Package 'POT'

April 18, 2019

Version 1.1-7

Title Generalized Pareto Distribution and Peaks Over Threshold

Author Mathieu Ribatet [aut], Christophe Dutang [ctb]

Maintainer Christophe Dutang <christophe.dutang@ensimag.fr>

Depends R (>= 3.0.0)

Description Some functions useful to perform a Peak Over Threshold analysis in univariate and bivariate cases, see Beirlant et al. (2004) <doi:10.1002/0470012382>. A user's guide is available.

License GPL (>= 2)

URL http://pot.r-forge.r-project.org/

Repository CRAN

NeedsCompilation yes

Date/Publication 2019-04-18 21:30:15 UTC

R topics documented:

`anova.bvpot` *Anova Tables: Bivariate Case*

Description

Computes analysis of deviance for "bvpot" object

Usage

```
## S3 method for class 'bvpot'
anova(object, object2, ..., half = FALSE)
```

Arguments

`object, object2`	Two objects of class "bvpot", most often return of the `fitbvgpd` function.
`...`	Other options to be passed to the `anova` function.
`half`	Logical. For some non-regular testing problems the deviance difference is known to be one half of a chi-squared random variable. Set half to `TRUE` in these cases.

Value

This function returns an object of class anova. These objects represent analysis-of-deviance tables.

Warning

Circumstances may arise such that the asymptotic distribution of the test statistic is not chi-squared. In particular, this occurs when the smaller model is constrained at the edge of the parameter space. It is up to the user recognize this, and to interpret the output correctly.

In some cases the asymptotic distribution is known to be one half of a chi-squared; you can set `half = TRUE` in these cases.

Author(s)

Mathieu Ribatet (Alec Stephenson for the "Warning" case)

See Also

`anova`, `anova.uvpot`

Examples

```
x <- rgpd(1000, 0, 1, -0.25)
y <- rgpd(1000, 2, 0.5, 0)
M0 <- fitbvgpd(cbind(x,y), c(0, 2))
M1 <- fitbvgpd(cbind(x,y), c(0,2), model = "alog")
anova(M0, M1)

##Non regular case
M0 <- fitbvgpd(cbind(x,y), c(0, 2))
M1 <- fitbvgpd(cbind(x,y), c(0, 2), alpha = 1)
anova(M0, M1, half = TRUE)
```

anova.uvpot *Anova Tables: Univariate Case*

Description

Computes analysis of deviance for "uvpot" object

Usage

```
## S3 method for class 'uvpot'
anova(object, object2, ...)
```

Arguments

object, object2	Two objects of class "uvpot", most often return of the fitgpd function.
...	Other options to be passed to the anova function.

Value

This function returns an object of class anova. These objects represent analysis-of-deviance tables.

Author(s)

Mathieu Ribatet

See Also

anova, anova.bvpot

Examples

```
x <- rgpd(1000, 0, 1, -0.15)
M0 <- fitgpd(x, 0, shape = -0.15)
M1 <- fitgpd(x, 0)
anova(M0, M1)
```

bvgpd *Parametric Bivariate GPD*

Description

Density, distribution function and random generation for six different parametric bivariate GPD

Usage

```
rbvgpd(n, alpha, model = "log", asCoef, asCoef1, asCoef2, mar1 =
c(0,1,0), mar2 = mar1)
pbvgpd(q, alpha, model = "log", asCoef, asCoef1, asCoef2, mar1 =
c(0,1,0), mar2 = mar1, lower.tail = TRUE)
```

Arguments

n	The number of observations to be simulated.
q	A matrix or vector with two columns at which the distribution is computed.
alpha	Dependence parameter for the logistic, asymmetric logistic, negative logistic, asymmetric negative logistic, mixed and asymmetric mixed models.
model	The specified model; a character string. Must be either "log" (the default), "alog", "nlog", "anlog", "mix" or "amix", for the logistic, asymmetric logistic, negative logistic, asymmetric negative logistic, mixed and asymmetric mixed models respectively.
asCoef,asCoef1,asCoef2	The asymmetric coefficients for the asymmetric logistic, asymmetric negative logistic and asymmetric mixed models.
mar1,mar2	Vectors of length 3 giving the marginal parameters.
lower.tail	Logical. If TRUE (the default), P[X $\leq$ x] is computed, else P[X $\geq$ x].

Details

The logistic and asymmetric logistic models respectively are simulated using bivariate versions of Algorithms 1.1 and 1.2 in Stephenson(2003). All other models are simulated using a root finding algorithm to simulate from the conditional distributions.

Value

Generate a random vector of length n.

Author(s)

Mathieu Ribatet (Alec Stephenson for the C codes)

References

Stephenson, A. G. (2003) Simulating multivariate extreme value distributions of logistic type. *Extremes*, **6**(1), 49–60.

Examples

```
x <- rbvgpd(1000, alpha = 0.25, model = "log", mar1 = c(0,1,0.25), mar2
= c(2,0.5, -0.15))
y <- rbvgpd(1000, alpha = 0.75, model = "nlog", mar1 = c(0,1,0.25), mar2
= c(2,0.5, -0.15))
par(mfrow=c(1,2))
plot(x);plot(y)
```

---

chimeas *Dependence Measures For Extreme Values Analysis*

Description

Provide two measures to assess for asymptotic dependence or independence

Usage

```
chimeas(data, u.range, n.u = 500, xlab, ylabs, ci = 0.95,  boot = FALSE,
n.boot = 250, block.size = 50, show.bound = TRUE, which = 1:2, ask =
nb.fig < length(which) && dev.interactive(), ..., col.ci = "grey",
col.bound = "blue", lty.ci = 1, lty.bound = 1)
```

Arguments

data	A matrix with 2 columns with the data.
u.range	Numeric vection of length 2 (may be missing): the range for the probabilities.
n.u	The number of probabilities to be considered
xlab,ylabs	The x-axis and ylabs labels. ylabs must be of length 2
ci	The probability level for the confidence intervals
boot	Logical. If TRUE, confidence intervals are computed by bootstraping contiguous blocks. This may be needed if there is dependence between observations. If FALSE (the default), confidence intervals are derived using the Delta method.
n.boot	The number of bootstrap replicates.
block.size	The size of the "contiguous" blocks. See details.
show.bound	Logical. If TRUE (the default), the theoretical bound for the two statistics are plotted.
which	Which plot should be plotted? 1 for the χ 2 for the $\overline{\chi}$ statistic and 1:2 for both of them.
ask	Logical. Should user be asked before each plot is computed?

`...` Additional options to be passed to the `plot` function.

`col.ci,col.bound`
The color for the confidence intervals and theoretical bounds.

`lty.ci,lty.bound`
The line type for the confidence intervals and theoretical bounds.

Details

These two plots help us to understand the dependence relationship between the two data set. The sign of $\chi(u)$ determines if the variables are positively or negatively correlated. Two variable are asymptotically independent if $\lim_{u\to 1}\chi(u) = 0$. For the independent case, $\chi(u) = 0$ for all u in (0,1). For the perfect dependence case, $\chi(u) = 1$ for all u in (0,1). Note that for a bivariate extreme value model, $\chi(u) = 2(1 - A(0.5))$ for all u in (0,1).

The measure $\overline{\chi}$ is only useful for asymptotically independent variables. Indeed, for asymptotically dependent variable, we have $\lim_{u\to 1}\overline{\chi}(u) = 1$. For asymptotically independent variables, $\lim_{u\to 1}\overline{\chi}(u)$ reflects the strength of the dependence between variables. For independent variables, $\overline{\chi}(u) = 0$ for all u in (0,1).

If there is (short range) dependence between observations, users may need to use bootstrap confidence intervals. Bootstrap series are obtained by sampling contiguous blocks, of length `l` say, uniformly with replacement from the original observations. The block length `l` should be chosen to be much greater than the short-range dependence and much smaller than the total number of observations.

Value

A graphic window.

Author(s)

Mathieu Ribatet

References

Coles, S., Heffernan, J. and Tawn, J. (1999) Dependence measures for extreme value analyses. *Extremes* **2** 339–365.

See Also

`tailind.test`, `specdens`, `tsdep.plot`

Examples

```
mc <- simmc(200, alpha = 0.9)
mc2 <- simmc(100, alpha = 0.2)
##An independent case
par(mfrow = c(1,2))
chimeas(cbind(mc[1:100], mc2))
##Asymptotic dependence
par(mfrow = c(1,2))
chimeas(cbind(mc[seq(1,200, by = 2)], mc[seq(2,200,by = 2)]))
```

```
##The same but with bootstrap ci
par(mfrow = c(1,2))
chimeas(cbind(mc[seq(1,200, by = 2)], mc[seq(2,200,by = 2)]), boot =
TRUE, n.boot=50)
```

---

clust *Identify Extreme Clusters within a Time Series*

Description

A function to identify clusters of exceedances of a time series.

Usage

```
clust(data, u, tim.cond = 1, clust.max = FALSE, plot = FALSE,
only.excess = TRUE, ...)
```

Arguments

`data`	A matrix/data.frame with two columns. Columns names **must be** `obs` for observations and `time` for the associated date of each observation.
`u`	Numeric. A value giving the threshold.
`tim.cond`	A time condition to ensure independence between events. Should be in the same unit than `data[,"time"]`.
`clust.max`	Logical. If `FALSE` (the default), a list containing the clusters of exceedances is returned. Else, a matrix containing the cluster maxima and related dates is returned.
`plot`	Logical. If `TRUE`, identified clusters are displayed. Else (the default), no plot is produced.
`only.excess`	Logical. If `TRUE` (the default), only exceedances are plotted. Else, all observations are displayed.
`...`	Optional parameters to be passed in `plot` function.

Details

The clusters of exceedances are defined as follows:

- The first exceedance initiates the first cluster;
- The first observation under the threshold `u` "ends" the current cluster unless `tim.cond` does not hold;
- The next exceedance initiates a new cluster;
- The process is iterated as needed.

This function differs from the function `clusters` of `evd` Package as independence condition i.e. `tim.cond` could be a "temporal" condition. That is, two events are considered independent if the inter-arrival time is greater than a fixed duration.

However, it is also possible to used the "index" independence as in `clust` by setting `data[,"time"] = 1:length(data[,"obs"])`.

Value

If `clust.max` is `FALSE`, a list containing the clusters of exceedances is returned. Else, a matrix containing the cluster maxima, related dates and indices are returned.

In any case, the returned object has an attribute `exi` giving an estimation of the Extremal Index, that is the inverse of the average cluster size.

Author(s)

Mathieu Ribatet

See Also

`clusters` of package `evd`.

