The evd Package

September 6, 2004

Version 2.1-6

Date 2004-08-30

Title Functions for extreme value distributions

Author Alec Stephenson. Function fbvpot by Chris Ferro.

Maintainer Alec Stephenson <alec_stephenson@hotmail.com>

Depends R (>= 1.5.0)

Description Extends simulation, distribution, quantile and density functions to univariate and multivariate parametric extreme value distributions, and provides fitting functions which calculate maximum likelihood estimates for univariate and bivariate maxima models, and for univariate and bivariate threshold models.

License GPL (Version 2 or above)

URL http://www.maths.lancs.ac.uk/~stephena/

R topics documented:

abvnonpar 2
abvpar 5
anova.evd 7
atvnonpar 8
atvpar 9
bvevd 11
chiplot 15
clusters 18
evd-internal 20
evmc 20
exi 21
extreme 23
failure 24
fbvevd 24
fbvpot 28
fextreme 31
fgev 33
forder 35
fox 37

fpot . . . 37
frechet . . . 40
gev . . . 41
gpd . . . 42
gumbel . . . 44
lisbon . . . 45
marma . . . 45
mrlplot . . . 47
mvevd . . . 48
ocmulgee . . . 50
oldage . . . 51
order . . . 52
oxford . . . 53
plot.bvevd . . . 53
plot.profile.evd . . . 55
plot.profile2d.evd . . . 56
plot.uvevd . . . 57
portpirie . . . 59
profile.evd . . . 60
profile2d.evd . . . 61
rweibull . . . 62
sask . . . 63
sealevel . . . 64
sealevel2 . . . 64
tcplot . . . 65
uccle . . . 66
venice . . . 67

Index **68**

`abvnonpar` *Non-parametric Estimates for Dependence Functions of the Bivariate Extreme Value Distribution*

Description

Calculate or plot non-parametric estimates for the dependence function A of the bivariate extreme value distribution.

Usage

```
abvnonpar(x = 0.5, data, nsloc1 = NULL, nsloc2 = NULL,
    method = c("cfg", "pickands", "deheuvels", "hall", "tdo"),
    convex = FALSE, wf = function(t) t, plot = FALSE,
    add = FALSE, lty = 1, lwd = 1, col = 1, blty = 3, xlim = c(0, 1),
    ylim = c(0.5, 1), xlab = "", ylab = "", ...)
```

Arguments

`x`	A vector of values at which the dependence function is evaluated (ignored if plot or add is `TRUE`). $A(1/2)$ is returned by default since it is often a useful summary of dependence.
`data`	A matrix or data frame with two columns, which may contain missing values.
`nsloc1, nsloc2`	A data frame with the same number of rows as `data`, for linear modelling of the location parameter on the first/second margin. The data frames are treated as covariate matrices, excluding the intercept. A numeric vector can be given as an alternative to a single column data frame.
`method`	The estimation method (see **Details**). The `"cfg"` method is used by default.
`convex`	Logical; take the convex minorant?
`wf`	The weight function used in the `"cfg"` method (see **Details**). The function must be vectorized.
`plot`	Logical; if `TRUE` the function is plotted. The x and y values used to create the plot are returned invisibly. If `plot` and `add` are `FALSE` (the default), the arguments following `add` are ignored.
`add`	Logical; add to an existing plot? The existing plot should have been created using either `abvnonpar` or abvpar, the latter of which plots (or calculates) the dependence function for a number of parametric models.
`lty, blty`	Function and border line types. Set `blty` to zero to omit the border.
`lwd`	Line width.
`col`	Line colour.
`xlim, ylim`	x and y-axis limits.
`xlab, ylab`	x and y-axis labels.
`...`	Other high-level graphics parameters to be passed to `plot`.

Details

The dependence function $A(\cdot)$ of the bivariate extreme value distribution is defined in abvpar. Non-parametric estimates are constructed as follows. Suppose (z_{i1}, z_{i2}) for $i = 1, \ldots, n$ are n bivariate observations that are passed using the `data` argument. The marginal parameters are estimated (under the assumption of independence) and the data is transformed using

$$y_{i1} = \{1 + \hat{s}_1(z_{i1} - \hat{a}_1)/\hat{b}_1\}_+^{-1/\hat{s}_1}$$

and

$$y_{i2} = \{1 + \hat{s}_2(z_{i2} - \hat{a}_2)/\hat{b}_2\}_+^{-1/\hat{s}_2}$$

for $i = 1, \ldots, n$, where $(\hat{a}_1, \hat{b}_1, \hat{s}_1)$ and $(\hat{a}_2, \hat{b}_2, \hat{s}_2)$ are the maximum likelihood estimates for the location, scale and shape parameters on the first and second margins. If `nsloc1` or `nsloc2` are given, the location parameters may depend on i (see `fgev`).

Five different estimators of the dependence function can be implemented. They are defined (on $0 \leq w \leq 1$) as follows.

`method = "cfg"` (Caperaa, Fougeres and Genest, 1997)

$$A_c(w) = \exp\left\{\{1 - p(w)\}\int_0^w \frac{H(x) - x}{x(1-x)}\,\mathrm{d}x - p(w)\int_w^1 \frac{H(x) - x}{x(1-x)}\,\mathrm{d}x\right\}$$

`method = "pickands"` (Pickands, 1981)

$$A_p(w) = n\left\{\sum_{i=1}^{n} \min\left(\frac{y_{i1}}{w}, \frac{y_{i2}}{1-w}\right)\right\}^{-1}$$

`method = "deheuvels"` (Deheuvels, 1991)

$$A_d(w) = n\left\{\sum_{i=1}^{n} \min\left(\frac{y_{i1}}{w}, \frac{y_{i2}}{1-w}\right) - w\sum_{i=1}^{n} y_{i1} - (1-w)\sum_{i=1}^{n} y_{i2} + n\right\}^{-1}$$

`method = "hall"` (Hall and Tajvidi, 2000)

$$A_h(w) = n\left\{\sum_{i=1}^{n} \min\left(\frac{y_{i1}}{\bar{y}_1 w}, \frac{y_{i2}}{\bar{y}_2(1-w)}\right)\right\}^{-1}$$

`method = "tdo"` (Tiago de Oliveira, 1997)

$$A_t(w) = 1 - \frac{1}{1+\log n}\sum_{i=1}^{n} \min\left(\frac{w}{1+ny_{i1}}, \frac{1-w}{1+ny_{i2}}\right)$$

In the estimator $A_h(\cdot)$, $\bar{y}_j = n^{-1}\sum_{i=1}^{n} y_{ij}$ for $j = 1, 2$. In the estimator $A_c(\cdot)$, $H(x)$ is the empirical distribution function of $x_1, \ldots, x_n$, where $x_i = y_{i1}/(y_{i1}+y_{i2})$ for $i = 1, \ldots, n$, and $p(w)$ is any bounded function on $[0, 1]$, which can be specified using the argument `wf`. By default `wf` is the identity function.

Let $A_n(\cdot)$ be any estimator of $A(\cdot)$. The constraint $A_n(0) = A_n(1) = 1$ is satisfied by $A_d(\cdot)$, $A_t(\cdot)$ and $A_h(\cdot)$, and by $A_c(\cdot)$ when $p(0) = 0$ and $p(1) = 1$. None of the estimators satisfy $\max(w, 1-w) \leq A_n(w) \leq 1$ for all $0 \leq w \leq 1$. An obvious modification is

$$A_n^{'}(w) = \min(1, \max\{A_n(w), w, 1-w\}).$$

This modification is always implemented.

$A_t(w)$ is the only estimator that is convex. Convex estimators can be derived from other methods by taking the convex minorant, which can be achieved by setting `convex` to `TRUE`.

Value

`abvnonpar` calculates or plots a non-parametric estimate of the dependence function of the bivariate extreme value distribution.

Note

Appendix A of the User's Guide contains a short simulation study that compares the estimators defined above. The estimators $A_p(\cdot)$, $A_d(\cdot)$ and $A_h(\cdot)$ are very similar, and may not be distinguishable when plotted.

References

Caperaa, P. Fougeres, A.-L. and Genest, C. (1997) A non-parametric estimation procedure for bivariate extreme value copulas. *Biometrika*, **84**, 567–577.

Deheuvels, P. (1991) On the limiting behaviour of the Pickands estimator for bivariate extreme-value distributions. *Statist. Probab. Letters*, **12**, 429–439.

Hall, P. and Tajvidi, N. (2000) Distribution and dependence-function estimation for bivariate extreme-value distributions. *Bernoulli*, **6**, 835–844.

Pickands, J. (1981) Multivariate extreme value distributions. *Proc. 43rd Sess. Int. Statist. Inst.*, **49**, 859–878.

Tiago de Oliveira, J. (1997) *Statistical Analysis of Extremes.* Pendor.

See Also

abvpar, atvnonpar, fgev

Examples

```
bvdata <- rbvevd(100, dep = 0.7, model = "log")
abvnonpar(seq(0, 1, length = 10), data = bvdata, convex = TRUE)
abvnonpar(data = bvdata, method = "d", plot = TRUE)

M1 <- fitted(fbvevd(bvdata, model = "log"))
abvpar(dep = M1["dep"], model = "log", plot = TRUE)
abvnonpar(data = bvdata, add = TRUE, lty = 2)
```

---

abvpar — *Parametric Dependence Functions of Bivariate Extreme Value Models*

Description

Calculate or plot the dependence function A for eight parametric bivariate extreme value models.

Usage

```
abvpar(x = 0.5, dep, asy = c(1,1), alpha, beta, model = "log",
     plot = FALSE, add = FALSE, lty = 1, lwd = 1, col = 1, blty = 3,
     xlim = c(0,1), ylim = c(0.5,1), xlab = "", ylab = "", ...)
```

Arguments

`x`	A vector of values at which the dependence function is evaluated (ignored if plot or add is `TRUE`). $A(1/2)$ is returned by default since it is often a useful summary of dependence.
`dep`	Dependence parameter for the logistic, asymmetric logistic, Husler-Reiss, negative logistic and asymmetric negative logistic models.
`asy`	A vector of length two, containing the two asymmetry parameters for the asymmetric logistic and asymmetric negative logistic models.
`alpha, beta`	Alpha and beta parameters for the bilogistic, negative bilogistic and Coles-Tawn models.
`model`	The specified model; a character string. Must be either `"log"` (the default), `"alog"`, `"hr"`, `"neglog"`, `"aneglog"`, `"bilog"`, `"negbilog"` or `"ct"` (or any unique partial match), for the logistic, asymmetric logistic, Husler-Reiss, negative logistic, asymmetric negative logistic, bilogistic, negative bilogistic and Coles-Tawn models respectively. The definition of

	each model is given in rbvevd. If parameter arguments are given that do not correspond to the specified model those arguments are ignored, with a warning.
plot	Logical; if TRUE the function is plotted. The x and y values used to create the plot are returned invisibly. If plot and add are FALSE (the default), the arguments following add are ignored.
add	Logical; add to an existing plot? The existing plot should have been created using either abvpar or abvnonpar, the latter of which plots (or calculates) a non-parametric estimate of the dependence function.
lty, blty	Function and border line types. Set blty to zero to omit the border.
lwd	Line width.
col	Line colour.
xlim, ylim	x and y-axis limits.
xlab, ylab	x and y-axis labels.
...	Other high-level graphics parameters to be passed to plot.

Details

Any bivariate extreme value distribution can be written as

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

for some function $A(\cdot)$ defined on $[0, 1]$, where

$$y_i = \{1 + s_i(z_i - a_i)/b_i\}^{-1/s_i}$$

for $1 + s_i(z_i - a_i)/b_i > 0$ and $i = 1, 2$, with the (generalized extreme value) marginal parameters given by (a_i, b_i, s_i), $b_i > 0$. If $s_i = 0$ then y_i is defined by continuity.

$A(\cdot)$ is called (by some authors) the dependence function. It follows that $A(0) = A(1) = 1$, and that $A(\cdot)$ is a convex function with $\max(x, 1 - x) \leq A(x) \leq 1$ for all $0 \leq x \leq 1$. The lower and upper limits of A are obtained under complete dependence and independence respectively. $A(\cdot)$ does not depend on the marginal parameters.

Value

abvpar calculates or plots the dependence function for one of eight parametric bivariate extreme value models, at specified parameter values.

