times` attribute containing (in an object of class `"POSIXct"`, or an object that can be converted to that class; see `as.POSIXct`) the times/dates of each observation, then `block` may instead take the character values `"month"`, `"quarter"`, `"semester"` or `"year"`.
`...`	arguments passed to `optim`

Details

This function is primarily intended for comparison with GEV for assessing the need for a heavy-tailed Frechet (or short-tailed Weibull) to model block maxima.

Value

An object of class `"gev"` describing the fit and including parameter estimates and standard errors. Fitting is carried out using maximum likelihood.

See Also

`plot.gev`, `gev`, `optim`, `as.POSIXct`

Examples

```
# Fit Gumbel to maxima of blocks of 100 observations
data(bmw)
out <- gumbel(bmw, 100)
# Fit Gumbel to the data in nidd.annual, the annual maximum water
# levels of the River Nidd, using the "BFGS" optimization method
data(nidd.annual)
out <- gumbel(nidd.annual, method = "BFGS", control = list(maxit = 500))
```

`hill` *Create Hill Plot*

Description

Plot the Hill estimate of the tail index of heavy-tailed data, or of an associated quantile estimate.

Usage

```
hill(data, option = c("alpha","xi","quantile"), start = 15,
    end = NA, reverse = FALSE, p = NA, ci = 0.95,
    auto.scale = TRUE, labels = TRUE, ...)
```

Arguments

`data`	data vector
`option`	whether `"alpha"`, `"xi"` (1/alpha) or `"quantile"` (a quantile estimate) should be plotted
`start`	lowest number of order statistics at which to plot a point
`end`	highest number of order statistics at which to plot a point
`reverse`	whether plot is to be by increasing threshold (`TRUE`) or increasing number of order statistics (`FALSE`)
`p`	probability required when option `"quantile"` is chosen
`ci`	probability for asymptotic confidence band; for no confidence band set `ci` to zero
`auto.scale`	whether or not plot should be automatically scaled; if not, `xlim` and `ylim` graphical parameters may be entered
`labels`	whether or not axes should be labelled
`...`	other graphics parameters

Details

This plot is usually calculated from the alpha perspective. For a generalized Pareto analysis of heavy-tailed data using the gpd function, it helps to plot the Hill estimates for xi.

See Also

`shape`, `quant`

Examples

```
## Not run: data(danish)
## Not run: hill(danish)
# Hill plot of heavy-tailed Danish fire insurance data
## Not run: hill(danish, option = "quantile", end = 500, p = 0.999)
# Hill plot of estimated 0.999 quantile of Danish fire insurance data
```

---

interpret.gpdbiv *Interpret Results of Bivariate GPD Fit*

Description

Interprets the results of a bivariate GPD model fitted using the bivariate POT method.

Usage

```
interpret.gpdbiv(out, x, y)
```

Arguments

`out`	a `gpdbiv` object
`x`	a scalar value greater than first threshold
`y`	a scalar value greater than second threshold

Details

First marginal probabilities of exceeding the points x and y are calculated, and then joint and conditional probabilities.

Value

A vector of probabilities is invisibly returned, in printed order.

Side Effects

A simple interpretation of the fit in terms of exceedance probabilities for the point (x,y) is printed.

See Also

`gpdbiv`, `plot.gpdbiv`

Examples

```
data(bmw) ; data(siemens)
out <- gpdbiv(-bmw, -siemens, ne1 = 100, ne2 = 100)
interpret.gpdbiv(out, 0.05, 0.05)
# probabilities of 5% falls in BMW and Siemens stock prices
```

---

meplot *Sample Mean Excess Plot*

Description

Plots sample mean excesses over increasing thresholds.

Usage

```
meplot(data, omit = 3, labels = TRUE, ...)
```

Arguments

`data`	data vector
`omit`	number of upper plotting points to be omitted
`labels`	whether or not axes are to be labelled
`...`	other graphics parameters

Details

An upward trend in plot shows heavy-tailed behaviour. In particular, a straight line with positive gradient above some threshold is a sign of Pareto behaviour in tail. A downward trend shows thin-tailed behaviour whereas a line with zero gradient shows an exponential tail. Because upper plotting points are the average of a handful of extreme excesses, these may be omitted for a prettier plot.