Examples

```
data(ardieres)
par(mfrow=c(1,2))
clust(ardieres, 4, 10 / 365)
clust(ardieres, 4, 10 / 365, clust.max = TRUE)
clust(ardieres, 4, 10 / 365, clust.max = TRUE, plot = TRUE)
##The same but with optional arguments passed to function ``plot''
clust(ardieres, 4, 10 / 365, clust.max = TRUE, plot = TRUE,
xlab = "Time (Years)", ylab = "Flood discharges",
xlim = c(1972, 1980))
```

`Clusters` *Extremal Index Plot*

Description

Plot estimates of the Extremal Index

Usage

```
exiplot(data, u.range, tim.cond = 1, n.u = 50, xlab, ylab, ...)
```

Arguments

`data`	A matrix/data.frame with two columns. Columns names **must be** `obs` for observations and `time` for the associated date of each observation.
`u.range`	A numeric vector of length 2. Specify the range of threshold for which the Extremal Index is estimated.
`tim.cond`	A time condition to ensure independence between events. Should be in the same unit that `data[,"time"]`.
`n.u`	Numeric. The number of thresholds at which the Extremal Index is estimated.
`xlab, ylab`	Optional character strings to label the x and y axis.
`...`	Optional options to be passed to the `plot` function.

Value

Returns invisibly a matrix with two columns. The first one `thresh` giving the threshold and the second one `exi` the related Extremal Index estimate.

Author(s)

Mathieu Ribatet

See Also

`clust`

`coef.pot` *Extract model coefficients of a* `'pot'` *model*

Description

`coef` extracts model coefficients of an object of class `'pot'`

Usage

```
## S3 method for class 'pot'
coef(object, ...)
```

Arguments

`object`	An object of class `'pot'`. Most often, this is an object return by the `fitgpd`, `fitbvgpd` and `fitmcgpd` functions.
`...`	Other arguments to be passed to the `str` function.

Value

Standard `coef` object: see `coef`.

Author(s)

Christophe Dutang

See Also

`coef`

Examples

```
set.seed(123)
x <- rgpd(500, 0, 1, -0.15)
mle <- fitgpd(x, 0)
coef(mle)
```

confint.uvpot *Generic Function to Compute (Profile) Confidence Intervals*

Description

Compute (profile) confidence intervals for the scale, shape GPD parameters and also for GPD quantiles.

Usage

```
## S3 method for class 'uvpot'
confint(object, parm, level = 0.95, ..., range, prob,
 prof = TRUE)
```

Arguments

object	R object given by function fitgpd.
parm	Charater string specifies for which variable confidence intervals are computed. One of "quant", "scale" or "shape".
level	Numeric. The confidence level.
...	Optional parameters. See details.
range	Vector of dimension two. It gives the lower and upper bound on which the profile likelihood is performed. Only required when "prof = TRUE".
prob	The probability of non exceedance.
prof	Logical. If TRUE (the default), profile confidence intervals are computed. Otherwise, it is Fisher ones.

Details

Additional options can be passed using "..." in the function call. Possibilites are related to the specific functions: link{gpd.fiscale}, link{gpd.fishape}, link{gpd.firl}, link{gpd.pfscale}, link{gpd.pfshape}, link{gpd.pfrl}.

Value

Returns a vector of the lower and upper bound for the (profile) confidence interval. Moreover, a graphic of the profile likelihood function is displayed when prof = TRUE.

Author(s)

Mathieu Ribatet

See Also

link{gpd.fiscale}, link{gpd.fishape}, link{gpd.firl}, link{gpd.pfscale}, link{gpd.pfshape} and link{gpd.pfrl}

Examples

```
x <- rgpd(100, 0, 1, 0.25)
mle <- fitgpd(x, 0)
confint(mle, prob = 0.2)
confint(mle, parm = "shape")
```

`convassess` *Convergence Assessment for Fitted Objects*

Description

`convassess` is a generic function used to assess the convergence of the estimation procedure to the global maximum. The function invokes particular `methods` which depend on the `class` of the first argument. This function uses several starting values to assess the sensitiveness of the fitted object with respect to starting values.

Usage

```
convassess(fitted, n = 50)

## S3 method for class 'uvpot'
convassess(fitted, n = 50)
## S3 method for class 'mcpot'
convassess(fitted, n = 50)
## S3 method for class 'bvpot'
convassess(fitted, n = 50)
```

Arguments

`fitted`	A fitted object. When using the POT package, an object of class `'uvpot'`, `'mcpot'` or `'bvpot'`. Generally, an object return by `fitgpd`, `fitmcgpd` or `fitbvgpd`.
`n`	The number of starting values to be tested.

Details

The starting values are defined using the unbiased probability weighted moments fitted on `n` bootstrap samples.

Value

Graphics: the considered starting values, the objective values derived from numerical optimizations and traceplots for all estimated parameters. In addition, it returns invisibly all these informations.

Author(s)

Mathieu Ribatet

See Also

fitgpd, fitmcgpd, fitbvgpd

Examples

```
##Univariate Case
x <- rgpd(30, 0, 1, 0.2)
med <- fitgpd(x, 0, "med")
convassess(med)
##Bivariate Case
x <- rbvgpd(50, model = "log", alpha = 0.5, mar1 = c(0, 1, 0.2))
log <- fitbvgpd(x, c(0,0))
convassess(log)
```

---

dens *Density Plot: Univariate Case*

Description

dens is a generic function used to plot the density. The function invokes particular methods which depend on the class of the first argument. So the function plots density for univariate POT models.

Usage

```
dens(fitted, ...)

## S3 method for class 'uvpot'
dens(fitted, main, xlab, ylab, dens.adj = 1,
kern.lty = 2, rug = TRUE, plot.kernel = TRUE, plot.hist = TRUE,
hist.col = NULL, ...)
```

Arguments

fitted	A fitted object. When using the POT package, an object of class 'uvpot'. Most often, the return of the fitgpd function.
main	The title of the graphic. If missing, the title is set to "Density Plot".
xlab,ylab	The labels for the x and y axis. If missing, they are set to "Quantile" and "Density" respectively.
dens.adj	Numeric. The adjustment for the kernel density estimation in the density function. The default is 1.
kern.lty	The line type for the kernel density estimation. This corresponds to the "lty" option of the lines functions. The default is 2.
rug	Logical. Should we call the rug function? Default is TRUE.
plot.kernel	Logical. Should the kernel density estimate be plotted?
plot.hist	Logical. Should the histogram be plotted?
hist.col	The color to fill the histogram.
...	Other arguments to be passed to the plot function.

Details

The density plot consists of plotting on the same windows the theoretical density and a kernel estimation one. If the theoretical model is correct, then the two densities should be "similar".

Value

A graphical window.

Author(s)

Mathieu Ribatet

See Also

dens, dens.uvpot

Examples

```
x <- rgpd(75, 1, 2, 0.1)
pwmu <- fitgpd(x, 1, "pwmu")
dens(pwmu)
```

---

dexi *Compute the Density of the Extremal Index*

Description

Compute the density of the extremal index using simulations from a fitted markov chain model.

Usage

```
dexi(x, n.sim = 1000, n.mc = length(x$data), plot = TRUE, ...)
```

Arguments

x	A object of class 'mcpot' - most often the returned object of the fitmcgpd function.
n.sim	The number of simulation of Markov chains.
n.mc	The length of the simulated Markov chains.
plot	Logical. If TRUE (default), the density of the extremal index is plotted.
...	Optional parameters to be passed to the plot function.

Details

The Markov chains are simulated using the simmc function to obtained dependent realisations u_i of standard uniform realizations. Then, they are transformed to correspond to the parameter of the fitted markov chain model. Thus, if u, σ, ξ is the location, scale and shape parameters ; and λ is the probability of exceedance of u, then by defining :

$$\sigma_* = \xi \times \frac{u}{\lambda^{-\xi} - 1}$$

the realizations $y_i = qgpd(u_i, 0, \sigma_*, \xi)$ are distributed such as the probability of exceedance of u is equal to λ.

At last, the extremal index for each generated Markov chain is estimated using the estimator of Ferro and Segers (2003) (and thus avoid any declusterization).

Value

The function returns a optionally plot of the kernel density estimate of the extremal index. In addition, the vector of extremal index estimations is returned invisibly.

Author(s)

Mathieu Ribatet

References

Fawcett L., and Walshaw D. (2006) Markov chain models for extreme wind speed. Environmetrics, **17**:(8) 795–809.

Ferro, C. and Segers, J. (2003) Inference for clusters of extreme values. Journal of the Royal Statistical Society. Series B **65**:(2) 545–556.

Ledford A., and Tawn, J. (1996) Statistics for near Independence in Multivariate Extreme Values. Biometrika, **83** 169–187.

Smith, R., and Tawn, J., and Coles, S. (1997) Markov chain models for threshold exceedances. Biometrika, **84** 249–268.

See Also

simmc, fitmcgpd, fitexi

Examples

```
mc <- simmc(100, alpha = 0.25)
mc <- qgpd(mc, 0, 1, 0.25)
log <- fitmcgpd(mc, 2, shape = 0.25, scale = 1)
dexi(log, n.sim = 100)
```

diplot *Threshold Selection: The Dispersion Index Plot*

Description

The Dispersion Index Plot

Usage

```
diplot(data, u.range, main, xlab, ylab, nt = max(200, nrow(data)),
conf=0.95, ...)
```

Arguments

data	A matrix with two column. The first one represents the date of events (in a numeric format) and the second the data associated with those dates.
u.range	A numeric vector of length two giving the limit of threshold analyzed. If missing, default values are taken.
main	The title of the plot.
xlab,ylab	Labels for the x and y axis.
nt	The number of thresholds at which the dispersion index plot is evaluated.
conf	The confident coefficient for the plotted confidence intervals.
...	Other arguments to be passed to the plot function.

Details

According to the Extreme Value Theory, the number of exceedance over a high threshold in a fixed period - generally a year - must be distributed as Poisson process. As for a random variable Poisson distributed, the ratio of the variance and the mean is equal to 1, one can test if the ratio $\mathtt{DI} = var/mean$ differs from 1. Moreover, confidence levels for DI can be calculated by testing against a χ^2 distribution with M-1 degree of freedom, M being the total number of fixed periods - generally the total number of years in the sample. So, the Poisson hypothesis is not rejected if the estimated DI is within the range

$$\left[\frac{\chi^2_{\alpha/2,\mathtt{M}-1}}{\mathtt{M}-1}, \frac{\chi^2_{1-\alpha/2,\mathtt{M}-1}}{\mathtt{M}-1}\right]$$

Value

It returns invisibly a list with two components. The first one 'thresh' gives the thresholds analyzed. The second 'DI' gives the dispersion index relative to the threshold.

Author(s)

Mathieu Ribatet

References

Cunnane, C. (1979) Note on the poisson assumption in partial duration series model. Water Resource Research, 15**(2)** :489–494.

Examples

```
data(ardieres)
ardieres <- clust(ardieres, 4, 10 / 365, clust.max = TRUE)
diplot(ardieres)
```

Fisher Confidence Interval

Fisher Based Confidence Interval for the GP Distribution

Description

Compute Fisher based confidence intervals on parameter and return level for the GP distribution. This is achieved through asymptotic theory and the Observed information matrix of Fisher and eventually the Delta method.