See Also

abvnonpar, fbvevd, rbvevd, atvpar

Examples

```
abvpar(dep = 2.7, model = "hr")
abvpar(seq(0,1,0.25), dep = 0.3, asy = c(.7,.9), model = "alog")
abvpar(alpha = 0.3, beta = 1.2, model = "negbi", plot = TRUE)

bvdata <- rbvevd(100, dep = 0.7, model = "log")
M1 <- fitted(fbvevd(bvdata, model = "log"))
abvpar(dep = M1["dep"], model = "log", plot = TRUE)
abvnonpar(data = bvdata, add = TRUE, lty = 2)
```

`anova.evd` *Compare Nested EVD Objects*

Description

Compute an analysis of deviance table for two or more nested evd objects.

Usage

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

Arguments

`object`	An object of class `"evd"`.
`object2`	An object of class `"evd"` that represents a model nested within `object`.
`...`	Further successively nested objects.

Value

An object of class `c("anova", "data.frame")`, with one row for each model, and the following five columns

`M.Df`	The number of parameters.
`Deviance`	The deviance.
`Df`	The number of parameters of the model in the previous row minus the number of parameters.
`Chisq`	The deviance minus the deviance of the model in the previous row.
`Pr(>chisq)`	The p-value calculated by comparing the quantile `Chisq` with a chi-squared distribution on `Df` degrees of freedom.

Warning

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

See Also

`fbvevd`, `fextreme`, `fgev`, `forder`

Examples

```
uvdata <- rgev(100, loc = 0.13, scale = 1.1, shape = 0.2)
trend <- (-49:50)/100
M1 <- fgev(uvdata, nsloc = trend)
M2 <- fgev(uvdata)
M3 <- fgev(uvdata, shape = 0)
anova(M1, M2, M3)
```

```
bvdata <- rbvevd(100, dep = 0.75, model = "log")
M1 <- fbvevd(bvdata, model = "log")
M2 <- fbvevd(bvdata, model = "log", dep = 0.75)
anova(M1, M2)
```

`atvnonpar` *Non-parametric Estimates for Dependence Functions of the Trivariate Extreme Value Distribution*

Description

Calculate or plot non-parametric estimates for the dependence function A of the trivariate extreme value distribution.

Usage

```
atvnonpar(x = rep(1/3,3), data, nsloc1 = NULL, nsloc2 = NULL, nsloc3 = NULL,
    method = c("pickands", "deheuvels", "hall"), plot = FALSE,
    col = heat.colors(12), blty = 0, grid = if(blty) 150 else 50,
    lower = 1/3, ord = 1:3, lab = as.character(1:3), lcex = 1)
```

Arguments

`x`	A vector of length three or a matrix with three columns, in which case the dependence function is evaluated across the rows (ignored if plot is `TRUE`). The elements/rows of the vector/matrix should be positive and should sum to one, or else they should have a positive sum, in which case the rows are rescaled and a warning is given. $A(1/3, 1/3, 1/3)$ is returned by default since it is often a useful summary of dependence.
`data`	A matrix or data frame with three columns, which may contain missing values.
`nsloc1, nsloc2, nsloc3`	A data frame with the same number of rows as `data`, for linear modelling of the location parameter on the first/second/third margin. The data frames are treated as covariate matrices, excluding the intercept. A numeric vector can be given as an alternative to a single column data frame.
`method`	The estimation method; a character string. Must be either `"pickands"` (the default), `"deheuvels"` or `"hall"` (or any unique partial match). The three estimators are very similar, and may not be distinguishable when plotted.
`plot`	Logical; if `TRUE` the function is plotted. The minimum (evaluated) value is returned invisibly. If `FALSE` (the default), the following arguments are ignored.
`col`	A list of colours (see `image`). The first colours in the list represent smaller values, and hence stronger dependence. Each colour represents an equally spaced interval between `lower` and one.
`blty`	The border line type, for the border that surrounds the triangular image. By default `blty` is zero, so no border is plotted. Plotting a border leads to (by default) an increase in `grid` (and hence computation time), to ensure that the image fits within it.

`grid`	For plotting, the function is evaluated at `grid^2` points.
`lower`	The minimum value for which colours are plotted. By default `lower` = 1/3 as this is the theoretical minimum of the dependence function of the trivariate extreme value distribution.
`ord`	A vector of length three, which should be a permutation of the set $\{1, 2, 3\}$. The points $(1, 0, 0)$, $(0, 1, 0)$ and $(0, 0, 1)$ (the vertices of the simplex) are depicted clockwise from the top in the order defined by `ord`.
`lab`	A character vector of length three, in which case the `i`th margin is labelled using the `i`th component, or `NULL`, in which case no labels are given. By default, `lab` is `as.character(1:3)`. The actual location of the margins, and hence the labels, is defined by `ord`.
`lcex`	A numerical value giving the amount by which the labels should be scaled relative to the default. Ignored if `lab` is `NULL`.

Value

`atvnonpar` calculates or plots a non-parametric estimate of the dependence function of the trivariate extreme value distribution.

Note

The rows of `data` that contain missing values are not used in the estimation of the dependence structure, but every non-missing value is used in estimating the generalized extreme value margins.

The dependence function of the trivariate extreme value distribution is defined in `atvpar`. The function `atvpar` calculates and plots dependence functions of trivariate logistic and trivariate asymmetric logistic models.

See Also

`atvpar`, `abvnonpar`, `fgev`

Examples

```
s3pts <- matrix(rexp(30), nrow = 10, ncol = 3)
s3pts <- s3pts/rowSums(s3pts)
sdat <- rmvevd(100, dep = 0.6, model = "log", d = 3)
atvnonpar(s3pts, sdat)

## Not run: atvnonpar(data = sdat, plot = TRUE)
## Not run: atvnonpar(data = sdat, plot = TRUE, ord = c(2,3,1), lab = LETTERS[1:3])
## Not run: atvpar(dep = 0.6, model = "log", plot = TRUE)
## Not run: atvpar(dep = 0.6, model = "log", plot = TRUE, blty = 1)
```

---

atvpar — *Parametric Dependence Functions of Trivariate Extreme Value Models*

Description

Calculate or plot the dependence function A for the trivariate logistic and trivariate asymmetric logistic models.

Usage

```
atvpar(x = rep(1/3,3), dep, asy, model = c("log", "alog"), plot =
    FALSE, col = heat.colors(12), blty = 0, grid = if(blty) 150 else 50,
    lower = 1/3, ord = 1:3, lab = as.character(1:3), lcex = 1)
```

Arguments

`x`	A vector of length three or a matrix with three columns, in which case the dependence function is evaluated across the rows (ignored if plot is `TRUE`). The elements/rows of the vector/matrix should be positive and should sum to one, or else they should have a positive sum, in which case the rows are rescaled and a warning is given. $A(1/3, 1/3, 1/3)$ is returned by default since it is often a useful summary of dependence.
`dep`	The dependence parameter(s). For the logistic model, should be a single value. For the asymmetric logistic model, should be a vector of length four, or a single value, in which case the value is used for each of the four parameters (see `rmvevd`).
`asy`	The asymmetry parameters for the asymmetric logistic model. Should be a list with seven vector elements; three of length one, three of length two and one of length three, containing the asymmetry parameters for each separate component (see `rmvevd` and **Examples**).
`model`	The specified model; a character string. Must be either `"log"` (the default) or `"alog"` (or any unique partial match), for the logistic and asymmetric logistic models respectively. The definition of each model is given (for general dimensions) in `rmvevd`.
`plot`	Logical; if `TRUE` the function is plotted. The minimum (evaluated) value is returned invisibly. If `FALSE` (the default), the following arguments are ignored.
`col`	A list of colours (see `image`). The first colours in the list represent smaller values, and hence stronger dependence. Each colour represents an equally spaced interval between `lower` and one.
`blty`	The border line type, for the border that surrounds the triangular image. By default `blty` is zero, so no border is plotted. Plotting a border leads to (by default) an increase in `grid` (and hence computation time), to ensure that the image fits within it.
`grid`	For plotting, the function is evaluated at `grid^2` points.
`lower`	The minimum value for which colours are plotted. By defualt `lower` $= 1/3$ as this is the theoretical minimum of the dependence function of the trivariate extreme value distribution.
`ord`	A vector of length three, which should be a permutation of the set $\{1, 2, 3\}$. The points $(1, 0, 0)$, $(0, 1, 0)$ and $(0, 0, 1)$ (the vertices of the simplex) are depicted clockwise from the top in the order defined by `ord`.
`lab`	A character vector of length three, in which case the `i`th margin is labelled using the `i`th component, or `NULL`, in which case no labels are given. The actual location of the margins, and hence the labels, is defined by `ord`.
`lcex`	A numerical value giving the amount by which the labels should be scaled relative to the default. Ignored if `lab` is `NULL`.

Details

Let $z = (z_1, z_2, z_3)$ and $w = (w_1, w_2, w_3)$. Any trivariate extreme value distribution can be written as

$$G(z) = \exp\left\{-\left\{\sum_{j=1}^{3} y_j\right\} A\left(\frac{y_1}{\sum_{j=1}^{3} y_j}, \frac{y_2}{\sum_{j=1}^{3} y_j}, \frac{y_3}{\sum_{j=1}^{3} y_j}\right)\right\}$$

for some function A defined on the simplex $S_3 = \{w \in R_+^3 : \sum_{j=1}^{3} w_j = 1\}$, where

$$y_i = \{1 + s_i(z_i - a_i)/b_i\}^{-1/s_i}$$

for $1 + s_i(z_i - a_i)/b_i > 0$ and $i = 1, 2, 3$, and where the (generalized extreme value) marginal parameters are given by (a_i, b_i, s_i), $b_i > 0$. If $s_i = 0$ then y_i is defined by continuity.

A is called (by some authors) the dependence function. It follows that $A(1, 0, 0) = A(0, 1, 0) = A(0, 0, 1) = 1$, and that A is a convex function with $\max(w_1, w_2, w_3) \leq A(w) \leq 1$ for all w in S_3. The lower and upper limits of A are obtained under complete dependence and mutual independence respectively. A does not depend on the marginal parameters.

Value

`atvpar` calculates or plots the dependence function for the trivariate logistic and trivariate asymmetric logistic models, at specified parameter values.

See Also

`atvnonpar`, `abvpar`, `rmvevd`, `image`

Examples

```
atvpar(dep = 0.5, model = "log")
s3pts <- matrix(rexp(30), nrow = 10, ncol = 3)
s3pts <- s3pts/rowSums(s3pts)
atvpar(s3pts, dep = 0.5, model = "log")
## Not run: atvpar(dep = 0.05, model = "log", plot = TRUE, blty = 1)
atvpar(dep = 0.95, model = "log", plot = TRUE, lower = 0.94)

asy <- list(.4, .1, .6, c(.3,.2), c(.1,.1), c(.4,.1), c(.2,.3,.2))
atvpar(s3pts, dep = 0.15, asy = asy, model = "alog")
atvpar(dep = 0.15, asy = asy, model = "al", plot = TRUE, lower = 0.7)
```

---

bvevd *Parametric Bivariate Extreme Value Distributions*

Description

Density function, distribution function and random generation for eight parametric bivariate extreme value models.

Usage

```
dbvevd(x, dep, asy = c(1, 1), alpha, beta, model = "log",
    mar1 = c(0, 1, 0), mar2 = mar1, log = FALSE)
pbvevd(q, dep, asy = c(1, 1), alpha, beta, model = "log",
    mar1 = c(0, 1, 0), mar2 = mar1, lower.tail = TRUE)
rbvevd(n, dep, asy = c(1, 1), alpha, beta, model = "log",
    mar1 = c(0, 1, 0), mar2 = mar1)
```

Arguments

`x, q`	A vector of length two or a matrix with two columns, in which case the density/distribution is evaluated across the rows.
`n`	Number of observations.
`dep`	Dependence parameter for the logistic, asymmetric logistic, Husler-Reiss, negative logistic and asymmetric negative logistic models.
`asy`	A vector of length two, containing the two asymmetry parameters for the asymmetric logistic and asymmetric negative logistic models.
`alpha, beta`	Alpha and beta parameters for the bilogistic, negative bilogistic and Coles-Tawn models.
`model`	The specified model; a character string. Must be either `"log"` (the default), `"alog"`, `"hr"`, `"neglog"`, `"aneglog"`, `"bilog"`, `"negbilog"` or `"ct"` (or any unique partial match), for the logistic, asymmetric logistic, Husler-Reiss, negative logistic, asymmetric negative logistic, bilogistic, negative bilogistic and Coles-Tawn models respectively. If parameter arguments are given that do not correspond to the specified model those arguments are ignored, with a warning.
`mar1, mar2`	Vectors of length three containing marginal parameters, or matrices with three columns where each column represents a vector of values to be passed to the corresponding marginal parameter.
`log`	Logical; if `TRUE`, the log density is returned.
`lower.tail`	Logical; if `TRUE` (default), the distribution function is returned; the survivor function is returned otherwise.