See Also

`gpd`, `qplot`

Examples

```
## Not run: data(danish)
## Not run: meplot(danish)
# Sample mean excess plot of heavy-tailed Danish fire insurance data
```

nidd.annual *The River Nidd Data*

Description

These data represent annual maximal levels of the River Nidd in Yorkshire. These data are suitable for analysis with gev.

Usage

```
data(nidd.annual)
```

Format

A numeric vector containing 35 observations.

nidd.thresh *The River Nidd Data*

Description

These data represent high river levels of the River Nidd in Yorkshire above a threshold value of 65. These data are suitable for analysis with gpd.

Usage

```
data(nidd.thresh)
```

Format

A numeric vector containing 154 observations.

plot.gev *Plot Fitted GEV Model*

Description

The plot method plot.gev provides two different residual plots for assessing fitted GEV model. The user selects the plot type from a menu. See the examples below.

Usage

```
## S3 method for class 'gev'
plot(x, ...)
```

Arguments

x a gev object

... other graphics parameters

Details

Data are converted to unit exponentially distributed residuals under null hypothesis that GEV fits. Two diagnostics for iid exponential data are offered.

See Also

gev, qplot

Examples

```
data(bmw)
out <- gev(bmw, 100)
## Not run: plot(out)

## Not run: Make a plot selection (or 0 to exit):
## Not run: 1: plot: Scatterplot of Residuals
## Not run: 2: plot: QQplot of Residuals
```

---

plot.gpd *Plot Fitted GPD Model*

Description

The plot method plot.gpd provides four different plots for assessing fitted GPD model. The user selects the plot type from a menu. See the examples below.

Usage

```
## S3 method for class 'gpd'
plot(x, optlog = NA, extend = 1.5, labels = TRUE, ...)
```

Arguments

x a gpd object

optlog optional argument for plots 1 and 2 giving a particular choice of logarithmic axes: "x" x-axis only; "y" y-axis only; "xy" both axes; "" neither axis.

extend optional argument for plots 1 and 2 expressing how far x-axis should extend as a multiple of the largest data value. This argument must take values greater than 1 and is useful for showing estimated quantiles beyond data.

labels optional argument for plots 1 and 2 specifying whether or not axes should be labelled

... other graphics parameters

Value

If plot 1 or 2 is selected as the final plot, a list object containing details of the plot is returned invisibly. This object should be used as the first argument of `gpd.q` or `gpd.sfall` to add quantile estimates or expected shortfall estimates to the plot.

See Also

`gpd`, `quant`, `shape`

Examples

```
data(danish)
out <- gpd(danish, 10)
## Not run: plot(out)

## Not run: Make a plot selection (or 0 to exit):
## Not run: 1: plot: Excess Distribution
## Not run: 2: plot: Tail of Underlying Distribution
## Not run: 3: plot: Scatterplot of Residuals
## Not run: 4: plot: QQplot of Residuals
```

---

plot.gpdbiv — *Plot Fitted Bivariate GPD Model*

Description

Provides a number of plots summarising a bivariate GPD model fitted using the bivariate POT method. See the examples below.

Usage

```
## S3 method for class 'gpdbiv'
plot(x, extend = 1.1, n.contours = 15, ...)
```

Arguments

`x`	a gpdbiv object
`extend`	optional argument expressing how far x-axis should extend as a multiple of the largest data value.
`n.contours`	number of contours in bivariate contour plots
`...`	other graphics parameters

Details

Option 1 plots the threshold exceedance data; option 2 plots contours of the fitted bivariate distribution function in the joint upper tail (above both thresholds); option 3 plots corresponding contours of the fitted joint survival function; plots 4 and 5 show the fitted tails of the marginal distributions.