Usage

```
gpd.fishape(fitted, conf = 0.95)
gpd.fiscale(fitted, conf = 0.95)
gpd.firl(fitted, prob, conf = 0.95)
```

Arguments

`fitted`	R object given by function `fitgpd`.
`prob`	The probability of non exceedance.
`conf`	Numeric. The confidence level.

Value

Returns a vector of the lower and upper bound for the confidence interval.

Author(s)

Mathieu Ribatet

See Also

`rp2prob`, `prob2rp`, `gpd.pfscale`, `gpd.pfshape`, `gpd.pfrl` and `confint`

Examples

```
data(ardieres)
ardieres <- clust(ardieres, 4, 10 / 365, clust.max = TRUE)
fitted <- fitgpd(ardieres[,"obs"], 5, 'mle')
gpd.fishape(fitted)
gpd.fiscale(fitted)
```

`Fit the GP Distribution`

Fitting a GPD to Peaks Over a Threshold

Description

Maximum (Penalized) Likelihood, Unbiased Probability Weighted Moments,Biased Probability Weighted Moments, Moments, Pickands', Minimum Density Power Divergence, Medians, Likelihood Moment and Maximum Goodness-of-Fit Estimators to fit Peaks Over a Threshold to a GP distribution.

Usage

```
fitgpd(data, threshold, est = "mle", ...)
```

Arguments

`data`	A numeric vector.
`threshold`	A numeric value giving the threshold for the GPD. The `'mle'` estimator allows varying threshold; so that threshold could be for this case a numeric vector. Be careful, varying thresholds are used cyclically if length doesn't match with `data`.
`est`	A string giving the names of the estimator. It can be `'mle'` (the default), `'mple'`, `'moments'`, `'pwmu'`, `'pwmb'`, `'mdpd'`, `'med'`, `'pickands'`, `'lme'` and `'mgf'` for the maximum likelihood, maximum penalized likelihood, moments, unbiased probability weighted moments, biased probability weigthed moments, minimum density power divergence, medians, Pickands', likelihood moment and maximum goodness-of-fit estimators respectively.
`...`	Other optional arguments to be passed to the `optim` function, allow hand fixed parameters (only for the `mle`, `mple` and `mgf` estimators) or passed several options to specific estimators - see the Note section.

Value

This function returns an object of class `"uvpot"` with components:

`fitted.values`	A vector containing the estimated parameters.
`std.err`	A vector containing the standard errors.
`fixed`	A vector containing the parameters of the model that have been held fixed.
`param`	A vector containing all parameters (optimized and fixed).

`deviance`	The deviance at the maximum likelihood estimates.
`corr`	The correlation matrix.
`convergence, counts, message`	Components taken from the list returned by `optim` - for the `mle` method.
`threshold`	The threshold passed to argument `threshold`.
`nat, pat`	The number and proportion of exceedances.
`data`	The data passed to the argument `data`.
`exceed`	The exceedances, or the maxima of the clusters of exceedances.
`scale`	The scale parameter for the fitted generalized Pareto distribution.
`std.err.type`	The standard error type - for `'mle'` only. That is `Observed` or `Expected` Information matrix of Fisher.
`var.thresh`	Logical. Specify if the threshold is a varying one - `'mle'` only. For other methods, threshold is always constant i.e. `var.thresh = FALSE`.

Note

The Maximum Likelihood estimator is obtained through a slightly modified version of Alec Stephenson's `fpot.norm` function in the `evd` package.

For the `'mple'` estimator, the likelihood function is penalized using the following function :

$$P(\xi) = \begin{cases} 1, & \xi \leq 0 \\ \exp\left[-\lambda\left(\frac{1}{1-\xi} - 1\right)^{\alpha}\right], & 0 < \xi < 1 \\ 0, & \xi \geq 1 \end{cases}$$

where α and λ are the penalizing constants. Coles and Dixon (1999) suggest the use of $\alpha = \lambda = 1$.

The `'lme'` estimator has a special parameter `'r'`. Zhang (2007) shows that a value of -0.5 should be accurate in most of the cases. However, other values such as `r < 0.5` can be explored. In particular, if `r` is approximatively equal to the opposite of the true shape parameter value, then the `lme` estimate is equivalent to the `mle` estimate.

The `'pwmb'` estimator has special parameters `'a'` and `'b'`. These parameters are called the "plotting-position" values. Hosking and Wallis (1987) recommend the use of `a = 0.35` and `b = 0` (the default). However, different values can be tested.

For the `'pwmu'` and `'pwmb'` approaches, one can pass the option `'hybrid = TRUE'` to use hybrid estimators as proposed by Dupuis and Tsao (1998). Hybrid estimators avoid to have no feasible points.

The `mdpd` estimator has a special parameter `'a'`. This is a parameter of the "density power divergence". Juarez and Schucany (2004) recommend the use of `a = 0.1`, but any value of `a` such as `a > 0` can be used (small values are recommend yet).

The `med` estimator admits two extra arguments `tol` and `maxit` to control the "stopping-rule" of the optimization process.

The `mgf` approach uses goodness-of-fit statistics to estimate the GPD parameters. There are currently 8 different statitics: the Kolmogorov-Smirnov `"KS"`, Cramer von Mises `"CM"`, Anderson Darling `"AD"`, right tail Anderson Darling `"ADR"`, left tail Anderson Darling `"ADL"`, right tail Anderson Darling (second degree) `"AD2R"`, left tail Anderson Darling (second degree) `"AD2L"` and the Anderson Darling (second degree) `"AD2"` statistics.

Author(s)

Mathieu Ribatet

References

Coles, S. (2001) *An Introduction to Statistical Modelling of Extreme Values*. Springer Series in Statistics. London.

Coles, S. and Dixon, M. (1999) Likelihood-Based Inference for Extreme Value Models. Extremes **2**(1):5–23.

Dupuis, D. and Tsao (1998) M. A hybrid estimator for generalized Pareto and extreme-value distributions. Communications in Statistics-Theory and Methods **27**:925–941.

Hosking, J. and Wallis, J. (1987) Parameters and Quantile Estimation for the Generalized Pareto Distribution. Technometrics **29**:339–349.

Juarez, S. and Schucany, W. (2004) Robust and Efficient Estimation for the Generalized Pareto Distribution. Extremes **7**:237–251.

Luceno, A. (2006) Fitting the generalized Pareto distribution to data using maximum goodness-of-fit estimators. Computational Statistics and Data Analysis **51**:904–917.

Peng, L. and Welsh, A. (2001) Robust Estimation of the Generalized Pareto Distribution. Extremes **4**:53–65.

Embrechts, P and Kluppelberg, C. and Mikosch, T (1997) *Modelling Extremal Events for Insurance and Finance.* Springers.

Pickands, J. (1975) Statistical Inference Using Extreme Order Statistics. Annals of Statistics. **3**:119–131.

Zhang, J. (2007) Likelihood Moment Estimation for the Generalized Pareto Distribution. Australian and New Zealand Journal of Statistics. **49**(1):69–77.

See Also

The following usual generic functions are available `print`, `plot`, `confint` and `anova` as well as new generic functions `retlev`, `qq`, `pp`, `dens` and `convassess`.

Examples

```
x <- rgpd(200, 1, 2, 0.25)
mle <- fitgpd(x, 1, "mle")$param
pwmu <- fitgpd(x, 1, "pwmu")$param
pwmb <- fitgpd(x, 1, "pwmb")$param
pickands <- fitgpd(x, 1, "pickands")$param    ##Check if Pickands estimates
                                              ##are valid or not !!!
med <- fitgpd(x, 1, "med",                    ##Sometimes the fitting algo is not
start = list(scale = 2, shape = 0.25))$param  ##accurate. So specify
                                              ##good starting values is
                                              ##a good idea.
mdpd <- fitgpd(x, 1, "mdpd")$param
lme <- fitgpd(x, 1, "lme")$param
mple <- fitgpd(x, 1, "mple")$param
ad2r <- fitgpd(x, 1, "mgf", stat = "AD2R")$param
```

```
print(rbind(mle, pwmu, pwmb, pickands, med, mdpd, lme,
 mple, ad2r))

##Use PWM hybrid estimators
fitgpd(x, 1, "pwmu", hybrid = FALSE)

##Now fix one of the GPD parameters
##Only the MLE, MPLE and MGF estimators are allowed !
fitgpd(x, 1, "mle", scale = 2)
fitgpd(x, 1, "mple", shape = 0.25)
```

`fitbvgpd` *Fitting Bivariate Peaks Over a Threshold Using Bivariate Extreme Value Distributions*

Description

Fitting a bivariate extreme value distribution to bivariate exceedances over thresholds using censored maximum likelihood procedure.

Usage

```
fitbvgpd(data, threshold, model = "log", start, ..., cscale = FALSE,
cshape = FALSE, std.err.type = "observed", corr = FALSE, warn.inf = TRUE,
method = "BFGS")
```

Arguments

`data`	A matrix with two columns which gives the observation vector for margin 1 and 2 respectively. `NA` values are considered to fall below the threshold.
`threshold`	A numeric vector for the threshold (of length 2).
`model`	A character string which specifies the model used. Must be one of `log` (the default), `alog`, `nlog`, `anlog`, `mix` and `amix` for the logistic, asymmetric logistic, negative logistic, asymmetric negative logistic, mixed and asymmetric mixed models.
`start`	Optional. A list for starting values in the fitting procedure.
`...`	Additional parameters to be passed to the `optim` function or to the bivariate model. In particular, parameter of the model can be hand fixed.
`cscale`	Logical. Should the two scale parameters be equal?
`cshape`	Logical. Should the two shape parameters be equal?
`std.err.type`	The type of the standard error. Currently, one must specify `"observed"` for observed Fisher information matrix or `"none"` for no computations of the standard errors.
`corr`	Logical. Should the correlation matrix be computed?
`warn.inf`	Logical. Should users be warned if likelihood is not finite at starting values?
`method`	The optimization method, see `optim`.

Details

The bivariate exceedances are fitted using censored likelihood procedure. This methodology is fully described in Ledford (1996).

Most of models are described in Kluppelberg (2006).

Value

The function returns an object of class `c("bvpot","pot")`. As usual, one can extract several features using `fitted` (or `fitted.values`), `deviance`, `logLik` and `AIC` functions.

`fitted.values`	The maximum likelihood estimates of the bivariate extreme value distribution.
`std.err`	A vector containing the standard errors - only present when the observed information matrix is not singular.
`var.cov`	The asymptotic variance covariance matrix - only presents when the observed information matrix is not singular.
`deviance`	The deviance.
`corr`	The correlation matrix.
`convergence, counts, message`	Informations taken from the `optim` function.
`threshold`	The marginal thresholds.
`pat`	The marginal proportion above the threshold.
`nat`	The marginal number above the threshold.
`data`	The bivariate matrix of observations.
`exceed1, exceed2`	The marginal exceedances.
`call`	The call of the current function.
`model`	The model for the bivariate extreme value distribution.
`chi`	The chi statistic of Coles (1999). A value near 1 (resp. 0) indicates perfect dependence (resp. independence).