Details

Define

$$y_i = y_i(z_i) = \{1 + s_i(z_i - a_i)/b_i\}^{-1/s_i}$$

for $1 + s_i(z_i - a_i)/b_i > 0$ and $i = 1, 2$, where the marginal parameters are given by `mari` $= (a_i, b_i, s_i)$, $b_i > 0$. If $s_i = 0$ then y_i is defined by continuity.

In each of the bivariate distributions functions $G(z_1, z_2)$ given below, the univariate margins are generalized extreme value, so that $G(z_i) = \exp(-y_i)$ for $i = 1, 2$. If $1 + s_i(z_i - a_i)/b_i \leq 0$ for some $i = 1, 2$, the value z_i is either greater than the upper end point (if $s_i < 0$), or less than the lower end point (if $s_i > 0$), of the ith univariate marginal distribution.

`model = "log"` (Gumbel, 1960)

The bivariate logistic distribution function with parameter `dep` $= r$ is

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

where $0 < r \leq 1$. This is a special case of the bivariate asymmetric logistic model. Complete dependence is obtained in the limit as r approaches zero. Independence is obtained when $r = 1$.

`model = "alog"` (Tawn, 1988)

The bivariate asymmetric logistic distribution function with parameters `dep` $= r$ and `asy` $= (t_1, t_2)$ is

$$G(z_1, z_2) = \exp\left\{-(1-t_1)y_1 - (1-t_2)y_2 - [(t_1 y_1)^{1/r} + (t_2 y_2)^{1/r}]^r\right\}$$

where $0 < r \leq 1$ and $0 \leq t_1, t_2 \leq 1$. When $t_1 = t_2 = 1$ the asymmetric logistic model is equivalent to the logistic model. Independence is obtained when either $r = 1$, $t_1 = 0$ or $t_2 = 0$. Complete dependence is obtained in the limit when $t_1 = t_2 = 1$ and r approaches zero. Different limits occur when t_1 and t_2 are fixed and r approaches zero.

`model = "hr"` (Husler and Reiss, 1989)

The Husler-Reiss distribution function with parameter `dep` $= r$ is

$$G(z_1, z_2) = \exp\left(-y_1\Phi\{r^{-1} + \tfrac{1}{2}r[\log(y_1/y_2)]\} - y_2\Phi\{r^{-1} + \tfrac{1}{2}r[\log(y_2/y_1)]\}\right)$$

where $\Phi(\cdot)$ is the standard normal distribution function and $r > 0$. Independence is obtained in the limit as r approaches zero. Complete dependence is obtained as r tends to infinity.

`model = "neglog"` (Galambos, 1975)

The bivariate negative logistic distribution function with parameter `dep` $= r$ is

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

where $r > 0$. This is a special case of the bivariate asymmetric negative logistic model. Independence is obtained in the limit as r approaches zero. Complete dependence is obtained as r tends to infinity. The earliest reference to this model appears to be Galambos (1975, Section 4).

`model = "aneglog"` (Joe, 1990)

The bivariate asymmetric negative logistic distribution function with parameters parameters `dep` $= r$ and `asy` $= (t_1, t_2)$ is

$$G(z_1, z_2) = \exp\left\{-y_1 - y_2 + [(t_1 y_1)^{-r} + (t_2 y_2)^{-r}]^{-1/r}\right\}$$

where $r > 0$ and $0 < t_1, t_2 \leq 1$. When $t_1 = t_2 = 1$ the asymmetric negative logistic model is equivalent to the negative logistic model. Independence is obtained in the limit as either r, t_1 or t_2 approaches zero. Complete dependence is obtained in the limit when $t_1 = t_2 = 1$ and r tends to infinity. Different limits occur when t_1 and t_2 are fixed and r tends to infinity. The earliest reference to this model appears to be Joe (1990), who introduces a multivariate extreme value distribution which reduces to $G(z_1, z_2)$ in the bivariate case.

`model = "bilog"` (Smith, 1990)

The bilogistic distribution function with parameters `alpha` $= \alpha$ and `beta` $= \beta$ is

$$G(z_1, z_2) = \exp\left\{-y_1 q^{1-\alpha} - y_2(1-q)^{1-\beta}\right\}$$

where $q = q(y_1, y_2; \alpha, \beta)$ is the root of the equation

$$(1-\alpha)y_1(1-q)^{\beta} - (1-\beta)y_2 q^{\alpha} = 0,$$

$0 < \alpha, \beta < 1$. When $\alpha = \beta$ the bilogistic model is equivalent to the logistic model with dependence parameter `dep` $= \alpha = \beta$. Complete dependence is obtained in the limit as $\alpha = \beta$ approaches zero. Independence is obtained as $\alpha = \beta$ approaches one, and when one of α, β is fixed and the other approaches one. Different limits occur when one of α, β is fixed and the other approaches zero. A bilogistic model is fitted in Smith (1990), where it appears to have been first introduced.

`model = "negbilog"` (Coles and Tawn, 1994)

The negative bilogistic distribution function with parameters `alpha` $= \alpha$ and `beta` $= \beta$ is

$$G(z_1, z_2) = \exp\left\{-y_1 - y_2 + y_1 q^{1+\alpha} + y_2(1-q)^{1+\beta}\right\}$$

where $q = q(y_1, y_2; \alpha, \beta)$ is the root of the equation

$$(1+\alpha)y_1 q^{\alpha} - (1+\beta)y_2(1-q)^{\beta} = 0,$$

$\alpha > 0$ and $\beta > 0$. When $\alpha = \beta$ the negative bilogistic model is equivalent to the negative logistic model with dependence parameter `dep` $= 1/\alpha = 1/\beta$. Complete dependence is obtained in the limit as $\alpha = \beta$ approaches zero. Independence is obtained as $\alpha = \beta$ tends to infinity, and when one of α, β is fixed and the other tends to infinity. Different limits occur when one of α, β is fixed and the other approaches zero.

`model = "ct"` (Coles and Tawn, 1991)

The Coles-Tawn distribution function with parameters `alpha` $= \alpha > 0$ and `beta` $= \beta > 0$ is

$$G(z_1, z_2) = \exp\left\{-y_1[1 - \mathrm{Be}(q; \alpha+1, \beta)] - y_2 \mathrm{Be}(q; \alpha, \beta+1)\right\}$$

where $q = \alpha y_2/(\alpha y_2 + \beta y_1)$ and $\mathrm{Be}(q; \alpha, \beta)$ is the beta distribution function evaluated at q with `shape1` $= \alpha$ and `shape2` $= \beta$. Complete dependence is obtained in the limit as $\alpha = \beta$ tends to infinity. Independence is obtained as $\alpha = \beta$ approaches zero, and when one of α, β is fixed and the other approaches zero. Different limits occur when one of α, β is fixed and the other tends to infinity.

Value

`dbvevd` gives the density function, `pbvevd` gives the distribution function and `rbvevd` generates random deviates, for one of eight parametric bivariate extreme value models.

Note

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.

The simulation of the bilogistic and negative bilogistic models requires a root finding algorithm to evaluate q within the root finding algorithm used to simulate from the conditional distributions. The generation of bilogistic and negative bilogistic random deviates is therefore relatively slow (about 2.8 seconds per 1000 random vectors on a 450MHz PIII, 512Mb RAM).

The bilogistic and negative bilogistic models can be represented under a single model, using the integral of the maximum of two beta distributions (Joe, 1997).

The Coles-Tawn model is called the Dirichelet model in Coles and Tawn (1991).

References

Coles, S. G. and Tawn, J. A. (1991) Modelling extreme multivariate events. *J. Roy. Statist. Soc., B*, **53**, 377–392.

Coles, S. G. and Tawn, J. A. (1994) Statistical methods for multivariate extremes: an application to structural design (with discussion). *Appl. Statist.*, **43**, 1–48.

Galambos, J. (1975) Order statistics of samples from multivariate distributions. *J. Amer. Statist. Assoc.*, **70**, 674–680.

Gumbel, E. J. (1960) Distributions des valeurs extremes en plusieurs dimensions. *Publ. Inst. Statist. Univ. Paris*, **9**, 171–173.

Husler, J. and Reiss, R.-D. (1989) Maxima of normal random vectors: between independence and complete dependence. *Statist. Probab. Letters*, **7**, 283–286.

Joe, H. (1990) Families of min-stable multivariate exponential and multivariate extreme value distributions. *Statist. Probab. Letters*, **9**, 75–81.

Joe, H. (1997) *Multivariate Models and Dependence Concepts*, London: Chapman & Hall.

Smith, R. L. (1990) Extreme value theory. In *Handbook of Applicable Mathematics* (ed. W. Ledermann), vol. 7. Chichester: John Wiley, pp. 437–471.

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

Tawn, J. A. (1988) Bivariate extreme value theory: models and estimation. *Biometrika*, **75**, 397–415.

See Also

`abvpar`, `rgev`, `rmvevd`

Examples

```
pbvevd(matrix(rep(0:4,2), ncol=2), dep = 0.7, model = "log")
pbvevd(c(2,2), dep = 0.7, asy = c(0.6,0.8), model = "alog")
pbvevd(c(1,1), dep = 1.7, model = "hr")

margins <- cbind(0, 1, seq(-0.5,0.5,0.1))
rbvevd(11, dep = 1.7, model = "hr", mar1 = margins)
rbvevd(10, dep = 1.2, model = "neglog", mar1 = c(10, 1, 1))
rbvevd(10, alpha = 0.7, beta = 0.52, model = "bilog")

dbvevd(c(0,0), dep = 1.2, asy = c(0.5,0.9), model = "aneglog")
dbvevd(c(0,0), alpha = 0.75, beta = 0.5, model = "ct", log = TRUE)
dbvevd(c(0,0), alpha = 0.7, beta = 1.52, model = "negbilog")
```

`chiplot` *Dependence Measure Plots*

Description

Plots of estimates of the dependence measures chi and chi-bar for bivariate data.

Usage

```
chiplot(data, nq = 100, qlim = NULL, which = 1:2, conf = 0.95, lty = 1,
    cilty = 2, col = 1, cicol = 1, xlim = c(0,1), ylim1 = NULL,
    ylim2 = c(-1,1), main1 = "Chi Plot", main2 = "Chi Bar Plot", xlab =
    "Quantile", ylab1 = "Chi", ylab2 = "Chi Bar", ask = nb.fig <
    length(which) && dev.interactive(), ...)
```

Arguments

`data`	A matrix or data frame with two columns. Rows (observations) with missing values are stripped from the data before any computations are performed.
`nq`	The number of quantiles at which the measures are evaluated.
`qlim`	The limits of the quantiles at which the measures are evaluated (see **Details**).
`which`	If only one plot is required, specify `1` for chi and `2` for chi-bar.
`conf`	The confidence coefficient of the plotted confidence intervals.
`lty, cilty`	Line types for the estimates of the measures and for the confidence intervals respectively. Use zero to supress.
`col, cicol`	Colour types for the estimates of the measures and for the confidence intervals respectively.
`xlim, xlab`	Limits and labels for the x-axis; they apply to both plots.
`ylim1`	Limits for the y-axis of the chi plot. If this is `NULL` (the default) the upper limit is one, and the lower limit is the minimum of zero and the smallest plotted value.
`ylim2`	Limits for the y-axis of the chi-bar plot.
`main1, main2`	The plot titles for the chi and chi-bar plots respectively.
`ylab1, ylab2`	The y-axis labels for the chi and chi-bar plots respectively.
`ask`	Logical; if `TRUE`, the user is asked before each plot.
`...`	Other arguments to be passed to `matplot`.