See Also

gpd, gpdbiv, tailplot, interpret.gpdbiv, plot.gpd

Examples

```
data(bmw) ; data(siemens)
out <- gpdbiv(-bmw, -siemens, ne1 = 100, ne2 = 100)
## Not run: plot(out)

## Not run: Make a plot selection (or 0 to exit):
## Not run: 1: plot: Exceedance data
## Not run: 2: plot: Contours of Bivariate Distribution Function
## Not run: 3: plot: Contours of Bivariate Survival Function
## Not run: 4: plot: Tail of Marginal 1
## Not run: 5: plot: Tail of Marginal 2
```

---

plot.potd *Plot Fitted POT Model*

Description

The plot method plot.potd provides seven different plots for assessing fitted POT model. The user selects the plot type from a menu. See the examples below.

Usage

```
## S3 method for class 'potd'
plot(x, ...)
```

Arguments

`x`	an object returned by the function `pot`
`...`	other graphics parameters

Details

Plot 1 displays the exceedance process of the chosen threshold. Plots 2-4 assess the Poisson nature of the exceedance process by looking at the scaled gaps between exceedances, which should be iid unit exponentially distributed. Plots 5-6 assess the GPD nature of the excesses by looking at suitably defined residuals, which should again be iid unit exponentially distributed. Option 8 allows the user to call GPD plotting functions.

Value

If plot 1 or 2 from the GPD plots is selected as the final plot (i.e. option 8 is selected, followed by option 1 or 2), a list object containing details of the plot is returned invisibly. This object should be used as the first argument of `gpd.q` or `gpd.sfall` to add quantile estimates or expected shortfall estimates to the plot.

See Also

gpd, pot, plot.gpd

Examples

```
data(danish)
out <- pot(danish,10)
## Not run: plot(out)

## Not run: Make a plot selection (or 0 to exit):
## Not run: 1: plot: Point Process of Exceedances
## Not run: 2: plot: Scatterplot of Gaps
## Not run: 3: plot: Qplot of Gaps
## Not run: 4: plot: ACF of Gaps
## Not run: 5: plot: Scatterplot of Residuals
## Not run: 6: plot: Qplot of Residuals
## Not run: 7: plot: ACF of Residuals
## Not run: 8: plot: Go to GPD Plots
```

---

pot *Peaks Over Thresholds Model*

Description

Fits a Poisson point process to the data, an approach sometimes known as peaks over thresholds (POT), and returns an object of class "potd".

Usage

```
pot(data, threshold = NA, nextremes = NA, run = NA, picture = TRUE,
    ...)
```

Arguments

data	numeric vector of data, which may have a `times` attribute containing (in an object of class "POSIXct", or an object that can be converted to that class; see as.POSIXct) the times/dates of each observation. If no `times` attribute exists, the data are assumed to be equally spaced.
threshold	a threshold value (either this or `nextremes` must be given but not both)
nextremes	the number of upper extremes to be used (either this or `threshold` must be given but not both)
run	if the data are to be declustered the run length parameter for the runs method (see decluster) should be entered here
picture	whether or not a picture should be drawn if declustering is performed
...	arguments passed to `optim`

Details

Uses optim for point process likelihood maximization.

Value

An object of class "potd" describing the fit and including parameter estimates and standard errors.

See Also

gpd, plot.potd, plot.gpd, decluster, optim, as.POSIXct

Examples

```
data(danish)
out <- pot(danish, 10)
# Fits POT model to Danish fire insurance losses
```

---

qplot *Exploratory QQplot for Extreme Value Analysis*

Description

Creates a QQplot for threshold data against the exponential distribution or the generalized Pareto distribution.

Usage

```
qplot(data, xi = 0, trim = NA, threshold = NA, line = TRUE,
    labels = TRUE, ...)
```

Arguments

data	data vector
xi	the xi value of a generalized Pareto distribution
trim	value at which data are to be right-truncated
threshold	value at which data are to be left-truncated
line	whether or not a straight line is to be added
labels	whether or not the axes are to be labelled
...	other graphics parameters

Details

If xi is zero the reference distribution is the exponential; if xi is non-zero the reference distribution is the generalized Pareto with that value of xi. In the case of the exponential, the plot is interpreted as follows. Concave departures from a straight line are a sign of heavy-tailed behaviour. Convex departures show thin-tailed behaviour.