Warnings

Because of numerical problems, their exists artificial numerical constraints imposed on each model. These are:

- For the logistic and asymmetric logistic models: α must lie in [0.05, 1] instead of [0,1];
- For the negative logistic model: α must lie in [0.01, 15] instead of $[0, \infty[$;
- For the asymmetric negative logistic model: α must lie in [0.2, 15] instead of $[0, \infty[$;
- For the mixed and asymmetric mixed models: None artificial numerical constraints are imposed.

For this purpose, users must check if estimates are near these artificial numerical constraints. Such cases may lead to substantial biases on the GP parameter estimates. One way to detect quickly if estimates are near the border constraints is to look at the standard errors for the dependence parameters. Small values (i.e. < 1e-5) often indicates that numerical constraints have been reached.

In addition, users must be aware that the mixed and asymmetric mixed models can not deal with perfect dependence.

Thus, user may want to plot the Pickands' dependence function to see if variable are near independence or dependence cases using the pickdep function.

Author(s)

Mathieu Ribatet

References

Coles, S., Heffernan, J. and Tawn, J. (1999) Dependence Measure for Extreme Value Analyses. Extremes, **2**:4 339–365.

Kl\"uppelberg, C., and May A. (2006) Bivariate extreme value distributions based on polynomial dependence functions. Mathematical Methods in the Applied Sciences, **29**: 1467–1480.

Ledford A., and Tawn, J. (1996) Statistics for near Independence in Multivariate Extreme Values. Biometrika, **83**: 169–187.

See Also

The following usual generic functions are available print, plot and anova as well as new generic functions retlev and convassess.

See also pickdep, specdens.

For optimization in R, see optim.

Examples

```
x <- rgpd(1000, 0, 1, 0.25)
y <- rgpd(1000, 3, 1, -0.25)
ind <- fitbvgpd(cbind(x, y), c(0, 3), "log")
ind
ind2 <- fitbvgpd(cbind(x, y), c(0, 3), "log", alpha = 1)
ind2
ind3 <- fitbvgpd(cbind(x, y), c(0, 3), cscale = TRUE)
ind3
##The mixed model can not deal with perfect dependent variables
##Thus, there is a substantial bias in GPD parameter estimates
dep <- fitbvgpd(cbind(x, x + 3), c(0, 3), "mix")
dep
```

---

fitexi *Extremal Index Estimation*

Description

Estimation of the extremal index using interexceedance times.

Usage

```
fitexi(data, threshold)
```

Arguments

`data`	A matrix with two columns: `obs` for the observations and `time` for the time.
`threshold`	The threshold.

Details

The extremal index estimator proposed by Ferro and Segers (2003) is based on interexceedance times. In particular, it does not require a specific declusterization of the time series.

The `tim.cond` gives an "automatic" procedure to decluster the time series without any subjective choice to define the independence condition between clusters.

Value

This function returns a list with two components. The first one `exi` gives the estimate of the extremal index; while the second, `tim.cond` gives the time condition for independence between events to be passed to the `clust` function.

Author(s)

Mathieu Ribatet

References

Ferro, C. and Segers, J. (2003) Inference for clusters of extreme values. Journal of the Royal Statistical Society. Series B **65:2** 545–556.

See Also

`clust`

Examples

```
n.obs <- 500
x <- rexp(n.obs + 1)
y <- pmax(x[-1], x[-(n.obs + 1)])## The extremal index is 0.5

u <- quantile(y, 0.95)
fitexi(y, u)
```

`fitmcgpd` *Fitting Markov Chain Models to Peaks Over a Threshold*

Description

Fitting a Markov chain to cluster exceedances using a bivariate extreme value distribution and a censored maximum likelihood procedure.

Usage

```
fitmcgpd(data, threshold, model = "log", start, ..., std.err.type =
"observed", corr = FALSE, warn.inf = TRUE, method = "BFGS")
```

Arguments

`data`	A vector of observations.
`threshold`	The threshold value.
`model`	A character string which specifies the model used. Must be one of `log` (the default), `alog`, `nlog`, `anlog`, `mix` and `amix` for the logistic, asymmetric logistic, negative logistic, asymmetric negative logistic, mixed and asymmetric mixed models.
`start`	Optional. A list for starting values in the fitting procedure.
`...`	Additional parameters to be passed to the `optim` function or to the bivariate model. In particular, parameter of the model can be hand fixed.
`std.err.type`	The type of the standard error. Currently, one must specify ``observed'' for observed Fisher information matrix or ``none'' for no computations of the standard errors.
`corr`	Logical. Should the correlation matrix be computed?
`warn.inf`	Logical. Should users be warned if likelihood is not finite at starting values?
`method`	The optimization method, see `optim`.

Details

The Markov Chain model is defined as follows:

$$L\left(y;\theta_1,\theta_2\right) = f\left(x_1;\theta_1\right)\prod_{i=2}^{n} f\left(y_i|y_{i-1};\theta_1,\theta_2\right)$$

As exceedances above a (high enough) threshold are of interest, it is assumed that the marginal are GPD distributed, while the joint distribution is represented by a bivariate extreme value distribution. Smith et al. (1997) present theoretical results about this Markov Chain model.

The bivariate exceedances are fitted using censored likelihood procedure. This methodology is fully described in Ledford (1996).

Most of models are described in Kluppelberg (2006).

Value

The function returns an object of class `c("mcpot", "uvpot", "pot")`. As usual, one can extract several features using `fitted` (or `fitted.values`), `deviance`, `logLik` and `AIC` functions.

`fitted.values`	The maximum likelihood estimates of the Markov chain including estimated parameters of the bivariate extreme value distribution.
`std.err`	A vector containing the standard errors - only present when the observed information matrix is not singular.
`var.cov`	The asymptotic variance covariance matrix - only presents when the observed information matrix is not singular.
`deviance`	The deviance.
`corr`	The correlation matrix.
`convergence, counts, message`	Informations taken from the `optim` function.
`threshold`	The threshold.
`pat`	The proportion above the threshold.
`nat`	The number above the threshold.
`data`	The observations.
`exceed`	The exceedances.
`call`	The call of the current function.
`model`	The model for the bivariate extreme value distribution.
`chi`	The chi statistic of Coles (1999). A value near 1 (resp. 0) indicates perfect dependence (resp. independence).

Warnings

Because of numerical problems, there exists artificial numerical constraints imposed on each model. These are:

- For the logistic and asymmetric logistic models: α must lie in [0.05, 1] instead of [0,1];
- For the negative logistic model: α must lie in [0.01, 15] instead of $[0, \infty[$;
- For the asymmetric negative logistic model: α must lie in [0.2, 15] instead of $[0, \infty[$;
- For the mixed and asymmetric mixed models: None artificial numerical constraints are imposed.

For this purpose, users must check if estimates are near these artificial numerical constraints. Such cases may lead to substantial biases on the GP parameter estimates. One way to detect quickly if estimates are near the border constraints is to look at the standard errors for the dependence parameters. Small values (i.e. < 1e-5) often indicates that numerical constraints have been reached.

In addition, users must be aware that the mixed and asymmetric mixed models can not deal with perfect dependence.

Thus, user may want to plot the Pickands' dependence function to see if variable are near independence or dependence cases using the `pickdep` function.

In addition, we recommend to fix the marginal parameters. Indeed, even this is a two steps optimization procedure, this avoid numerical troubles - the likelihood function for the Markov chain model seems to be problematic. Thus, estimates are often better using the two stages approach.

Author(s)

Mathieu Ribatet

References

Kl\"uppelberg, C., and May A. (2006) Bivariate extreme value distributions based on polynomial dependence functions. Mathematical Methods in the Applied Sciences, **29** 1467–1480.

Ledford A., and Tawn, J. (1996) Statistics for near Independence in Multivariate Extreme Values. Biometrika, **83** 169–187.

Smith, R., and Tawn, J., and Coles, S. (1997) Markov chain models for threshold exceedances. Biometrika, **84** 249–268

See Also

The following usual generic functions are available `print`, `plot` as well as new generic functions `retlev` and `convassess`.

See also `pickdep`.

For optimization in R, see `optim`.

Examples

```
mc <- simmc(1000, alpha = 0.25)
mc <- qgpd(mc, 0, 1, 0.25)
##A first application when marginal parameter are estimated
fitmcgpd(mc, 0)
##Another one where marginal parameters are fixed
mle <- fitgpd(mc, 0)
fitmcgpd(mc, 0, scale = mle$param["scale"], shape = mle$param["shape"])
```

---

fitpp *Fitting the point process characterisation to exceedances above a threshold*

Description

This function estimates the point process characterisation from exceedances above a threshold.

Usage

```
fitpp(data, threshold, noy = length(data) / 365.25, start, ...,
std.err.type = "observed", corr = FALSE, method = "BFGS", warn.inf = TRUE)
```

Arguments

data	A numeric vector.
threshold	A numeric value giving the threshold for the GPD.
noy	Numeric. The number of year of observation.
start	A named list that gives the starting values for the optimization routine. Each list argument must correspond to one parameter to be estimated. May be missing.
...	Other optional arguments to be passed to the optim function, allow hand fixed parameters (only - see the Note section.
std.err.type	A character string. If "observed", the standard errors are derived from the observed Fisher information matrix. If "none", standard errors are not computed.
corr	Logical. Does the asymptotic correlation matrix has to be computed? Default is "not computed" - e.g. FALSE.
method	A character string specifying which numerical optimization procedure has to be used. See optim for more details.
warn.inf	Logical. If TRUE (default), users will be warned if the log-likelihood is not finite at starting values - as it may cause some problem during the optimation stage.

Value

This function returns a list with components:

fitted.values	A vector containing the estimated parameters.
std.err	A vector containing the standard errors.
fixed	A vector containing the parameters of the model that have been held fixed.
param	A vector containing all parameters (optimized and fixed).
deviance	The deviance at the maximum likelihood estimates.
corr	The correlation matrix.
convergence, counts, message	Components taken from the list returned by optim - for the mle method.
threshold	The threshold passed to argument threshold.
nat, pat	The number and proportion of exceedances.
data	The data passed to the argument data.
exceed	The exceedances, or the maxima of the clusters of exceedances.
scale	The scale parameter for the fitted generalized Pareto distribution.
std.err.type	The standard error type - for 'mle' only. That is Observed Information matrix of Fisher.
var.thresh	Logical. Specify if the threshold is a varying one - 'mle' only. For other methods, threshold is always constant i.e. var.thresh = FALSE. Not implemented yet.

Author(s)

Mathieu Ribatet

References

Coles, S. (2001) *An Introduction to Statistical Modelling of Extreme Values*. Springer Series in Statistics. London.

Embrechts, P and Kluppelberg, C. and Mikosch, T (1997) *Modelling Extremal Events for Insurance and Finance.* Springers.

Pickands, J. (1975) Statistical Inference Using Extreme Order Statistics. Annals of Statistics. **3**:119–131.

Examples

```
x <- rgpd(1000, 0, 1, 0.2)
fitpp(x, 0)
```

`Flood Flows` *High Flood Flows of the Ardieres River at Beaujeu*

Description

A data frame containing flood discharges, in units of cubic meters per second, of the Ardieres River at Beaujeu (FRANCE), over a period of 33 years and the related date of those events.