Details

These measures are explained in full detail in Coles, Heffernan and Tawn (1999). A brief treatment is also given in Section 8.4 of Coles(2001). A short summary is given as follows. We assume that the data are *iid* random vectors with common bivariate distribution function G, and we define the random vector (X, Y) to be distributed according to G.

The chi plot is a plot of q against empirical estimates of

$$\chi(q) = 2 - \log(\Pr(F_X(X) < q, F_Y(Y) < q))/\log(q)$$

where F_X and F_Y are the marginal distribution functions, and where q is in the interval (0,1). The quantity $\chi(q)$ is bounded by

$$2 - \log(2u - 1)/\log(u) \leq \chi(q) \leq 1$$

where the lower bound is interpreted as `-Inf` for $q \leq 1/2$ and zero for $q = 1$. These bounds are reflected in the corresponding estimates.

The chi bar plot is a plot of q against empirical estimates of

$$\bar{\chi}(q) = 2\log(1-q)/\log(\Pr(F_X(X) > q, F_Y(Y) > q)) - 1$$

where F_X and F_Y are the marginal distribution functions, and where q is in the interval (0,1). The quantity $\bar{\chi}(q)$ is bounded by $-1 \leq \bar{\chi}(q) \leq 1$ and these bounds are reflected in the corresponding estimates.

Note that the empirical estimators for $\chi(q)$ and $\bar{\chi}(q)$ are undefined near $q = 0$ and $q = 1$. By default the function takes the limits of q so that the plots depicts all values at which the estimators are defined. This can be overridden by the argument `qlim`, which must represent a subset of the default values (and these can be determined using the component `quantile` of the invisibly returned list; see **Value**).

The confidence intervals within the plot assume that observations are independent, and that the marginal distributions are estimated exactly. The intervals are constructed using the delta method; this may lead to poor interval estimates near $q = 0$ and $q = 1$.

The function $\chi(q)$ can be interpreted as a quantile dependent measure of dependence. In particular, the sign of $\chi(q)$ determines whether the variables are positively or negatively associated at quantile level q. By definition, variables are said to be asymptotically independent when $\chi(1)$ (defined in the limit) is zero. For independent variables, $\chi(q) = 0$ for all q in (0,1). For perfectly dependent variables, $\chi(q) = 1$ for all q in (0,1). For bivariate extreme value distributions, $\chi(q) = 2(1 - A(1/2))$ for all q in (0,1), where A is the dependence function, as defined in `abvpar`. If a bivariate threshold model is to be fitted (using `fbvpot`), this plot can therefore act as a threshold identification plot, since e.g. the use of 95% marginal quantiles as threshold values implies that $\chi(q)$ should be approximately constant above $q = 0.95$.

The function $\bar{\chi}(q)$ can again be interpreted as a quantile dependent measure of dependence; it is most useful within the class of asymptotically independent variables. For asymptotically dependent variables (i.e. those for which $\chi(1) < 1$), we have $\bar{\chi}(1) = 1$, where $\bar{\chi}(1)$ is again defined in the limit. For asymptotically independent variables, $\bar{\chi}(1)$ provides a measure that increases with dependence strength. For independent variables $\bar{\chi}(q) = 0$ for all q in (0,1), and hence $\bar{\chi}(1) = 0$.

Value

A list with components `quantile`, `chi` (if `1` is in `which`) and `chibar` (if `2` is in `which`) is invisibly returned. The components `quantile` and `chi` contain those objects that were passed to the formal arguments `x` and `y` of `matplot` in order to create the chi plot. The components `quantile` and `chibar` contain those objects that were passed to the formal arguments `x` and `y` of `matplot` in order to create the chi-bar plot.

Author(s)

Jan Heffernan and Alec Stephenson

References

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

Coles, S. G. (2001) *An Introduction to Statistical Modelling of Extreme Values*, London: Springer–Verlag.

See Also

fbvevd, fbvpot, matplot

Examples

```
par(mfrow = c(1,2))
smdat1 <- rbvevd(1000, dep = 0.6, model = "log")
smdat2 <- rbvevd(1000, dep = 1, model = "log")
chiplot(smdat1)
chiplot(smdat2)
```

---

clusters *Identify Clusters of Exceedences*

Description

Identify clusters of exceedences.

Usage

```
clusters(data, u, r = 1, ulow = -Inf, rlow = 1, cmax = FALSE, keep.names
    = TRUE, plot = FALSE, xdata = seq(along = data), lvals = TRUE, lty =
    1, lwd = 1, pch = par("pch"), col = if(n > 250) NULL else "grey",
    xlab = "Index", ylab = "Data", ...)
```

Arguments

data	A numeric vector, which may contain missing values.
u	A single value giving the threshold, unless a time varying threshold is used, in which case u should be a vector of thresholds, typically with the same length as data (or else the usual recycling rules are applied).
r	A postive integer denoting the clustering interval length. By default the interval length is one.
ulow	A single value giving the lower threshold, unless a time varying lower threshold is used, in which case ulow should be a vector of lower thresholds, typically with the same length as data (or else the usual recycling rules are applied). By default there is no lower threshold (or equivalently, the lower threshold is -Inf).
rlow	A postive integer denoting the lower clustering interval length. The lower clustering interval length is only relevant if it is less than the clustering interval length r and if there exists a lower threshold (greater than -Inf).
cmax	Logical; if FALSE (the default), a list containing the clusters of exceedences is returned. If TRUE a numeric vector containing the cluster maxima is returned.
keep.names	Logical; if FALSE, the function makes no attempt to retain the names/indices of the observations within the returned object. If data contains a large number of observations, this can make the function run much faster. The argument is mainly designed for internal use.

`plot`	Logical; if `TRUE` a plot is given that depicts the identified clusters, and the clusters (if `cmax` is `FALSE`) or cluster maxima (if `cmax` is `TRUE`) are returned invisibly. If `FALSE` (the default), the following arguments are ignored.
`xdata`	A numeric vector with the same length as `data`, giving the values to be plotted on the x-axis.
`lvals`	Logical; should the values below the threshold and the line depicting the lower threshold be plotted?
`lty, lwd`	Line type and width for the lines depicting the threshold and the lower threshold.
`pch`	Plotting character.
`col`	Strips of colour `col` are used to identify the clusters. An observation is contained in the cluster if the centre of the corresponding plotting character is contained in the coloured strip. If `col` is `NULL` the strips are omitted. By default the strips are coloured `"grey"`, but are omitted whenever `data` contains more than 250 observations.
`xlab, ylab`	Labels for the x and y axis.
`...`	Other graphics parameters.

Details

The clusters of exceedences are identified as follows. The first exceedence of the threshold initiates the first cluster. The first cluster then remains active until either `r` consecutive values fall below (or are equal to) the threshold, or until `rlow` consecutive values fall below (or are equal to) the lower threshold. The next exceedence of the threshold (if it exists) then initiates the second cluster, and so on. Missing values are allowed, in which case they are treated as falling below (or equal to) the threshold, but falling above the lower threshold.

Value

If `cmax` is `FALSE` (the default), a list with one component for each identified cluster. If `cmax` is `TRUE`, a numeric vector containing the cluster maxima. In any case, the returned object has an attribute `acs`, giving the average cluster size (where the cluster size is defined as the number of exceedences within a cluster), which will be `NaN` if there are no values above the threshold (and hence no clusters).

If `plot` is `TRUE`, the list of clusters, or vector of cluster maxima, is returned invisibly.

See Also

exi

Examples

```
data(portpirie)
clusters(portpirie, 4.2, 3)
clusters(portpirie, 4.2, 3, cmax = TRUE)
clusters(portpirie, 4.2, 3, 3.8, plot = TRUE)
clusters(portpirie, 4.2, 3, 3.8, plot = TRUE, lvals = FALSE)
tvu <- c(rep(4.2, 20), rep(4.1, 25), rep(4.2, 20))
clusters(portpirie, tvu, 3, plot = TRUE)
```

`evd-internal` *Internal Functions*

Description

The evd package contains many internal functions that are not designed to be called by the user.

The generic functions `dens`, `pp`, `qq` and `rl` create the diagnostic plots generated by `plot.uvevd`. Similarly, `bvdens`, `bvcpp` and `bvdp` create the diagnostic plots generated by `plot.bvevd`.

There are internal fitting, simulation, distribution and density functions for each bivariate and multivariate parametric model, which are called from functions such as `rbvevd` and `rmvevd`. There also exists internal functions for the calculation and plotting of dependence functions of bivariate and trivariate models, which are called from `abvdep` and `atvdep`. The dependence functions are ultimately plotted by the low-level functions `bvdepfn` and `tvdepfn`.

The function `pcint` calculates profile confidence intervals, and is called from the function `plot.profile.evd`. The fitting function `fgev` calls the internal functions `fgev.quantile` and `fgev.norm` for fits under different parameterizations. The fitting function `fpot` calls the internal functions `fpot.norm` and `fpot.quantile`. Marginal transformations are executed using `mtransform`.

The function `ccop` calculates condition copulas (i.e. conditional distributions under uniform margins) for each bivariate parametric model, and `ccop.case` does the same for when a case indicator is implemented, conditioning also on the case. They are needed to create the conditional P-P plots generated by `bvcpp`.

The functions `nsloc.transform`, `na.vals`, `bvpost.optim`, `bvstart.vals` and `sep.bvdata` are used in the fitting of bivariate models. The function `mvalog.check` checks and transforms the `asy` argument for the multivariate asymmetric model. The function `subsets` lists all subsets of `1:n`; it is called by `mvalog.check` and multivariate distribution functions.

For fitting bivariate threshold models, internal functions exist for the censored and (currently unimplemented) point process likelihoods, and each of these calls a further internal function corresponding to the specified model. The internal function `bvtpost.optim` is then used for post optimization processing.

`evmc` *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 eight parametric bivariate extreme value distributions.

Usage

```
evmc(n, dep, asy = c(1,1), alpha, beta, model = "log",
    margins = "uniform")
```

Arguments

`n`	Number of observations.
`dep`	Dependence parameter for the logistic, asymmetric logistic, Husler-Reiss, negative logistic and asymmetric negative logistic models.
`asy`	A vector of length two, containing the two asymmetry parameters for the asymmetric logistic and asymmetric negative logistic models.
`alpha, beta`	Alpha and beta parameters for the bilogistic, negative bilogistic and Coles-Tawn models.
`model`	The specified model; a character string. Must be either `"log"` (the default), `"alog"`, `"hr"`, `"neglog"`, `"aneglog"`, `"bilog"`, `"negbilog"` or `"ct"` (or any unique partial match), for the logistic, asymmetric logistic, Husler-Reiss, negative logistic, asymmetric negative logistic, bilogistic, negative bilogistic and Coles-Tawn models respectively. The definition of each model is given in `rbvevd`. If parameter arguments are given that do not correspond to the specified model those arguments are ignored, with a warning.
`margins`	The marginal distribution of each value; a character string. Must be either `"uniform"` (the default), `"exponential"`, `"frechet"` or `"gumbel"` (or any unique partial match), for the uniform, standard exponential, standard Gumbel and standard Frechet distributions respectively.

Value

A numeric vector of length `n`.

See Also

`marma`, `rbvevd`

Examples

```
evmc(100, alpha = 0.1, beta = 0.1, model = "bilog")
evmc(100, dep = 10, model = "hr", margins = "exp")
```

---

exi *Estimates of the Extremal Index*

Description

Estimates of the extremal index.

Usage

```
exi(data, u, r = 1, ulow = rep(-Inf, ncol(u)), rlow =
    rep(1, length(r)), dimnames = list(NULL, NULL), drop = TRUE)
```

Arguments

data	A numeric vector, which may contain missing values.
u	A numeric vector of thresholds, unless time varying thresholds are used, in which case u should be a matrix, typically with length(data) rows (or else the usual recycling rules are applied).
r	A numeric vector of postive integers denoting the clustering interval lengths.
ulow	A numeric vector of lower thresholds with length length(u) (if u is a vector) or ncol(u) (if u is a matrix), unless time varying lower thresholds are used, in which case ulow should be a matrix with length(u) or ncol(u) columns, and typically with length(data) rows (or else the usual recycling rules are applied). By default there are no lower thresholds (or equivalently, the lower thresholds are -Inf).
rlow	A numeric vector of postive integers, of length length(r), denoting the lower clustering interval lengths. The lower clustering interval length is only relevant if it is less than the clustering interval length and if there exists a lower threshold (greater than -Inf).
dimnames	The dimnames of the result.
drop	Logical; return a vector if either u or r has one element/column?