See Also

gpd, meplot

Examples

```
## Not run: data(danish)
## Not run: qplot(danish)
# QQplot of heavy-tailed Danish fire insurance data
```

---

quant *Plot of GPD Tail Estimate of a High Quantile*

Description

Creates a plot showing how the estimate of a high quantile in the tail of a dataset based on the GPD approximation varies with threshold or number of extremes.

Usage

```
quant(data, p = 0.99, models = 30, start = 15, end = 500, reverse =
    TRUE, ci = 0.95, auto.scale = TRUE, labels = TRUE, ...)
```

Arguments

`data`	numeric vector of data
`p`	desired probability for quantile estimate (e.g. 0.99 gives 99th percentile)
`models`	number of consecutive gpd models to be fitted
`start`	lowest number of exceedances to be considered
`end`	maximum number of exceedances to be considered
`reverse`	should plot be by increasing threshold (`TRUE`) or number of extremes (`FALSE`)
`ci`	probability for asymptotic confidence band; for no confidence band set to zero
`auto.scale`	whether or not plot should be automatically scaled; if not, xlim and ylim graphical parameters may be entered
`labels`	whether or not axes should be labelled
`...`	other graphics parameters

Details

For every model gpd is called. Evaluation may be slow. Confidence intervals by the Wald method (which is fastest).

Value

A table of results is returned invisibly.

See Also

gpd, plot.gpd, gpd.q, shape

Examples

```
## Not run: data(danish)
## Not run: quant(danish, 0.999)
# Estimates of the 99.9th percentile of the Danish losses using
# the GPD model with various thresholds
```

---

records — *Calculate Record Development*

Description

Creates a data frame showing the development of records in a dataset and calculating the expected behaviour for iid data.

Usage

```
records(data, do.plot = TRUE, conf.level = 0.95, ...)
```

Arguments

`data`	data vector
`do.plot`	whether a plot of record development should be created
`conf.level`	confidence level for record development plot
`...`	graphics parameters

Details

Records are counted and the observations at which they occur recorded. This is compared with the expected behaviour for iid data.

Value

A data frame.

Examples

```
## Not run: data(danish)
## Not run: records(danish)
# Record fire insurance losses in Denmark
```

riskmeasures *Calculates Quantiles and Expected Shortfalls*

Description

Makes a rapid calculation of point estimates of prescribed quantiles and expected shortfalls using the output of the function gpd.

Usage

```
riskmeasures(x, p)
```

Arguments

x	results of a gpd fit
p	a vector of probability levels

Details

This function simply calculates point estimates and (at present) makes no attempt to calculate confidence intervals for the risk measures. If confidence levels are required use `gpd.q` and `gpd.sfall` which interact with graphs of the tail of a loss distribution and are much slower.

Value

A matrix with three columns: probability level, quantile estimate, shortfall estimate.

See Also

`gpd`, `tailplot`, `gpd.q`, `gpd.sfall`

Examples

```
data(danish)
out <- gpd(danish, 10)
riskmeasures(out, c(0.999, 0.9999))
# gives estimates of 0.999 and 0.9999 quantiles of Danish loss
# distribution as well as the associated expected shortfall estimates
```

rlevel.gev *Calculate Return Levels Based on GEV Fit*

Description

Calculates the k-block return level and 95% confidence interval based on a GEV model for block maxima, where k is specified by the user. The k-block return level is that level exceeded once every k blocks, on average.

Usage

```
rlevel.gev(out, k.blocks = 20, add = FALSE, ...)
```

Arguments

`out`	an object returned by the function gev
`k.blocks`	specifies the particular return level to be estimated; default set arbitrarily to 20
`add`	whether the return level should be added graphically to a time series plot; if `FALSE` a graph of the profile likelihood curve showing the return level and its confidence interval is produced
`...`	other graphics parameters

Details

The GEV likelihood is reparameterized in terms of the unknown return level and profile likelihood arguments are used to construct a confidence interval.

Value

Vector containing lower 95% bound of confidence interval, estimated return level and upper 95% bound.