Usage

```
data(ardieres)
```

Format

A data frame with two columns: "time" and "obs".

Author(s)

Mathieu Ribatet

Examples

```
data(ardieres)
plot(ardieres, xlab = "Time (Years)", ylab = expression(paste("Flood
discharges ", m^2/s, sep="")), type = "l")
```

`Generalized Pareto` *The Generalized Pareto Distribution*

Description

Density, distribution function, quantile function and random generation for the GP distribution with location equal to 'loc', scale equal to 'scale' and shape equal to 'shape'.

Usage

```
rgpd(n, loc = 0, scale = 1, shape = 0)
pgpd(q, loc = 0, scale = 1, shape = 0, lower.tail = TRUE, lambda = 0)
qgpd(p, loc = 0, scale = 1, shape = 0, lower.tail = TRUE, lambda = 0)
dgpd(x, loc = 0, scale = 1, shape = 0, log = FALSE)
```

Arguments

`x, q`	vector of quantiles.
`p`	vector of probabilities.
`n`	number of observations.
`loc`	vector of the location parameters.
`scale`	vector of the scale parameters.
`shape`	a numeric of the shape parameter.
`lower.tail`	logical; if TRUE (default), probabilities are $\Pr[X \leq x]$, otherwise, $\Pr[X > x]$.
`log`	logical; if TRUE, probabilities p are given as log(p).
`lambda`	a single probability - see the "value" section.

Value

If 'loc', 'scale' and 'shape' are not specified they assume the default values of '0', '1' and '0', respectively.

The GP distribution function for loc = u, scale = σ and shape = ξ is

$$G(x) = 1 - \left[1 + \frac{\xi(x-u)}{\sigma}\right]^{-1/\xi}$$

for $1 + \xi(x-u)/\sigma > 0$ and $x > u$, where $\sigma > 0$. If $\xi = 0$, the distribution is defined by continuity corresponding to the exponential distribution.

By definition, the GP distribution models exceedances above a threshold. In particular, the G function is a suited candidate to model

$$\Pr\left[X \geq x | X > u\right] = 1 - G(x)$$

for u large enough.

However, it may be usefull to model the "non conditional" quantiles, that is the ones related to $\Pr[X \leq x]$. Using the conditional probability definition, one have :

$$\Pr[X \geq x] = (1-\lambda)\left(1+\xi\frac{x-u}{\sigma}\right)^{-1/\xi}$$

where $\lambda = \Pr[X \leq u]$.

When $\lambda = 0$, the "conditional" distribution is equivalent to the "non conditional" distribution.

Examples

```
dgpd(0.1)
rgpd(100, 1, 2, 0.2)
qgpd(seq(0.1, 0.9, 0.1), 1, 0.5, -0.2)
pgpd(12.6, 2, 0.5, 0.1)
##for non conditional quantiles
qgpd(seq(0.9, 0.99, 0.01), 1, 0.5, -0.2, lambda = 0.9)
pgpd(2.6, 2, 2.5, 0.25, lambda = 0.5)
```

`gpd2frech` *Transforms GPD Observations to Unit Frechet Ones and Vice Versa*

Description

Transforms GPD observations to unit Frechet ones and vice versa

Usage

```
gpd2frech(x, loc = 0, scale = 1, shape = 0, pat = 1)
frech2gpd(z, loc = 0, scale = 1, shape = 0, pat = 1)
```

Arguments

`x,z`	The vector of observations.
`loc,scale,shape`	The location, scale and shape parameters respectively.
`pat`	The proportion above the threshold, i.e. Pr[X > log] = pat.

Details

Let x_i, $i = 1, \ldots, n$ be the realisation of a GPD random variable. Thus, the transformation to unit Frechet is defined as:

$$z_i = -\frac{1}{\log\left[1 - pat\left(1 + shape\frac{x_i - loc}{scale}\right)_+^{-1/shape}\right]}$$

Value

A numeric vector.

Author(s)

Mathieu Ribatet

Examples

```
x <- rgpd(10, 0, 1, 0.25)
z <- gpd2frech(x, 0, 1, 0.25)
z
all(frech2gpd(z, 0, 1, 0.25) == x)
```

L-moments *Compute Sample L-moments*

Description

Compute the sample L-moments - unbiased version.

Usage

```
samlmu(x, nmom = 4, sort.data = TRUE)
```

Arguments

x	a vector of data
nmom	a numeric value giving the number of sample L-moments to be computed
sort.data	a logical which specifies if the vector of data x should be sorted or not.

Value

This function returns a vector of length nmom corresponding to the sample L-moments. Note that for orders greater or equal than 3 it is the L-moments ratio that is sample L-coefficient of variation, sample L-skewness, sample L-kurtosis, ...

References

Hosking, J. R. M. (1990) L-moment analysis and estimation of order statistics. Journal of the Royal Statistical Society Series B, **52**: 105–124.

Examples

```
x <- runif(50)
samlmu(x, nmom = 5)
```

`lmomplot` *Threshold Selection: The L-moments Plot*

Description

Plots of sample L-Skewness ans L-Kurtosis estimates at various thresholds for peaks over threshold modelling, using the Generalized Pareto parametrization.

Usage

```
lmomplot(data, u.range, nt = max(50, length(data)), identify = TRUE,
...)
```

Arguments

`data`	A numeric vector.
`u.range`	A numeric vector of length two, giving the limits for the thresholds at which the model is fitted.
`nt`	The number of thresholds at which the sample L-moments are evaluated.
`identify`	Logical. If `TRUE`, points on the plot are identify using `identify` function.
`...`	Other arguments to be passed to the model fit function `fitgpd`.

Details

For each thresholds, sample L-skewness and L-kurtosis are computed. If data are GP distributed, one have :

$$\tau_4 = \frac{\tau_3 \left(1 + 5\tau_3\right)}{5 + \tau_3}$$

So, a threshold is acceptable if sample (τ_3, τ_4) are near the theoretical curve.

Warnings

L-moments plot are really difficult to interpret. It can help us to say if the GP distribution is suited to model data.

Author(s)

Mathieu Ribatet

References

Hosking, J. R. M. and Wallis, J. R. (1997) *Regional Frequency Analysis*. Cambridge University Press.

Begueria, S. (2005) Uncertainties in partial duration series modelling of extremes related to the choice of the threshold value. Journal of Hydrology, 303**(1-4)**: 215–230.

See Also

fitgpd, mrlplot, tcplot

Examples

```
data(ardieres)
ardieres <- clust(ardieres, 4, 10 / 365, clust.max = TRUE)
flows <- ardieres[, "obs"]
lmomplot(flows, identify = FALSE)
```

---

logLik.pot *Extract Log-Likelihood*

Description

Extract Log-Likelihood for object of class 'pot'

Usage

```
## S3 method for class 'pot'
logLik(object, ...)
```

Arguments

object	An object of class 'pot'. Most often, this is an object return by the fitgpd, fitbvgpd and fitmcgpd functions.
...	Other arguments to be passed to the logLik function.

Value

Standard logLik object: see logLik.

Author(s)

Mathieu Ribatet

See Also

logLik

Examples

```
x <- rgpd(500, 0, 1, -0.15)
mle <- fitgpd(x, 0)
logLik(mle)
```

`mrlplot` *Threshold Selection: The Empirical Mean Residual Life Plot*

Description

The empirical mean residual life plot.

Usage

```
mrlplot(data, u.range, main, xlab, ylab, nt = max(100, length(data)),
lty = rep(1,3), col = c('grey', 'black', 'grey'), conf = 0.95, lwd = c(1,
1.5, 1), ...)
```

Arguments

`data`	A numeric vector.
`u.range`	A numeric vector of length two, giving the limits for the thresholds at which the mean residual life plot is evaluated. If `u.range` is not given, sensible defaults are used.
`main`	Plot title.
`xlab, ylab`	x and y axis labels.
`nt`	The number of thresholds at which the mean residual life plot is evaluated.
`lty, col, lwd`	Arguments passed to `matplot`. The first and last elements of `lty` correspond to the lower and upper confidence limits respectively. Use zero to supress.
`conf`	The (pointwise) confidence coefficient for the plotted confidence intervals.
`...`	Other arguments to be passed to `matplot`.

Details

The empirical mean residual life plot is the locus of points

$$\left(u, \frac{1}{n_u}\sum_{i=1}^{n_u}(x_{(i)} - u)\right)$$

where $x_{(1)}, \ldots, x_{(n_u)}$ are the n_u observations that exceed the threshold u. If the exceedances of a threshold u_0 are generalized Pareto, the empirical mean residual life plot should be approximately linear for $u > u_0$.

The confidence intervals within the plot are symmetric intervals based on the approximate normality of sample means.

Value

A list with components `x` and `y` is invisibly returned. The components contain those objects that were passed to the formal arguments `x` and `y` of `matplot` in order to create the mean residual life plot.

Author(s)

Stuart Coles and Alec Stephenson

References

Coles, S. (2001) *An Introduction to Statistical Modelling of Extreme Values*. Springer Series in Statistics. London.

Embrechts, P., Kl\"uppelberg, C., and Mikosch, T. (1997) *Modelling Extremal Events for Insurance and Finance.*

See Also

fitgpd, matplot, tcplot

Examples

```
data(ardieres)
ardieres <- clust(ardieres, 4, 10 / 365, clust.max = TRUE)
flows <- ardieres[, "obs"]
mrlplot(flows)
```

---

pickdep *The Pickands' Dependence Function*

Description

Return and optionally plot the Pickands' dependence function.

Usage

```
pickdep(fitted, main, bound = TRUE, plot = TRUE, ...)
```

Arguments

fitted	A object of class bvpot. Usually, fitted is the return of function fitbvgpd.
main	May be missing. If present, the plot title.
bound	Logical. Should the perfect dependent and independent case bounds be plotted?
plot	Logical. Should the dependence function be plotted?
...	Optional parameters to be passed to the lines function.

Details

It is common to parametrize a bivariate extreme value distribution according to the Pickands' representation (Pickands, 1981). That is, if G is any bivariate extreme value distribution, then it has the following parametrization:

$$G\left(y_1, y_2\right) = \exp\left[-\left(\frac{1}{z_1} + \frac{1}{z_2}\right) A\left(\frac{z_2}{z_1 + z_2}\right)\right]$$

where z_i are unit Frechet.

A is the Pickands' dependence function. It has the following properties:

- A is defined on [0,1];
- $A(0) = A(1) = 1$;
- $\max\left(w, 1-w\right) \leq A(w) \leq 1, \quad \forall w$;
- A is a convex function;
- For two independent (unit Frechet) random variables, $A(w) = 1, \quad \forall w$;
- For two perfectly dependent (unit Frechet) random variables, $A(w) = \max(w, 1-w)$.

Value

The function returns an invisible function: the Pickands' dependence function. Moreover, the returned object has an attribute which specifies the model for the bivariate extreme value distribution.