Details

The extremal index is estimated using the inverse of the average cluster size, using the clusters of exceedences derived from clusters. If the threshold is larger than (or equal to) max(data), so that no clusters are derived, then the corresponding estimate is NaN.

Value

A matrix with length(u) (if u is a vector) or ncol(u) (if u is a matrix) rows and length(r) columns, such that the ijth element is the estimate of the extremal index using the threshold u[i] or u[,i], the clustering interval length r[j], the lower threshold ulow[i] or ulow[,i], and the lower clustering interval length rlow[j]. If drop is TRUE (the default), a numeric vector is returned in preference to a matrix with one row/column.

See Also

clusters

Examples

```
data(portpirie)
exi(portpirie, 4.2, 3, 3.8)
us <- seq(3.9, 4.2, len = 10)
exi(portpirie, us, 3)
exi(portpirie, us, 1:3)
tvu <- c(rep(4.2, 20), rep(4.1, 25), rep(4.2, 20))
exi(portpirie, as.matrix(tvu), 1:3)

us <- seq(3.9, 4.2, len = 100)
eis <- exi(portpirie, us, 1:3)
matplot(us, eis, type = "l", xlab = "Threshold", ylab = "Ext. Index")
```

`extreme` *Distributions of Maxima and Minima*

Description

Density function, distribution function, quantile function and random generation for the maximum/minimum of a given number of independent variables from a specified distribution.

Usage

```
dextreme(x, densfun, distnfun, ..., distn, mlen = 1, largest = TRUE,
    log = FALSE)
pextreme(q, distnfun, ..., distn, mlen = 1, largest = TRUE,
    lower.tail = TRUE)
qextreme(p, quantfun, ..., distn, mlen = 1, largest = TRUE,
    lower.tail = TRUE)
rextreme(n, quantfun, ..., distn, mlen = 1, largest = TRUE)
```

Arguments

`x, q`	Vector of quantiles.
`p`	Vector of probabilities.
`n`	Number of observations.
`densfun, distnfun, quantfun`	Density, distribution and quantile function of the specified distribution. The density function must have a `log` argument (a simple wrapper can always be constructed to achieve this).
`...`	Parameters of the specified distribution.
`distn`	A character string, optionally given as an alternative to `densfun`, `distnfun` and `quantfun` such that the density, distribution and quantile functions are formed upon the addition of the prefixes `d`, `p` and `q` respectively.
`mlen`	The number of independent variables.
`largest`	Logical; if `TRUE` (default) use maxima, otherwise minima.
`log`	Logical; if `TRUE`, the log density is returned.
`lower.tail`	Logical; if `TRUE` (default) probabilities are P[X <= x], otherwise P[X > x].

Value

`dextreme` gives the density function, `pextreme` gives the distribution function and `qextreme` gives the quantile function of the maximum/minimum of `mlen` independent variables from a specified distibution. `rextreme` generates random deviates.

See Also

`rgev`, `rorder`

Examples

```
dextreme(2:4, dnorm, pnorm, mean = 0.5, sd = 1.2, mlen = 5)
dextreme(2:4, distn = "norm", mean = 0.5, sd = 1.2, mlen = 5)
dextreme(2:4, distn = "exp", mlen = 2, largest = FALSE)
pextreme(2:4, distn = "exp", rate = 1.2, mlen = 2)
qextreme(seq(0.9, 0.6, -0.1), distn = "exp", rate = 1.2, mlen = 2)
rextreme(5, qgamma, shape = 1, mlen = 10)
p <- (1:9)/10
pexp(qextreme(p, distn = "exp", rate = 1.2, mlen = 1), rate = 1.2)
## [1] 0.1 0.2 0.3 0.4 0.5 0.6 0.7 0.8 0.9
```

---

failure *Failure Times*

Description

Failure times.

Usage

```
data(failure)
```

Format

A vector containing 24 observations.

Source

van Montfort, M. A. J. and Otten, A. (1978) On testing a shape parameter in the presence of a scale parameter. *Math. Operations Forsch. Statist., Ser. Statistics*, **9**, 91–104.

---

fbvevd *Maximum-likelihood Fitting of Bivariate Extreme Value Distributions*

Description

Fit models for one of eight parametric bivariate extreme value distributions, including linear modelling of the marginal location parameters, and allowing any of the parameters to be held fixed if desired.

Usage

```
fbvevd(x, model = "log", start, ..., sym = FALSE, nsloc1 = NULL,
    nsloc2 = NULL, cshape = cscale, cscale = cloc, cloc = FALSE,
    std.err = TRUE, dsm = TRUE, corr = FALSE, method = "BFGS",
    warn.inf = TRUE)
```

Arguments

`x`	A matrix or data frame, ordinarily with two columns, which may contain missing values. A data frame may also contain a third column of mode `logical`, which itself may contain missing values (see **More Details**).
`model`	The specified model; a character string. Must be either `"log"` (the default), `"alog"`, `"hr"`, `"neglog"`, `"aneglog"`, `"bilog"`, `"negbilog"` or `"ct"` (or any unique partial match), for the logistic, asymmetric logistic, Husler-Reiss, negative logistic, asymmetric negative logistic, bilogistic, negative bilogistic and Coles-Tawn models respectively. The definition of each model is given in `rbvevd`.
`start`	A named list giving the initial values for the parameters over which the likelihood is to be maximized. If `start` is omitted the routine attempts to find good starting values using marginal maximum likelihood estimators.
`...`	Additional parameters, either for the bivariate extreme value model or for the optimization function `optim`. If parameters of the model are included they will be held fixed at the values given (see **Examples**).
`sym`	Logical; if `TRUE`, the dependence structure of the models `"alog"`, `"aneglog"` or `"ct"` are constrained to be symmetric (see **Details**). For all other models, the argument is ignored (and a warning is given).
`nsloc1, nsloc2`	A data frame with the same number of rows as `x`, for linear modelling of the location parameter on the first/second margin (see **Details**). The data frames are treated as covariate matrices, excluding the intercept. A numeric vector can be given as an alternative to a single column data frame.
`cshape`	Logical; if `TRUE`, a common shape parameter is fitted to each margin.
`cscale`	Logical; if `TRUE`, a common scale parameter is fitted to each margin, and the default value of `cshape` is then `TRUE`, so that under this default common scale and shape parameters are fitted.
`cloc`	Logical; if `TRUE`, a common location parameter is fitted to each margin, and the default values of `cshape` and `cscale` are then `TRUE`, so that under these defaults common marginal parameters are fitted.
`std.err`	Logical; if `TRUE` (the default), the standard errors are returned.
`dsm`	Logical; if `TRUE` (the default), summaries of the dependence structure are returned.
`corr`	Logical; if `TRUE`, the correlation matrix is returned.
`method`	The optimization method (see `optim` for details).
`warn.inf`	Logical; if `TRUE` (the default), a warning is given if the negative log-likelihood is infinite when evaluated at the starting values.

Details

The dependence parameter names are one or more of `dep`, `asy1`, `asy2`, `alpha` and `beta`, depending on the model selected (see `rbvevd`). The marginal parameter names are `loc1`, `scale1` and `shape1` for the first margin, and `loc2`, `scale2` and `shape2` for the second margin. If `nsloc1` is not `NULL`, so that a linear model is implemented for the first marginal location parameter, the parameter names for the first margin are `loc1`, `loc1`*x1*, ..., `loc1`*xn*, `scale` and `shape`, where *x1*, ..., *xn* are the column names of `nsloc1`, so that `loc1` is the

intercept of the linear model, and `loc1`*x1*, ..., `loc1`*xn* are the `ncol(nsloc1)` coefficients. When `nsloc2` is not `NULL`, the parameter names for the second margin are constructed similarly.

It is recommended that the covariates within the linear models for the location parameters are (at least approximately) centered and scaled (i.e. that the columns of `nsloc1` and `nsloc2` are centered and scaled), particularly if automatic starting values are used, since the starting values for the associated parameters are then zero. If `cloc` is `TRUE`, both `nsloc1` and `nsloc2` must be identical, since a common linear model is then implemented on both margins.

If `cshape` is true, the models are constrained so that `shape2 = shape1`. The parameter `shape2` is then taken to be specified, so that e.g. the common shape parameter can only be fixed at zero using `shape1 = 0`, since using `shape2 = 0` gives an error. Similar comments apply for `cscale` and `cloc`.

If `sym` is `TRUE`, the asymmetric logistic and asymmetric negative logistic models are constrained so that `asy2 = asy1`, and the Coles-Tawn model is constrained so that `beta = alpha`. The parameter `asy2` or `beta` is then taken to be specified, so that e.g. the parameters `asy1` and `asy2` can only be fixed at `0.8` using `asy1 = 0.8`, since using `asy2 = 0.8` gives an error.

Bilogistic and negative bilogistic models constrained to symmetry are logistic and negative logistic models respectively. The mixed model (e.g. Tawn, 1998) is obtained by the asymmetric negative logistic model upon setting the dependence parameter to be one, and constraining the asymmetry parameters to be equal to each other. It can therefore be fitted using `model = "anegl"` with `dep = 1` and `sym = TRUE` (see **Examples**).

If `dsm` is `TRUE`, three values are returned which summarize the dependence structure, based on the fitted dependence function A (see `abvpar`). Two are measures of the strength of dependence. The first (Dependence One) is given by $2(1-A(1/2))$. The second (Dependence Two) is the integral of $4(1 - A(x))$, taken over $0 \leq x \leq 1$. Both measures are zero at independence and one at complete dependence.

The third value (Asymmetry) is a measure of asymmetry, given by the integral of $4(A(x) - A(1-x))/(3-2\sqrt{2})$, taken over $0 \leq x \leq 0.5$. This lies in the closed interval [-1,1], with larger absolute values representing stronger asymmetry. For the logistic, Husler-Reiss and negative logistic models $A(x) = A(1-x)$ for all $0 \leq x \leq 0.5$, so the value will be zero.

For numerical reasons the parameters of each model are subject the artificial constraints given in Table 1 of the User's Guide.

Value

Returns an object of class `c("bvevd","evd")`.

The generic accessor functions `fitted` (or `fitted.values`), `std.errors`, `deviance`, `logLik` and `AIC` extract various features of the returned object.

The functions `profile` and `profile2d` can be used to obtain deviance profiles. The function `anova` compares nested models, and the function `AIC` compares non-nested models. The function `plot` produces diagnostic plots.

An object of class `c("bvevd","evd")` is a list containing the following components

`estimate`	A vector containing the maximum likelihood estimates.
`std.err`	A vector containing the standard errors.
`fixed`	A vector containing the parameters that have been fixed at specific values within the optimization.

`fixed2` A vector containing the parameters that have been set to be equal to other model parameters.

`param` A vector containing all parameters (those optimized, those fixed to specific values, and those set to be equal to other model parameters).

`deviance` The deviance at the maximum likelihood estimates.

`dep.summary` A vector of three values, summarizing the dependence structure of the fitted model (see **Details**).

`corr` The correlation matrix.

`convergence, counts, message`
Components taken from the list returned by `optim`.

`data` The data passed to the argument `x`.

`tdata` The data, transformed to stationarity (for non-stationary models).

`nsloc1, nsloc2`
The arguments `nsloc1` and `nsloc2`.

`n` The number of rows in `x`.

`sym` The argument `sym`.

`cmar` The vector `c(cloc, cscale, cshape)`.

`model` The argument `model`.

`call` The call of the current function.

More Details

If `x` is a data frame with a third column of mode `logical`, then the model is fitted using the likelihood derived by Stephenson and Tawn (2004). This is appropriate when each bivariate data point comprises componentwise maxima from some underlying bivariate process, and where the corresponding logical value denotes whether or not the maxima were caused by the same event within that process.

Under this scheme the diagnostic plots that are produced using `plot` are somewhat different to those described in `plot.bvevd`. In particular, there is no comparative non-parametric dependence function estimate, and the conditional P-P plots condition on both the logical case and the given margin (which requires numerical integration at each data point).