See Also

`gev`, `plot.gev`

Examples

```
data(bmw)
out <- gev(bmw, "month")
# Fit GEV to monthly maxima of daily returns on BMW share price
## Not run: rlevel.gev(out, 40)
# Calculate the 40 month return level
```

shape *Plot for GPD Shape Parameter*

Description

Creates a plot showing how the estimate of shape varies with threshold or number of extremes.

Usage

```
shape(data, models = 30, start = 15, end = 500, reverse = TRUE,
    ci = 0.95, auto.scale = TRUE, labels = TRUE, ...)
```

Arguments

`data`	numeric vector of data
`models`	number of consecutive gpd models to be fitted
`start`	lowest number of exceedances to be considered
`end`	maximum number of exceedances to be considered
`reverse`	should plot be by increasing threshold (`TRUE`) or number of extremes (`FALSE`)
`ci`	probability for asymptotic confidence band; for no confidence band set to zero
`auto.scale`	whether or not plot should be automatically scaled; if not, `xlim` and `ylim` graphical parameters may be entered
`labels`	whether or not axes should be labelled
`...`	other graphics parameters

Details

For every model gpd is called. Evaluation may be slow.

Value

A table of results is returned invisibly.

See Also

`gpd`, `plot.gpd`, `hill`

Examples

```
## Not run: data(danish)
## Not run: shape(danish)
# Shape plot of heavy-tailed Danish fire insurance data
```

siemens — *Daily Log Returns on Siemens Share Price*

Description

These data are the daily log returns on Siemens share price from Tuesday 2nd January 1973 until Tuesday 23rd July 1996. The data are contained in a numeric vector. The dates of each observation are contained in a `times` attribute, which is an object of class `"POSIXct"` (see `DateTimeClasses`). Note that these data form an irregular time series because no trading takes place at the weekend.

Usage

```
data(siemens)
```

Format

A numeric vector containing 6146 observations, with a `times` attribute which is a `POSIXct` object of the same length.

sp.raw — *SP Data to June 1993*

Description

The daily closing values of the S&P index from Monday 4th January 1960 until Friday 11th June 1993. The data are contained in a numeric vector. The dates of each observation are contained in a `times` attribute, which is an object of class `"POSIXct"` (see `DateTimeClasses`).

Usage

```
data(sp.raw)
```

Format

A numeric vector containing 8415 observations, with a `times` attribute which is a `POSIXct` object of the same length.

`spto87` *SP Return Data to October 1987*

Description

The daily log returns on the S&P index value from Tuesday 5th January 1960 until Friday 16 October 1987. The data are contained in a numeric vector. The dates of each observation are contained in a `times` attribute, which is an object of class `"POSIXct"` (see `DateTimeClasses`).

Usage

```
data(spto87)
```

Format

A numeric vector containing 6985 observations, with a `times` attribute which is a `POSIXct` object of the same length.

`tailplot` *Plot Tail Estimate From GPD Model*

Description

Interacts with the output of gpd to produce a plot of the tail of the underlying distribution of the data. This is one of the options of `plot.gpd`, but tailplot enables the user to bypass the menu of the former.

Usage

```
tailplot(x, optlog = NA, extend = 1.5, labels = TRUE, ...)
```

Arguments

`x`	a gpd object
`optlog`	optional argument giving a particular choice of logarithmic axes: `"x"` x-axis only; `"y"` y-axis only; `"xy"` both axes; `""` neither axis.
`extend`	optional argument expressing how far x-axis should extend as a multiple of the largest data value. This argument must take values greater than 1 and is useful for showing estimated quantiles beyond data.
`labels`	optional argument specifying whether or not axes should be labelled
`...`	other graphics parameters

Value

A list object containing details of the plot is returned invisibly. This object should be used as the first argument of `gpd.q` or `gpd.sfall` to add quantile estimates or expected shortfall estimates to the plot.

See Also

`gpd`, `plot.gpd`, `shape`, `quant`

Examples

```
data(danish)
out <- gpd(danish, 10)
## Not run: tailplot(out)
```

Index