If `plot = TRUE`, then the dependence function is plotted.

Author(s)

Mathieu Ribatet

References

Pickands, J. (1981) Multivariate Extreme Value Distributions *Proceedings 43rd Session International Statistical Institute*

Examples

```
x <- rbvgpd(1000, alpha = 0.9, model = "mix", mar1 = c(0,1,0.25),
 mar2 = c(2,0.5,0.1))
Mmix <- fitbvgpd(x, c(0,2), "mix")
pickdep(Mmix)
```

plot.bvpot *Graphical Diagnostics: the Bivariate Extreme Value Distribution Model.*

Description

Plot several graphics to judge goodness of fit of the fitted model.

Usage

```
## S3 method for class 'bvpot'
plot(x, mains, which = 1:3, ask = nb.fig < length(which)
&& dev.interactive(), ...)
```

Arguments

x	An object of class "bvpot". Most often, the object returned by the fitbvgpd function.
mains	May be missing. If present a 3–vector of character strings which gives the titles of the plots.
which	a numeric vector which specifies which plot must be drawn : '1' for Pickands' Dependence Function plot, '2' for a bivariate return level plot, '3' for the spectral density plot.
ask	Logical. If TRUE, user is asked before each plot.
...	Other parameters to pass to the plot function.

Value

Several plots.

Author(s)

Mathieu Ribatet

See Also

fitbvgpd

Examples

```
x <- rbvgpd(1000, alpha = 0.55, mar1 = c(0,1,0.25), mar2 = c(2,0.5,0.1))
Mlog <- fitbvgpd(x, c(0, 2), "log")
layout(matrix(c(1,1,2,2,0,3,3,0), 2, byrow = TRUE))
plot(Mlog)
```

plot.mcpot *Graphical Diagnostics: Markov Chains for All Exceedances.*

Description

Plot several graphics to judge goodness of fit of the fitted model.

Usage

```
## S3 method for class 'mcpot'
plot(x, opy, exi, mains, which = 1:4, ask = nb.fig <
length(which) && dev.interactive(), acf.type = "partial", ...)
```

Arguments

x	An object of class "bvpot". Most often, the object returned by the fitbvgpd function.
opy	Numeric. The number of **O**bservation **P**er **Y**ear (or more generally per block). If missing, the function warns and set it to 365.
exi	Numeric. The extremal index value. If missing, the estimator of Ferro and Segers (2003) is used.
mains	May be missing. If present a 4–vector of character strings which gives the titles of the plots.
which	a numeric vector which specifies which plot must be drawn: '1' for the auto correlation plot, '2' for Pickands' Dependence Function plot, '3' for the spectral density plot and '4' for a bivariate return level plot.
ask	Logical. If TRUE, user is asked before each plot.
acf.type	The type of auto correlation to be plotted. Must be one of "correlation", "covariance" or "partial" (the default). See the acf function.
...	Other parameters to pass to the plot function.

Value

Several plots and returns invisibly the return level function.

Warning

See the warning for the return level estimation in documentation of the retlev.mcpot function.

Note

For the return level plot, the observations are not plotted as these are dependent realisations. In particular, the return periods computed using the prob2rp are inaccurate.

Author(s)

Mathieu Ribatet

References

Ferro, C. and Segers, J. (2003). Inference for clusters of extreme values. Journal of the Royal Statistical Society B. **65**: 545–556.

See Also

fitmcgpd, acf, retlev

Examples

```
set.seed(123)
mc <- simmc(200, alpha = 0.5)
mc <- qgpd(mc, 0, 1, 0.25)
Mclog <- fitmcgpd(mc, 1)
par(mfrow=c(2,2))
rlMclog <- plot(Mclog)
rlMclog(T = 3)
```

---

plot.uvpot *Graphical Diagnostic: the Univariate GPD Model*

Description

Produces QQ-plot, Probability Plot and a Density Plot of the fitted model versus the empirical one. Another function computes the Return Level Plot of the fitted model.

Usage

```
## S3 method for class 'uvpot'
plot(x, npy, main, which = 1:4, ask = nb.fig <
length(which) && dev.interactive(),ci = TRUE, ...)
```

Arguments

x	A fitted object of class 'uvpot'. Generally, an object return by fitgpd
npy	The mean **N**umber of events **P**er **Y**ear - or more generally a block.
main	optional. A string vector corresponding to the title of each plot.
which	a numeric vector which specifies which plot must be drawn : '1' for Probability Plot, '2' for QQ-Plot,'3' for Density Plot and '4' for a Return Level Plot.
ask	Logical. If TRUE, user is asked before each plot.
ci	Logical. If TRUE, the simulated 95% confidence interval is plotted.
...	Other parameters to pass to the plot function.

Author(s)

Mathieu Ribatet

Examples

```
data(ardieres)
ardieres <- clust(ardieres, 4, 10 / 365, clust.max = TRUE)
fitted <- fitgpd(ardieres[, "obs"], 6, 'mle')
npy <- fitted$nat / 33.4 ##33.4 is the total record length (in year)
par(mfrow=c(2,2))
plot(fitted, npy = npy)
```

---

pp *Probability Probability Plot*

Description

pp is a generic function used to show probability-probability plot. The function invokes particular methods which depend on the class of the first argument. So the function makes a probability probability plot for univariate POT models.

Usage

```
pp(fitted, ...)

## S3 method for class 'uvpot'
pp(fitted, main, xlab, ylab, ci = TRUE, ...)
```

Arguments

fitted	A fitted object. When using the POT package, an object of class 'uvpot'. Most often, the return of the fitgpd function.
main	The title of the graphic. If missing, the title is set to "Probability plot".
xlab,ylab	The labels for the x and y axis. If missing, they are set to "Empirical" and "Model" respectively.
ci	Logical. If TRUE (the default), 95% intervals are plotted.
...	Other arguments to be passed to the plot function.

Details

The probability probability plot consists of plotting the theoretical probabilities in function of the empirical model ones. The theoretical probabilities are computed from the fitted GPD, while the empirical ones are computing from a particular plotting position estimator. This plotting position estimator is suited for the GPD case (Hosking, 1995) and are defined by:

$$p_{j:n} = \frac{j - 0.35}{n}$$

where n is the total number of observations.

If the theoretical model is correct, then points should be "near" the line $y = x$.

Value

A graphical window.

Author(s)

Mathieu Ribatet

References

Hosking, J. R. M. and Wallis, J. R. (1995). A comparison of unbiased and plotting-position estimators of L moments. Water Resources Research. **31**(8): 2019–2025.

See Also

qq, qq.uvpot

Examples

```
x <- rgpd(75, 1, 2, 0.1)
pwmb <- fitgpd(x, 1, "pwmb")
pp(pwmb)
```

---

print.bvpot *Printing bvpot objects*

Description

Print a "bvpot" object

Usage

```
## S3 method for class 'bvpot'
print(x, digits = max(3, getOption("digits") - 3), ...)
```

Arguments

x	An object of class 'bvpot'. Most often, returns of the fitbvgpd function.
digits	The number of digits to be printed.
...	Other options to be passed to the print function.

Value

Print on screen.

Author(s)

Mathieu Ribatet

See Also

print.uvpot, print.mcpot, print

Examples

```
set.seed(123)
x <- rgpd(500, 0, 1, 0.2)
y <- rgpd(500, 2, 0.5, -0.1)
Mlog <- fitbvgpd(cbind(x, y), c(0, 2))
Mlog
```

---

print.mcpot *Printing mcpot objects*

Description

Print an "mcpot" object

Usage

```
## S3 method for class 'mcpot'
print(x, digits = max(3, getOption("digits") - 3), ...)
```

Arguments

x	An object of class 'mcpot'. Most often, returns of the fitmcgpd function.
digits	The number of digits to be printed.
...	Other options to be passed to the print function.

Value

Print on screen.

Author(s)

Mathieu Ribatet

See Also

print.uvpot, print.bvpot, print

Examples

```
x <- simmc(1000, alpha = 0.5)
x <- qgpd(x, 0, 1, 0.15)
Mc <- fitmcgpd(x, 0)
Mc
```

`print.uvpot` *Printing uvpot objects*

Description

Print an "uvpot" object

Usage

```
## S3 method for class 'uvpot'
print(x, digits = max(3, getOption("digits") - 3), ...)
```

Arguments

`x`	An object of class `'uvpot'`. Most often, returns of the `fitgpd` function.
`digits`	The number of digits to be printed.
`...`	Other options to be passed to the `print` function.

Value

Print on screen.

Author(s)

Mathieu Ribatet

See Also

`print.bvpot`, `print.mcpot`, `print`

Examples

```
x <- rgpd(500, 0, 1, 0.2)
MLE <- fitgpd(x, 0)
MLE
```

`Profiled Confidence Intervals`
Profiled Confidence interval for the GP Distribution

Description

Compute profiled confidence intervals on parameter and return level for the GP distribution. This is achieved through the profile likelihood procedure.

Usage

```
gpd.pfshape(fitted, range, xlab, ylab, conf = 0.95, nrang = 100,
vert.lines = TRUE, ...)
gpd.pfscale(fitted, range, xlab, ylab, conf = 0.95, nrang = 100,
vert.lines = TRUE, ...)
gpd.pfrl(fitted, prob, range, thresh, xlab, ylab, conf = 0.95, nrang =
100, vert.lines = TRUE, ...)
```

Arguments

`fitted`	`R` object given by function `fitgpd`.
`prob`	The probability of non exceedance.
`range`	Vector of dimension two. It gives the lower and upper bound on which the profile likelihood is performed.
`thresh`	Optional. The threshold. Only needed with non constant threshold.
`xlab, ylab`	Optional Strings. Allows to label the x-axis and y-axis. If missing, default value are considered.
`conf`	Numeric. The confidence level.
`nrang`	Numeric. It specifies the number of profile likelihood computed on the whole range `range`.
`vert.lines`	Logical. If `TRUE` (the default), vertical lines are plotted.
`...`	Optional parameters to be passed to the `plot` function.

Value

Returns a vector of the lower and upper bound for the profile confidence interval. Moreover, a graphic of the profile likelihood function is displayed.

Author(s)

Mathieu Ribatet

References

Coles, S. (2001). *An Introduction to Statistical Modelling of Extreme Values*. Springer Series in Statistics. London.

See Also

gpd.fiscale, gpd.fishape, gpd.firl and confint

Examples

```
data(ardieres)
events <- clust(ardieres, u = 4, tim.cond = 8 / 365,
clust.max = TRUE)
MLE <- fitgpd(events[, "obs"], 4, 'mle')
gpd.pfshape(MLE, c(0, 0.8))
rp2prob(10, 2)
gpd.pfrl(MLE, 0.95, c(12, 25))
```

---

qq *Quantile Quantile Plot*

Description

qq is a generic function used to show quantile-quantile plot. The function invokes particular methods which depend on the class of the first argument. So the function makes a quantile quantile plot for univariate POT models.