Warning

The standard errors and the correlation matrix in the returned object are taken from the observed information, calculated by a numerical approximation. They must be interpreted with caution when either of the marginal shape parameters are less than -0.5, because the usual asymptotic properties of maximum likelihood estimators do not then hold (Smith, 1985).

References

Smith, R. L. (1985) Maximum likelihood estimation in a class of non-regular cases. *Biometrika*, **72**, 67–90.

Stephenson, A. G. and Tawn, J. A. (2004) Exploiting Occurence Times in Likelihood Inference for Componentwise Maxima. *Biometrika* (To Appear).

Tawn, J. A. (1988) Bivariate extreme value theory: models and estimation. *Biometrika*, **75**, 397–415.

See Also

anova.evd, optim, plot.bvevd, profile.evd, profile2d.evd, rbvevd

Examples

```
bvdata <- rbvevd(100, dep = 0.6, model = "log", mar1 = c(1.2,1.4,0.4))
M1 <- fbvevd(bvdata, model = "log")
M2 <- fbvevd(bvdata, model = "log", dep = 0.75)
anova(M1, M2)
par(mfrow = c(2,2))
plot(M1)
plot(M1, mar = 1)
plot(M1, mar = 2)
## Not run: par(mfrow = c(1,1))
## Not run: M1P <- profile(M1, which = "dep")
## Not run: plot(M1P)

trend <- (-49:50)/100
rnd <- runif(100, min = -.5, max = .5)
fbvevd(bvdata, model = "log", nsloc1 = trend)
fbvevd(bvdata, model = "log", nsloc1 = trend, nsloc2 = data.frame(trend
= trend,  random = rnd))
fbvevd(bvdata, model = "log", nsloc1 = trend, nsloc2 = data.frame(trend
= trend, random = rnd), loc2random = 0)

bvdata <- rbvevd(100, dep = 1, asy = c(0.5,0.5), model = "anegl")
anlog <- fbvevd(bvdata, model = "anegl")
mixed <- fbvevd(bvdata, model = "anegl", dep = 1, sym = TRUE)
anova(anlog, mixed)
```

---

fbvpot — *Maximum-likelihood Fitting of Bivariate Extreme Value Distributions to Threshold Exceedances*

Description

Fit models for one of seven parametric bivariate extreme-value distributions using threshold exceedances, allowing any of the parameters to be held fixed if desired.

Usage

```
fbvpot(x, threshold, model = "log", likelihood = "censored", start,
    ..., sym = FALSE, cshape = cscale, cscale = FALSE, std.err =
    TRUE, dsm = TRUE, corr = FALSE, method = "BFGS", warn.inf = TRUE)
```

Arguments

x	A matrix or data frame with two columns. If this contains missing values, those values are treated as if they fell below the corresponding marginal threshold.
threshold	A vector of two thresholds.

`model`	The specified model; a character string. Must be either `"log"` (the default), `"alog"`, `"hr"`, `"neglog"`, `"aneglog"`, `"bilog"`, `"negbilog"` or `"ct"` (or any unique partial match), for the logistic, asymmetric logistic, Husler-Reiss, negative logistic, asymmetric negative logistic, bilogistic, negative bilogistic and Coles-Tawn models respectively. The definition of each model is given in `rbvevd`.
`likelihood`	Unimplemented.
`start`	A named list giving the initial values for all of the parameters in the model. If `start` is omitted the routine attempts to find good starting values using marginal maximum likelihood estimators.
`...`	Additional parameters, either for the bivariate extreme value model or for the optimization function `optim`. If parameters of the model are included they will be held fixed at the values given (see **Examples**).
`sym`	Logical; if `TRUE`, the dependence structure of the models `"alog"`, `"aneglog"` or `"ct"` are constrained to be symmetric (see **Details**). For all other models, the argument is ignored (and a warning is given).
`cshape`	Logical; if `TRUE`, a common shape parameter is fitted to each margin.
`cscale`	Logical; if `TRUE`, a common scale parameter is fitted to each margin, and the default value of `cshape` is then `TRUE`, so that under this default common marginal parameters are fitted.
`std.err`	Logical; if `TRUE` (the default), the standard errors are returned.
`dsm`	Logical; if `TRUE` (the default), summaries of the dependence structure are returned.
`corr`	Logical; if `TRUE`, the correlation matrix is returned.
`method`	The optimization method (see `optim` for details).
`warn.inf`	Logical; if `TRUE` (the default), a warning is given if the negative log-likelihood is infinite when evaluated at the starting values.

Details

The bivariate peaks over threshold models are fitted by maximizing the censored likelihood as given in e.g. Section 8.3.1 of Coles(2001).

The dependence parameter names are one or more of `dep`, `asy1`, `asy2`, `alpha` and `beta`, depending on the model selected (see `rbvevd`). The marginal parameter names are `scale1` and `shape1` for the first margin, and `scale2` and `shape2` for the second margin.

If `cshape` is true, the models are constrained so that `shape2 = shape1`. The parameter `shape2` is then taken to be specified, so that e.g. the common shape parameter can only be fixed at zero using `shape1 = 0`, since using `shape2 = 0` gives an error. Similar comments apply for `cscale`.

If `sym` is `TRUE`, the asymmetric logistic and asymmetric negative logistic models are constrained so that `asy2 = asy1`, and the Coles-Tawn model is constrained so that `beta = alpha`. The parameter `asy2` or `beta` is then taken to be specified, so that e.g. the parameters `asy1` and `asy2` can only be fixed at `0.8` using `asy1 = 0.8`, since using `asy2 = 0.8` gives an error.

Bilogistic and negative bilogistic models constrained to symmetry are logistic and negative logistic models respectively. The mixed model (e.g. Tawn, 1998) is obtained by the asymmetric negative logistic model upon setting the dependence parameter to be one, and

constraining the asymmetry parameters to be equal to each other. It can therefore be fitted using `model = "anegl"` with `dep = 1` and `sym = TRUE`.

If `dsm` is `TRUE`, three values are returned which summarize the dependence structure, based on the fitted dependence function A (see `fbvevd` for details).

For numerical reasons the parameters of each model are subject the artificial constraints given in Table 1 of the User's Guide.

Value

Returns an object of class `c("bvpot","evd")`.

The generic accessor functions `fitted` (or `fitted.values`), `std.errors`, `deviance`, `logLik` and `AIC` extract various features of the returned object.

The functions `profile` and `profile2d` can be used to obtain deviance profiles. The function `anova` compares nested models, and the function `AIC` compares non-nested models. There is currently no plot method available.

An object of class `c("bvpot","evd")` is a list containing the following components

`estimate`	A vector containing the maximum likelihood estimates.
`std.err`	A vector containing the standard errors.
`fixed`	A vector containing the parameters that have been fixed at specific values within the optimization.
`fixed2`	A vector containing the parameters that have been set to be equal to other model parameters.
`param`	A vector containing all parameters (those optimized, those fixed to specific values, and those set to be equal to other model parameters).
`deviance`	The deviance at the maximum likelihood estimates.
`dep.summary`	A vector of three values, summarizing the dependence structure of the fitted model (see **Details**).
`corr`	The correlation matrix.
`convergence, counts, message`	Components taken from the list returned by `optim`.
`data`	The data passed to the argument `x`.
`threshold`	The argument `threshold`.
`n`	The number of rows in `x`.
`nat`	The vector of length three containing the number of exceedances on the first, second and both margins respectively.
`sym`	The argument `sym`.
`cmar`	The vector `c(cscale, cshape)`.
`model`	The argument `model`.
`call`	The call of the current function.

Warning

The standard errors and the correlation matrix in the returned object are taken from the observed information, calculated by a numerical approximation. They must be interpreted with caution when either of the marginal shape parameters are less than -0.5, because the usual asymptotic properties of maximum likelihood estimators do not then hold (Smith, 1985).

Author(s)

Chris Ferro and Alec Stephenson

References

Coles, S. G. (2001) *An Introduction to Statistical Modelling of Extreme Values*, London: Springer–Verlag.

Smith, R. L. (1985) Maximum likelihood estimation in a class of non-regular cases. *Biometrika*, **72**, 67–90.

See Also

`abvpar`, `anova.evd`, `fbvevd`, `optim`, `rbvevd`

Examples

```
bvdata <- rbvevd(1000, dep = 0.5, model = "log")
u <- apply(bvdata, 2,  quantile, probs = 0.9)
M1 <- fbvpot(bvdata, u, model = "log")
M2 <- fbvpot(bvdata, u, "log", dep = 0.5)
anova(M1, M2)
```

`fextreme` *Maximum-likelihood Fitting of Maxima and Minima*

Description

Maximum-likelihood fitting for the distribution of the maximum/minimum of a given number of independent variables from a specified distribution.

Usage

```
fextreme(x, start, densfun, distnfun, ..., distn, mlen = 1, largest =
    TRUE, std.err = TRUE, corr = FALSE, method = "Nelder-Mead")
```

Arguments

`x`	A numeric vector.
`start`	A named list giving the initial values for the parameters over which the likelihood is to be maximized.
`densfun, distnfun`	Density and distribution function of the specified distribution.
`...`	Additional parameters, either for the specified distribution or for the optimization function `optim`. If parameters of the distribution are included they will be held fixed at the values given (see **Examples**). If parameters of the distribution are not included either here or as a named component in `start` they will be held fixed at the default values specified in the corresponding density and distribution functions (assuming they exist; an error will be generated otherwise).

`distn`	A character string, optionally specified as an alternative to `densfun` and `distnfun` such that the density and distribution functions are formed upon the addition of the prefixes `d` and `p` respectively.
`mlen`	The number of independent variables.
`largest`	Logical; if `TRUE` (default) use maxima, otherwise minima.
`std.err`	Logical; if `TRUE` (the default), the standard errors are returned.
`corr`	Logical; if `TRUE`, the correlation matrix is returned.
`method`	The optimization method (see `optim` for details).

Details

Maximization of the log-likelihood is performed. The estimated standard errors are taken from the observed information, calculated by a numerical approximation.

If the density and distribution functions are user defined, the order of the arguments must mimic those in R base (i.e. data first, parameters second). Density functions must have `log` arguments.

Value

Returns an object of class `c("extreme","evd")`.

The generic accessor functions `fitted` (or `fitted.values`), `std.errors`, `deviance`, `logLik` and `AIC` extract various features of the returned object. The function `anova` compares nested models.

An object of class `c("extreme","evd")` is a list containing at most the following components

`estimate`	A vector containing the maximum likelihood estimates.
`std.err`	A vector containing the standard errors.
`deviance`	The deviance at the maximum likelihood estimates.
`corr`	The correlation matrix.
`convergence, counts, message`	Components taken from the list returned by `optim`.
`call`	The call of the current function.
`data`	The data passed to the argument `x`.
`n`	The length of `x`.

See Also

`anova.evd`, `forder`, `optim`

Examples

```
uvdata <- rextreme(100, qnorm, mean = 0.56, mlen = 365)
fextreme(uvdata, list(mean = 0, sd = 1), distn = "norm", mlen = 365)
fextreme(uvdata, list(rate = 1), distn = "exp", mlen = 365)
fextreme(uvdata, list(scale = 1), shape = 1, distn = "gamma", mlen = 365)
fextreme(uvdata, list(shape = 1, scale = 1), distn = "gamma", mlen = 365)
```

`fgev` *Maximum-likelihood Fitting of the Generalized Extreme Value Distribution*

Description

Maximum-likelihood fitting for the generalized extreme value distribution, including linear modelling of the location parameter, and allowing any of the parameters to be held fixed if desired.