Usage

```
qq(fitted, ...)

## S3 method for class 'uvpot'
qq(fitted, main, xlab, ylab, ci = TRUE, ...)
```

Arguments

fitted	A fitted object. When using the POT package, an object of class 'uvpot'. Most often, the return of the fitgpd function.
main	The title of the graphic. If missing, the title is set to "QQ-plot".
xlab,ylab	The labels for the x and y axis. If missing, they are set to "Model" and "Empirical" respectively.
ci	Logical. If TRUE (the default), 95% intervals are plotted.
...	Other arguments to be passed to the plot function.

Details

The quantile quantile plot consists of plotting the observed quantiles in function of the theoretical ones. The theoretical quantiles $Q_{Theo,j}$ are computed from the fitted GPD, that is:

$$Q_{Theo,j} = F^{-1}(p_j)$$

where F^{-1} is the fitted quantile function and p_j are empirical probabilities defined by :

$$p_{j:n} = \frac{j - 0.35}{n}$$

where n is the total number of observations - see Hosking (1995).

If the theoretical model is correct, then points should be "near" the line $y = x$.

Value

A graphical window.

Author(s)

Mathieu Ribatet

References

Hosking, J. R. M. and Wallis, J. R. (1995). A comparison of unbiased and plotting-position estimators of L moments. Water Resources Research. **31**(8): 2019–2025.

See Also

qq, qq.uvpot

Examples

```
x <- rgpd(75, 1, 2, 0.1)
pwmu <- fitgpd(x, 1, "pwmu")
qq(pwmu)
```

---

retlev *Return Level Plot*

Description

retlev is a generic function used to show return level plot. The function invokes particular methods which depend on the class of the first argument. So the function makes a return level plot for POT models.

Usage

```
retlev(fitted, ...)

## S3 method for class 'uvpot'
retlev(fitted, npy, main, xlab, ylab, xlimsup,
ci = TRUE, points = TRUE, ...)
## S3 method for class 'mcpot'
retlev(fitted, opy, exi, main, xlab, ylab, xlimsup,
...)
```

Arguments

`fitted`	A fitted object. When using the POT package, an object of class `'uvpot'` or `'mcpot'`. Most often, the return of `fitgpd` or `fitmcgpd` functions.
`npy`	The mean **N**umber of events **P**er **Y**ear (or more generally per block).if missing, setting it to 1.
`main`	The title of the graphic. If missing, the title is set to `"Return Level Plot"`.
`xlab,ylab`	The labels for the x and y axis. If missing, they are set to `"Return Period (Years)"` and `"Return Level"` respectively.
`xlimsup`	Numeric. The right limit for the x-axis. If missing, a suited value is computed.
`ci`	Logical. Should the 95% pointwise confidence interval be plotted?
`points`	Logical. Should observations be plotted?
`...`	Other arguments to be passed to the `plot` function.
`opy`	The number of **O**bservations **P**er **Y**ear (or more generally per block). If missing, it is set it to 365 i.e. daily values with a warning.
`exi`	Numeric. The extremal index. If missing, an estimate is given using the `fitexi` function.

Details

For class `"uvpot"`, the return level plot consists of plotting the theoretical quantiles in function of the return period (with a logarithmic scale for the x-axis). For the definition of the return period see the `prob2rp` function. Thus, the return level plot consists of plotting the points defined by:

$$(T(p), F^{-1}(p))$$

where $T(p)$ is the return period related to the non exceedance probability p, F^{-1} is the fitted quantile function.

If `points = TRUE`, the probabilities p_j related to each observation are computed using the following plotting position estimator proposed by Hosking (1995):

$$p_j = \frac{j - 0.35}{n}$$

where n is the total number of observations.

If the theoretical model is correct, the points should be "close" to the "return level" curve.

For class `"mcpot"`, let $X_1, \ldots, X_n$ be the first n observations from a stationary sequence with marginal distribution function F. Thus, we can use the following (asymptotic) approximation:

$$\Pr\left[\max\left\{X_1, \ldots, X_n\right\} \leq x\right] = \left[F(x)\right]^{n\theta}$$

where θ is the extremal index.

Thus, to obtain the T-year return level, we equate this equation to $1 - 1/T$ and solve for x.

Value

A graphical window. In addition, it returns invisibly the return level function.

Warning

For class "mcpot", though this is computationally expensive, we recommend to give the extremal index estimate using the dexi function. Indeed, there is a severe bias when using the Ferro and Segers (2003) estimator - as it is estimated using observation and not the Markov chain model.

Author(s)

Mathieu Ribatet

References

Hosking, J. R. M. and Wallis, J. R. (1995). A comparison of unbiased and plotting-position estimators of L moments. Water Resources Research. **31**(8): 2019–2025.

Ferro, C. and Segers, J. (2003). Inference for clusters of extreme values. Journal of the Royal Statistical Society B. **65**: 545–556.

See Also

prob2rp, fitexi.

Examples

```
#for uvpot class
x <- rgpd(75, 1, 2, 0.1)
pwmu <- fitgpd(x, 1, "pwmu")
rl.fun <- retlev(pwmu)
rl.fun(100)

#for mcpot class
data(ardieres)
Mcalog <- fitmcgpd(ardieres[,"obs"], 5, "alog")
retlev(Mcalog, opy = 990)
```

---

retlev.bvpot *Return Level Plot: Bivariate Case*

Description

Plot return levels for a fitted bivariate extreme value distribution.

Usage

```
## S3 method for class 'bvpot'
retlev(fitted, p = seq(0.75,0.95,0.05), main, n = 5000,
only.excess = FALSE, ...)
```

Arguments

`fitted`	An object of class `"bvpot"`. Most often, the return object of the `fitbvgpd` function.
`p`	A vector of probabilities for which return levels must be drawn.
`main`	The title of the graphic window. May be missing.
`n`	The number (default: 5000) of points needed to draw return levels lines.
`only.excess`	Logical. If `FALSE` (the default), all observations are plotted, otherwise, only exceedances above at least one of the two thresholds are plotted.
`...`	Other parameters to pass to the `plot` function.

Details

Any bivariate extreme value distribution has the Pickands' representation form i.e.:

$$G(y_1, y_2) = \exp\left[-\left(\frac{1}{z_1} + \frac{1}{z_2}\right) A(w)\right]$$

where z_i corresponds to y_i transformed to be unit Frechet distributed and $w = \frac{z_2}{z_1+z_2}$ which lies in $[0, 1]$.

Thus, for a fixed probability p and w, we have the corresponding z_1, z_2 values:

$$z_1 = -\frac{A(w)}{w \log(p)}$$

$$z_2 = \frac{z_1 w}{1 - w}$$

At last, the z_i are transformed back to their original scale.

Value

Plot return levels for a fitted bivariate extreme value distribution. Moreover, an invisible list is return which gives the points used to draw the current plot.

Author(s)

Mathieu Ribatet

See Also

`fitbvgpd`, `plot`

Examples

```
x <- rbvgpd(1000, alpha = 0.25, mar1 = c(0, 1, 0.25))
Mlog <- fitbvgpd(x, c(0, 0), "log")
retlev(Mlog)
```

Return Periods Tools *Converts Return Periods to Probability and Vice Versa*

Description

Compute return period from probability of non exceedance and vice versa.

Usage

```
rp2prob(retper, npy)
prob2rp(prob, npy)
```

Arguments

`retper`	The return period.
`prob`	the probability of non exceedance.
`npy`	The mean **N**umber of events **p**er **y**ear (block).

Details

The return period is defined by:

$$T = \frac{1}{npy(1-p)}$$

where npy is the mean number of events per year (block), p is the probability of non exceedance.

Value

Returns a table with mean numbers of events per year, return periods and probabilities of non exceedance associated.

Author(s)

Mathieu Ribatet

Examples

```
rp2prob(50, 1.8)
prob2rp(0.6, 2.2)
```

simmc *Simulate Markov Chains With Extreme Value Dependence Structures*

Description

Simulation of first order Markov chains, such that each pair of consecutive values has the dependence structure of one of nine parametric bivariate extreme value distributions.

Usage

```
simmc(n, alpha, model = "log", asCoef, asCoef1, asCoef2, margins =
"uniform")
```

Arguments

n	Number of observations.
alpha	Dependence parameter for the logistic, asymmetric logistic, negative logistic, asymmetric negative logistic, mixed and asymmetric mixed models.
asCoef,asCoef1,asCoef2	The asymmetric coefficients for the asymmetric logistic, asymmetric negative logistic and asymmetric mixed models.
model	The specified model; a character string. Must be either "log" (the default), "alog", "nlog", "anlog", "mix" or "amix", for the logistic, asymmetric logistic, negative logistic, asymmetric negative logistic, mixed and asymmetric mixed models respectively.
margins	The marginal distribution of each value; a character string. Must be either "uniform" (the default), "rweibull", "frechet" or "gumbel", for the uniform, standard reversed Weibull, standard Gumbel and standard Frechet distributions respectively.

Value

A numeric vector of length n.

Author(s)

Alec Stephenson (modified for the POT package by Mathieu Ribatet)

Examples

```
simmc(100, alpha = 0.1, model = "log")
simmc(100, alpha = 1.2, model = "nlog", margins = "gum")
```

simmcpot *Simulate an Markov Chain with a Fixed Extreme Value Dependence from a Fitted mcpot Object*

Description

Simulate a synthetic Markov chain from a fitted 'mcpot' object.

Usage

```
simmcpot(fitted, plot = TRUE, ...)
```

Arguments

fitted	An object of class 'mcpot'; most often the returned object of the fitmcgpd function.
plot	Logical. If TRUE (the default), the simulated Markov chain is plotted.
...	Other optional arguments to be passed to the plot function.

Details

The simulated Markov chain is computed as follows:

1. Simulate a Markov chain prob with uniform margins on (0,1) and with the fixed extreme value dependence given by fitted;
2. For all prob such as $prob \leq 1 - pat$, set $mc = NA$ (where pat is given by fitted$pat);
3. For all prob such as $prob \geq 1 - pat$, set $prob2 = \frac{prob-1+pat}{pat}$. Thus, prob2 are uniformly distributed on (0,1);
4. For all prob2, set mc = qgpd(prob2, thresh, scale, shape), where thresh, scale, shape are given by the fitted$threshold, fitted$param["scale"] and fitted$param["shape"] respectively.

Value

A Markov chain which has the same features as the fitted object. If plot = TRUE, the Markov chain is plotted.

Author(s)

Mathieu Ribatet

See Also

fitmcgpd, simmc

Examples

```
data(ardieres)
flows <- ardieres[,"obs"]

Mclog <- fitmcgpd(flows, 5)
par(mfrow = c(1,2))
idx <- which(flows <= 5)
flows[idx] <- NA
plot(flows, main = "Ardieres Data")
flowsSynth <- simmcpot(Mclog, main = "Simulated Data")
```

---

specdens *Spectral Density Plot*

Description

Plot the spectral density for a bivariate extreme value distribution or an extreme Markov chain model.