Usage

```
fgev(x, start, ..., nsloc = NULL, prob = NULL, std.err = TRUE,
    corr = FALSE, method = "BFGS", warn.inf = TRUE)
```

Arguments

`x`	A numeric vector, which may contain missing values.
`start`	A named list giving the initial values for the parameters over which the likelihood is to be maximized. If `start` is omitted the routine attempts to find good starting values using moment estimators.
`...`	Additional parameters, either for the GEV model or for the optimization function `optim`. If parameters of the model are included they will be held fixed at the values given (see **Examples**).
`nsloc`	A data frame with the same number of rows as the length of `x`, for linear modelling of the location parameter. The data frame is treated as a covariate matrix (excluding the intercept). A numeric vector can be given as an alternative to a single column data frame.
`prob`	Controls the parameterization of the model (see **Details**). Should be either `NULL` (the default), or a probability in the closed interval [0,1].
`std.err`	Logical; if `TRUE` (the default), the standard errors are returned.
`corr`	Logical; if `TRUE`, the correlation matrix is returned.
`method`	The optimization method (see `optim` for details).
`warn.inf`	Logical; if `TRUE` (the default), a warning is given if the negative log-likelihood is infinite when evaluated at the starting values.

Details

If `prob` is `NULL` (the default):

For stationary models the parameter names are `loc`, `scale` and `shape`, for the location, scale and shape parameters respectively. For non-stationary models, the parameter names are `loc`, `loc`*x1*, ..., `loc`*xn*, `scale` and `shape`, where *x1*, ..., *xn* are the column names of `nsloc`, so that `loc` is the intercept of the linear model, and `loc`*x1*, ..., `loc`*xn* are the `ncol(nsloc)` coefficients. If `nsloc` is a vector it is converted into a single column data frame with column name `trend`, and hence the associated trend parameter is named `loctrend`.

If `prob` $= p$ is a probability:

The fit is performed using a different parameterization. Let a, b and s denote the location, scale and shape parameters of the GEV distribution. For stationary models, the distribution is parameterized using (z_p, b, s), where

$$z_p = a - b/s(1 - (-\log(1-p))^s)$$

is such that $G(z_p) = 1-p$, where G is the GEV distribution function. `prob` $= p$ is therefore the probability in the upper tail corresponding to the quantile z_p. If `prob` is zero, then z_p is the upper end point $a - b/s$, and s is restricted to the negative (Weibull) axis. If `prob` is one, then z_p is the lower end point $a - b/s$, and s is restricted to the positive (Frechet) axis. The parameter names are `quantile`, `scale` and `shape`, for z_p, b and s respectively.

For non-stationary models the parameter z_p is again given by the equation above, but a becomes the intercept of the linear model for the location parameter, so that `quantile` replaces (the intercept) `loc`, and hence the parameter names are `quantile`, `loc`*x1*, ..., `loc`*xn*, `scale` and `shape`, where *x1*, ..., *xn* are the column names of `nsloc`.

In either case:

For non-stationary fitting it is recommended that the covariates within the linear model for the location parameter are (at least approximately) centered and scaled (i.e. that the columns of `nsloc` are centered and scaled), particularly if automatic starting values are used, since the starting values for the associated parameters are then zero.

Value

Returns an object of class `c("gev","uvevd","evd")`.

The generic accessor functions `fitted` (or `fitted.values`), `std.errors`, `deviance`, `logLik` and `AIC` extract various features of the returned object.

The functions `profile` and `profile2d` are used to obtain deviance profiles for the model parameters. In particular, profiles of the quantile z_p can be calculated and plotted when `prob` $= p$. The function `anova` compares nested models. The function `plot` produces diagnostic plots.

An object of class `c("gev","uvevd","evd")` is a list containing at most the following components

`estimate`	A vector containing the maximum likelihood estimates.
`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`.
`data`	The data passed to the argument `x`.
`tdata`	The data, transformed to stationarity (for non-stationary models).
`nsloc`	The argument `nsloc`.
`n`	The length of `x`.
`prob`	The argument `prob`.
`loc`	The location parameter. If `prob` is `NULL` (the default), this will also be an element of `param`.
`call`	The call of the current function.

Warning

The standard errors and the correlation matrix in the returned object are taken from the observed information, calculated by a numerical approximation. They must be interpreted with caution when the shape parameter is less than -0.5, because the usual asymptotic properties of maximum likelihood estimators do not then hold (Smith, 1985).

References

Smith, R. L. (1985) Maximum likelihood estimation in a class of non-regular cases. *Biometrika*, **72**, 67–90.

See Also

anova.evd, optim, plot.uvevd, profile.evd, profile2d.evd

Examples

```
uvdata <- rgev(100, loc = 0.13, scale = 1.1, shape = 0.2)
trend <- (-49:50)/100
M1 <- fgev(uvdata, nsloc = trend, control = list(trace = 1))
M2 <- fgev(uvdata)
M3 <- fgev(uvdata, shape = 0)
M4 <- fgev(uvdata, scale = 1, shape = 0)
anova(M1, M2, M3, M4)
par(mfrow = c(2,2))
plot(M2)
## Not run: M2P <- profile(M2)
## Not run: plot(M2P)

rnd <- runif(100, min = -.5, max = .5)
fgev(uvdata, nsloc = data.frame(trend = trend, random = rnd))
fgev(uvdata, nsloc = data.frame(trend = trend, random = rnd), locrandom = 0)

uvdata <- rgev(100, loc = 0.13, scale = 1.1, shape = 0.2)
M1 <- fgev(uvdata, prob = 0.1)
M2 <- fgev(uvdata, prob = 0.01)
## Not run: M1P <- profile(M1, which = "quantile")
## Not run: M2P <- profile(M2, which = "quantile")
## Not run: plot(M1P)
## Not run: plot(M2P)
```

---

forder *Maximum-likelihood Fitting of Order Statistics*

Description

Maximum-likelihood fitting for the distribution of a selected order statistic of a given number of independent variables from a specified distribution.

Usage

```
forder(x, start, densfun, distnfun, ..., distn, mlen = 1, j = 1,
largest = TRUE, std.err = TRUE, corr = FALSE, method = "Nelder-Mead")
```

Arguments

`x`	A numeric vector.
`start`	A named list giving the initial values for the parameters over which the likelihood is to be maximized.
`densfun, distnfun`	Density and distribution function of the specified distribution.
`...`	Additional parameters, either for the specified distribution or for the optimization function `optim`. If parameters of the distribution are included they will be held fixed at the values given (see **Examples**). If parameters of the distribution are not included either here or as a named component in `start` they will be held fixed at the default values specified in the corresponding density and distribution functions (assuming they exist; an error will be generated otherwise).
`distn`	A character string, optionally specified as an alternative to `densfun` and `distnfun` such that the density and distribution and functions are formed upon the addition of the prefixes `d` and `p` respectively.
`mlen`	The number of independent variables.
`j`	The order statistic, taken as the `j`th largest (default) or smallest of `mlen`, according to the value of `largest`.
`largest`	Logical; if `TRUE` (default) use the `j`th largest order statistic, otherwise use the `j`th smallest.
`std.err`	Logical; if `TRUE` (the default), the standard errors are returned.
`corr`	Logical; if `TRUE`, the correlation matrix is returned.
`method`	The optimization method (see `optim` for details).

Details

Maximization of the log-likelihood is performed. The estimated standard errors are taken from the observed information, calculated by a numerical approximation.

If the density and distribution functions are user defined, the order of the arguments must mimic those in R base (i.e. data first, parameters second). Density functions must have `log` arguments.

Value

Returns an object of class `c("extreme","evd")`. This class is defined in `fextreme`.

The generic accessor functions `fitted` (or `fitted.values`), `std.errors`, `deviance`, `logLik` and `AIC` extract various features of the returned object. The function `anova` compares nested models.

See Also

`anova.evd`, `fextreme`, `optim`

Examples

```
uvd <- rorder(100, qnorm, mean = 0.56, mlen = 365, j = 2)
forder(uvd, list(mean = 0, sd = 1), distn = "norm", mlen = 365, j = 2)
forder(uvd, list(rate = 1), distn = "exp", mlen = 365, j = 2)
forder(uvd, list(scale = 1), shape = 1, distn = "gamma", mlen = 365, j = 2)
forder(uvd, list(shape = 1, scale = 1), distn = "gamma", mlen = 365, j = 2)
```

fox — *Maximum Annual Flood Discharges of the Fox River*

Description

The `fox` data frame has 33 rows and 2 columns. The columns contain maximum annual flood discharges, in units of 1000 cubed feet per second, from the Fox River in Wisconsin, USA at Berlin (upstream) and Wrightstown (downstream), for the years 1918 to 1950. The row names give the years of observation.

Usage

```
data(fox)
```

Format

This data frame contains the following columns:

berlin A numeric vector containing maximum annual flood discharges at Berlin (upstream).

wright A numeric vector containing maximum annual flood discharges at Wrightstown (downstream).

Source

Gumbel, E. J. and Mustafi, C. K. (1967) Some analytical properties of bivariate extremal distributions. *J. Amer. Statist. Assoc.*, **62**, 569–588.

fpot — *Peaks Over Threshold Modelling using the Generalized Pareto or Point Process Representation*

Description

Maximum-likelihood fitting for peaks over threshold modelling, using the Generalized Pareto or Point Process representation, allowing any of the parameters to be held fixed if desired.

Usage

```
fpot(x, threshold, model = c("gpd", "pp"), start, npp = length(x),
    cmax = FALSE, r = 1, ulow = -Inf, rlow = 1, mper = NULL, ...,
    std.err = TRUE, corr = FALSE, method = "BFGS", warn.inf = TRUE)
```

Arguments

`x`	A numeric vector. If this contains missing values, those values are treated as if they fell below the threshold.
`threshold`	The threshold.
`model`	The model; either `"gpd"` (the default) or `"pp"`, for the Generalized Pareto or Point Process representations respectively.

`start`	A named list giving the initial values for the parameters over which the likelihood is to be maximized. If `start` is omitted the routine attempts to find good starting values using moment estimators.
`npp`	The data should contain `npp` observations per "period", where the return level plot produced by `plot.pot` will represent return periods in units of "periods". By default `npp = length(x)`, so that the "period" is the period of time over which the entire data set is collected. It may often be useful to change this default so that more sensible units are used. For example, if yearly periodic units are required, use `npp = 365.25` for daily data and `npp = 52.18` for weekly data. The argument only makes a difference to the actual fit if `mper` is not `NULL` or if `model = "pp"` (see **Details**).
`cmax`	Logical; if `FALSE` (the default), the model is fitted using all exceedences over the threshold. If `TRUE`, the model is fitted using cluster maxima, using clusters of exceedences derived from `clusters`.
`r, ulow, rlow`	Arguments used for the identification of clusters of exceedences (see `clusters`). Ignored if `cmax` is `FALSE` (the default).
`mper`	Controls the parameterization of the generalized Pareto model. Should be either `NULL` (the default), or a positive number (see **Details**). If `mper` is not `NULL` and `model = "pp"`, an error is returned.
`...`	Additional parameters, either for the model or for the optimization function `optim`. If parameters of the model are included they will be held fixed at the values given (see **Examples**).
`std.err`	Logical; if `TRUE` (the default), the standard errors are returned.
`corr`	Logical; if `TRUE`, the correlation matrix is returned.
`method`	The optimization method (see `optim` for details).
`warn.inf`	Logical; if `TRUE` (the default), a warning is given if the negative log-likelihood is infinite when evaluated at the starting values.

Details

The exeedances over the threshold `threshold` (if `cmax` is `FALSE`) or the maxima of the clusters of exeedances (if `cmax` is `TRUE`) are (if `model = "gpd"`) fitted to a generalized Pareto distribution (GPD) with location `threshold`. If `model = "pp"` the exceedances are fitted to a non-homogeneous Poisson process (Coles, 2001).

If `mper` is `NULL` (the default), the parameters of the model (if `model = "gpd"`) are `scale` and `shape`, for the scale and shape parameters of the GPD. If `model = "pp"` the parameters are `loc`, `scale` and `shape`. Under `model = "pp"` the parameters can be interpreted as parameters of the Generalized Extreme Value distribution, fitted to the maxima of `npp` random variables. In this case, the value of `npp` should be reasonably large.

For both characterizations, the shape parameters are equivalent. The scale parameter under the generalized Pareto characterization is equal to $b + s(u - a)$, where a, b and s are the location, scale and shape parameters under the Point Process characterization, and where u is the threshold.

If `mper` $= m$ is a positive value, then the generalized Pareto model is reparameterized so that the parameters are `rlevel` and `shape`, where `rlevel` is the m "period" return level, where "period" is defined via the argument `npp`.