Usage

```
specdens(fitted, main, plot = TRUE, ...)
```

Arguments

fitted	An object of class 'bvpot' or 'mcpot'. Most often, the return object of the fitbvgpd or fitmcgpd function.
main	The title of the graphic window. May be missing.
plot	Logical. Should the spectral density be plotted? The default is to plot it.
...	Other options to be passed to the plot function.

Details

Any bivariate extreme value distribution has the following representation:

$$G(y_1, y_2) = \exp\left[-\int_0^1 \max\left(\frac{q}{z_1}, \frac{1-q}{z_2}\right) dH(q)\right]$$

where H holds:

$$\int_0^1 q dH(q) = \int_0^1 (1-q) dH(q) = 1$$

H is called the spectral measure with density h. Thus, h is called the spectral density. In addition, H has a total mass of 2.

For two independent random variables, the spectral measure consists of two points of mass 1 at $q = 0, 1$. For two perfect dependent random variables, the spectral measure consists of a single point of mass 2 at $q = 0.5$.

Value

Plot the spectral density for a fitted bivariate extreme value distribution. Moreover, the spectral density is returned invisibly.

Author(s)

Mathieu Ribatet

See Also

`retlev.bvpot`, `pickdep` and `plot.bvpot`

Examples

```
par(mfrow=c(1,2))
##Spectral density for a Markov Model
mc <- simmc(1000, alpha = 0.25, model = "log")
mc <- qgpd(mc, 0, 1, 0.1)
Mclog <- fitmcgpd(mc, 0, "log")
specdens(Mclog)
##Spectral density for a bivariate POT model
x <- rgpd(500, 5, 1, -0.1)
y <- rgpd(500, 2, 0.2, -0.25)
Manlog <- fitbvgpd(cbind(x,y), c(5,2), "anlog")
specdens(Manlog)
```

---

summary.pot *Compactly display the structure*

Description

Compactly display the structure of an object of class `'pot'`

Usage

```
## S3 method for class 'pot'
summary(object, ...)
```

Arguments

`object`	An object of class `'pot'`. Most often, this is an object return by the `fitgpd`, `fitbvgpd` and `fitmcgpd` functions.
`...`	Other arguments to be passed to the `str` function.

Value

Standard `summary` object: see `summary`.

Author(s)

Christophe Dutang

See Also

summary

Examples

```
set.seed(123)
x <- rgpd(500, 0, 1, -0.15)
mle <- fitgpd(x, 0)
summary(mle)
```

---

tailind.test *Testing for Tail Independence in Extreme Value Models*

Description

Several tests for tail independence (e.g. asymptotic independence) for a bivariate extreme value distribution

Usage

```
tailind.test(data, c = -0.1, emp.trans = TRUE, chisq.n.class = 4)
```

Arguments

data	A matrix with two columns given the data.
c	A negative numeric. Must be close to zero to approximate accurately asymptotic results.
emp.trans	Logical. If TRUE (the default), "data" is transformed to reverse exponential using empirical estimates. Otherwise, "data" is supposed to be reverse exponential distributed.
chisq.n.class	A numeric given the number of classes for the Chi squared test.

Details

These tests are based on an asymptotic results shown by Falk and Michel (2006). Let (X, Y) be a random vector which follows in its upper tail a bivariate extreme value distribution with reverse exponential margins. The conditional distribution function of $X + Y$, given that $X + Y > c$, converges to $F(t) = t^2$, $t \in [0, 1]$, if $c \to 0^-$ iff X and Y are asymptotically independent. Otherwise, the limit is $F(t) = t$

Value

This function returns a table with the Neymann-Pearson, Fisher, Kolmogorov-Smirnov and Chi-Square statistics and the related p-values.

Author(s)

Mathieu Ribatet

References

Falk, M. and Michel, Rene(2006) Testing for tail independence in extreme value models. *Annals of the Institute of Statistical Mathematics* **58**: 261–290

See Also

chimeas, specdens

Examples

```
##A total independence example
x <- rbvgpd(7000, alpha = 1, mar1 = c(0, 1, 0.25))
tailind.test(x)
##An asymptotically dependent example
y <- rbvgpd(7000, alpha = 0.75, model = "nlog", mar1 = c(0, 1, 0.25),
mar2 = c(2, 0.5, -0.15))
tailind.test(y)
##A perfect dependence example
z <- rnorm(7000)
tailind.test(cbind(z, 2*z - 5))
```

---

tcplot *Threshold Selection: The Threshold Choice Plot*

Description

Plots of parameter estimates at various thresholds for peaks over threshold modelling, using the Generalized Pareto or Point Process representation.

Usage

```
tcplot(data, u.range, cmax = FALSE, r = 1,
    ulow = -Inf, rlow = 1, nt = 25, which = 1:npar, conf = 0.95,
    lty = 1, lwd = 1, type = "b", cilty = 1, ask = nb.fig <
    length(which) && dev.interactive(), ...)
```

Arguments

data	A numeric vector.
u.range	A numeric vector of length two, giving the limits for the thresholds at which the model is fitted.
cmax	Logical; if FALSE (the default), the models are fitted using all exceedances over the thresholds. If TRUE, the models are fitted using cluster maxima.

r, ulow, rlow	Arguments used for the identification of clusters of exceedances. Ignored if cmax is FALSE (the default).
nt	The number of thresholds at which the model is fitted.
which	If a subset of the plots is required, specify a subset of the numbers 1:npar, where npar is the number of parameters.
conf	The (pointwise) confidence coefficient for the plotted confidence intervals. Use zero to suppress.
lty, lwd	The line type and width of the line connecting the parameter estimates.
type	The form taken by the line connecting the parameter estimates and the points denoting these estimates. Possible values include "b" (the default) for points joined by lines, "o" for over plotted points and lines, and "l" for an unbroken line with no points.
cilty	The line type of the lines depicting the confidence intervals.
ask	Logical; if TRUE, the user is asked before each plot.
...	Other arguments to be passed to the model fit function fitgpd.

Details

For each of the nt thresholds a peaks over threshold model is fitted using the function fitgpd. The maximum likelihood estimates for the shape and the modified scale parameter (modified by subtracting the shape multiplied by the threshold) are plotted against the thresholds. If the threshold u is a valid threshold to be used for peaks over threshold modelling, the parameter estimates depicted should be approximately constant above u.

Value

A list is invisibly returned. Each component is a matrix with three columns giving parameter estimates and confidence limits.

Author(s)

Stuart Coles and Alec Stephenson

References

Coles, S. (2001) *An Introduction to Statistical Modelling of Extreme Values*. Springer Series in Statistics. London.

See Also

fitgpd, mrlplot

Examples

```
data(ardieres)
ardieres <- clust(ardieres, 4, 10 / 365, clust.max = TRUE)
flows <- ardieres[, "obs"]
par(mfrow=c(1,2))
tcplot(flows, u.range = c(0, 15) )
```

`ts2tsd` *Mobile Window on a Time Series*

Description

This function performs a mobile average windows on the whole time series. Thus, if the time series represents flood discharges, it returns the averaged discharges over a specific duration.

Usage

```
ts2tsd(ts, d, vol = FALSE, method = "linear")
```

Arguments

`ts`	The time series. It consists of two columns: one named `"time"` and the second `"obs"`.
`d`	Numeric which corresponds of the duration for the mobile window.
`vol`	Logical. If `FALSE` -the default, average values are computed, else volumes.
`method`	Specifies the interpolation method to be used. Choices are `"linear"` or `"constant"`.

Details

A mobile windows of length `d` is performed on the whole time sire. The "discrete" time series in first transformed in a function; interpolation are obtained using the `approx` function. Thus, if f(t) is the function representing the time series, volume over duration `d` is defined by:

$$vol(t) = \int_{t-d/2}^{t+d/2} f(u)du$$

while average values are:

$$ave(t) = \frac{1}{d}\int_{t-d/2}^{t+d/2} f(u)du$$

Value

Returns a time series like object `ts`. In particular `ts[,"time"]` and `tsd[,"time"]` are identical.

Warnings

Please note that as the time series is interpolated, caution should be taken if the method to interpolate is not efficient.

Note that object `d` should have the same unit than `ts[,"time"]`.

Author(s)

Mathieu Ribatet

See Also

approx

Examples

```
data(ardieres)
tsd <- ts2tsd(ardieres, 3 / 365)
plot(ardieres, type = "l", col = "blue")
lines(tsd, col = "green")
```

---

tsdep.plot *Diagnostic for Dependence within Time Series Extremes*

Description

A diagnostic tool to assess for short range asymptotic dependence within a stationary time series.

Usage

```
tsdep.plot(data, u, ..., xlab, ylab, n.boot = 100, show.lines = TRUE,
lag.max, ci = 0.95, block.size = 5 * lag.max, angle = 90, arrow.length =
0.1)
```

Arguments

data	The time series observations.
u	The threshold.
...	Optional arguments to be passed to the plot function.
xlab,ylab	The x and y-axis labels.
n.boot	Numeric. The number of replicates to compute the bootstrap confidence interval.
show.lines	Logical. If TRUE (the default), the theoretical lines for the asymptotic dependence and "near" independence are drawn.
lag.max	The maximum lag to be explored - may be missing.
ci	The level for the bootstrap confidence interval. The default is the 95% confidence interval.
block.size	The size for the contiguous bootstrap approach.
angle	The angle at the end of the error bar. If 0, error bars are only segments.
arrow.length	The length to be passed in the function arrows.

Details

Let `X_t` be a stationary sequence of unit Frechet random variables. By stationarity, the joint survivor function $\overline{F}_\tau(\cdot,\cdot)$ of $(X_t, X_{t+\tau})$ does not depend on t.

One parametric representation for $\overline{F}_\tau(\cdot,\cdot)$ is given by

$$\overline{F}_\tau(s,s) = L_\tau(s) s^{-1/\eta_\tau}$$

for some parameter $\eta_\tau \in (0,1]$ and a slowly varying function L_τ.

The Λ_τ statistic is defined by

$$\Lambda_\tau = 2\eta_\tau - 1$$

This statistic belongs to (-1,1] and is a measure of extremal dependence. $\Lambda_\tau = 1$ corresponds to asymptotic dependence, $0 < \Lambda_\tau < 1$ to positive extremal association, $\Lambda_\tau = 0$ to "near" independence and $\Lambda_\tau < 0$ to negative extremal association.

Value

This function plot the Λ_τ statictics against the lag. Bootstrap confidence intervals are also drawn. The function returns invisibly this statistic and the confidence bounds.

Author(s)

Mathieu Ribatet

References

Ledford, A. and Tawn, J. (2003) Diagnostics for dependence within time series extremes. *L. R. Statist. Soc. B.* **65**, Part 2, 521–543.

Ledford, A. and Tawn, J (1996) Statistics for near independence in multivariate extreme values. *Biometrika* **83** 169–187.

See Also

`chimeas`, `tailind.test`

Examples

```
##An independent case
tsdep.plot(runif(5000), u = 0.95, lag.max = 5)
##Asymptotic dependence
mc <- simmc(5000, alpha = 0.2)
tsdep.plot(mc, u = 0.95, lag.max = 5)
```

Index