The m "period" return level is defined as follows. Let G be the fitted generalized Pareto distribution function, with location `threshold` $= u$, so that $1-G(z)$ is the fitted probability of an exceedance over $z > u$ given an exceedance over u. The fitted probability of an

exceedance over $z > u$ is therefore $p(1 - G(z))$, where p is the estimated probabilty of exceeding u, which is given by the empirical proportion of exceedances. The m "period" return level z_m satisfies $p(1 - G(z_m)) = 1/(mN)$, where N is the number of points per period (multiplied by the estimate of the extremal index, if cluster maxima are fitted). In other words, z_m is the quantile of the fitted model that corresponds to the upper tail probability $1/(mN)$. If `mper` is infinite, then z_m is the upper end point, given by `threshold` minus `scale`/`shape`, and the shape parameter is then restricted to be negative.

Value

Returns an object of class `c("pot","uvevd","pot")`.

The generic accessor functions `fitted` (or `fitted.values`), `std.errors`, `deviance`, `logLik` and `AIC` extract various features of the returned object.

The function `profile` can be used to obtain deviance profiles for the model parameters. In particular, profiles of the m `period` return level z_m can be calculated and plotted when `mper` $= m$. The function `anova` compares nested models. The function `plot` produces diagnostic plots.

An object of class `c("pot","uvevd","pot")` is a list containing the following components

`estimate` A vector containing the maximum likelihood estimates.

`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`.

`threshold, r, ulow, rlow, npp`
The arguments of the same name.

`nhigh` The number of exceedences (if `cmax` is `FALSE`) or the number of clusters of exceedences (if `cmax` is `TRUE`).

`nat, pat` The number and proportion of exceedences.

`extind` The estimate of the extremal index (i.e. `nhigh` divided by `nat`). If `cmax` is `FALSE`, this is `NULL`.

`data` The data passed to the argument `x`.

`exceedances` The exceedences, or the maxima of the clusters of exceedences.

`mper` The argument `mper`.

`scale` The scale parameter for the fitted generalized Pareto distribution. If `mper` is `NULL` and `model = "gpd"` (the defaults), this will also be an element of `param`.

`call` The call of the current function.

Warning

The standard errors and the correlation matrix in the returned object are taken from the observed information, calculated by a numerical approximation. They must be interpreted with caution when the shape parameter is less than -0.5, because the usual asymptotic properties of maximum likelihood estimators do not then hold (Smith, 1985).

References

Smith, R. L. (1985) Maximum likelihood estimation in a class of non-regular cases. *Biometrika*, **72**, 67–90.

See Also

anova.evd, optim, plot.uvevd, profile.evd, profile2d.evd, mrlplot, tcplot

Examples

```
uvdata <- rgpd(100, loc = 0, scale = 1.1, shape = 0.2)
M1 <- fpot(uvdata, 1)
M2 <- fpot(uvdata, 1, shape = 0)
anova(M1, M2)
par(mfrow = c(2,2))
plot(M1)
## Not run: M1P <- profile(M1)
## Not run: plot(M1P)

M1 <- fpot(uvdata, 1, mper = 10)
M2 <- fpot(uvdata, 1, mper = 100)
## Not run: M1P <- profile(M1, which = "rlevel", conf=0.975, mesh=0.1)
## Not run: M2P <- profile(M2, which = "rlevel", conf=0.975, mesh=0.1)
## Not run: plot(M1P)
## Not run: plot(M2P)
```

---

frechet *The Frechet Distribution*

Description

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

Usage

```
dfrechet(x, loc=0, scale=1, shape=1, log = FALSE)
pfrechet(q, loc=0, scale=1, shape=1, lower.tail = TRUE)
qfrechet(p, loc=0, scale=1, shape=1, lower.tail = TRUE)
rfrechet(n, loc=0, scale=1, shape=1)
```

Arguments

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

Details

The Frechet distribution function with parameters `loc` $= a$, `scale` $= b$ and `shape` $= s$ is

$$G(z) = \exp\left\{-\left(\frac{z-a}{b}\right)^{-s}\right\}$$

for $z > a$ and zero otherwise, where $b > 0$ and $s > 0$.

Value

`dfrechet` gives the density function, `pfrechet` gives the distribution function, `qfrechet` gives the quantile function, and `rfrechet` generates random deviates.

See Also

`rgev`, `rgumbel`, `rrweibull`

Examples

```
dfrechet(2:4, 1, 0.5, 0.8)
pfrechet(2:4, 1, 0.5, 0.8)
qfrechet(seq(0.9, 0.6, -0.1), 2, 0.5, 0.8)
rfrechet(6, 1, 0.5, 0.8)
p <- (1:9)/10
pfrechet(qfrechet(p, 1, 2, 0.8), 1, 2, 0.8)
## [1] 0.1 0.2 0.3 0.4 0.5 0.6 0.7 0.8 0.9
```

---

gev *The Generalized Extreme Value Distribution*

Description

Density function, distribution function, quantile function and random generation for the generalized extreme value (GEV) distribution with location, scale and shape parameters.

Usage

```
dgev(x, loc=0, scale=1, shape=0, log = FALSE)
pgev(q, loc=0, scale=1, shape=0, lower.tail = TRUE)
qgev(p, loc=0, scale=1, shape=0, lower.tail = TRUE)
rgev(n, loc=0, scale=1, shape=0)
```

Arguments

`x, q`	Vector of quantiles.
`p`	Vector of probabilities.
`n`	Number of observations.
`loc, scale, shape`	Location, scale and shape parameters; the `shape` argument cannot be a vector (must have length one).
`log`	Logical; if `TRUE`, the log density is returned.
`lower.tail`	Logical; if `TRUE` (default), probabilities are P[X <= x], otherwise, P[X > x]

Details

The GEV distribution function with parameters `loc` $= a$, `scale` $= b$ and `shape` $= s$ is

$$G(z) = \exp\left[-\{1 + s(z-a)/b\}^{-1/s}\right]$$

for $1 + s(z-a)/b > 0$, where $b > 0$. If $s = 0$ the distribution is defined by continuity. If $1 + s(z-a)/b \leq 0$, the value z is either greater than the upper end point (if $s < 0$), or less than the lower end point (if $s > 0$).

The parametric form of the GEV encompasses that of the Gumbel, Frechet and reversed Weibull distributions, which are obtained for $s = 0$, $s > 0$ and $s < 0$ respectively. It was first introduced by Jenkinson (1955).

Value

`dgev` gives the density function, `pgev` gives the distribution function, `qgev` gives the quantile function, and `rgev` generates random deviates.

References

Jenkinson, A. F. (1955) The frequency distribution of the annual maximum (or minimum) of meteorological elements. *Quart. J. R. Met. Soc.*, **81**, 158–171.

See Also

`fgev`, `rfrechet`, `rgumbel`, `rrweibull`

Examples

```
dgev(2:4, 1, 0.5, 0.8)
pgev(2:4, 1, 0.5, 0.8)
qgev(seq(0.9, 0.6, -0.1), 2, 0.5, 0.8)
rgev(6, 1, 0.5, 0.8)
p <- (1:9)/10
pgev(qgev(p, 1, 2, 0.8), 1, 2, 0.8)
## [1] 0.1 0.2 0.3 0.4 0.5 0.6 0.7 0.8 0.9
```

---

gpd *The Generalized Pareto Distribution*

Description

Density function, distribution function, quantile function and random generation for the generalized Pareto distribution (GPD) with location, scale and shape parameters.

Usage

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

Arguments

`x, q`	Vector of quantiles.
`p`	Vector of probabilities.
`n`	Number of observations.
`loc, scale, shape`	Location, scale and shape parameters; the `shape` argument cannot be a vector (must have length one).
`log`	Logical; if `TRUE`, the log density is returned.
`lower.tail`	Logical; if `TRUE` (default), probabilities are P[X <= x], otherwise, P[X > x]

Details

The generalized Pareto distribution function (Pickands, 1975) with parameters `loc` $= a$, `scale` $= b$ and `shape` $= s$ is

$$G(z) = 1 - \{1 + s(z - a)/b\}^{-1/s}$$

for $1 + s(z - a)/b > 0$ and $z > a$, where $b > 0$. If $s = 0$ the distribution is defined by continuity.

Value

`dgpd` gives the density function, `pgpd` gives the distribution function, `qgpd` gives the quantile function, and `rgpd` generates random deviates.

References

Pickands, J. (1975) Statistical inference using extreme order statistics. *Annals of Statistics*, **3**, 119–131.

See Also

`fpot`, `rgev`

Examples

```
dgpd(2:4, 1, 0.5, 0.8)
pgpd(2:4, 1, 0.5, 0.8)
qgpd(seq(0.9, 0.6, -0.1), 2, 0.5, 0.8)
rgpd(6, 1, 0.5, 0.8)
p <- (1:9)/10
pgpd(qgpd(p, 1, 2, 0.8), 1, 2, 0.8)
## [1] 0.1 0.2 0.3 0.4 0.5 0.6 0.7 0.8 0.9
```

`gumbel` *The Gumbel Distribution*

Description

Density function, distribution function, quantile function and random generation for the Gumbel distribution with location and scale parameters.

Usage

```
dgumbel(x, loc=0, scale=1, log = FALSE)
pgumbel(q, loc=0, scale=1, lower.tail = TRUE)
qgumbel(p, loc=0, scale=1, lower.tail = TRUE)
rgumbel(n, loc=0, scale=1)
```

Arguments

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

Details

The Gumbel distribution function with parameters `loc` $= a$ and `scale` $= b$ is

$$G(z) = \exp\left\{-\exp\left[-\left(\frac{z-a}{b}\right)\right]\right\}$$

for all real z, where $b > 0$.

Value

`dgumbel` gives the density function, `pgumbel` gives the distribution function, `qgumbel` gives the quantile function, and `rgumbel` generates random deviates.

Interesting Fact

Gumbel is the name of my teddy bear. And a famous statistician.

See Also

`rfrechet`, `rgev`, `rrweibull`

Examples

```
dgumbel(-1:2, -1, 0.5)
pgumbel(-1:2, -1, 0.5)
qgumbel(seq(0.9, 0.6, -0.1), 2, 0.5)
rgumbel(6, -1, 0.5)
p <- (1:9)/10
pgumbel(qgumbel(p, -1, 2), -1, 2)
## [1] 0.1 0.2 0.3 0.4 0.5 0.6 0.7 0.8 0.9
```

---

lisbon *Annual Maximum Wind Speeds at Lisbon*

Description

A numeric vector containing annual maximum wind speeds, in kilometers per hour, from 1941 to 1970 at Lisbon, Portugal.

Usage

```
data(lisbon)
```

Format

A vector containing 30 observations.

Source

Tiago de Oliveira, J. (1997) *Statistical Analysis of Extremes.* Pendor.

---

marma *Simulate MARMA(p,q) Processes*

Description

Simulation of MARMA(p,q) processes.

Usage

```
marma(n, p = 0, q = 0, psi, theta, init = rep(0, p), n.start = p,
    rand.gen = rfrechet, ...)
mar(n, p = 1, psi, init = rep(0, p), n.start = p, rand.gen =
    rfrechet, ...)
mma(n, q = 1, theta, rand.gen = rfrechet, ...)
```

Arguments

`n`	The number of observations.
`p`	The AR order of the MARMA process.
`q`	The MA order of the MARMA process.
`psi`	A vector of non-negative parameters, of length `p`. Can be omitted if `p` is zero.
`theta`	A vector of non-negative parameters, of length `q`. Can be omitted if `q` is zero.
`init`	A vector of non-negative starting values, of length `p`.
`n.start`	A non-negative value denoting the length of the burn-in period. If `n.start` is less than `p`, then `p` minus `n.start` starting values will be included in the output series.
`rand.gen`	A simulation function to generate the innovations.
`...`	Additional arguments for `rand.gen`. Most usefully, the scale and shape parameters of the innovations generated by `rfrechet` can be specified by `scale` and `shape` respectively.

Details

A max autoregressive moving average process $\{X_k\}$, denoted by MARMA(p,q), satisfies

$$X_k = \max\{\phi_1 X_{k-1}, \ldots, \phi_p X_{k-p}, \epsilon_k, \theta_1 \epsilon_{k-1}, \ldots, \theta_q \epsilon_{k-q}\}$$

where `phi` $= (\phi_1, \ldots, \phi_p)$ and `theta` $= (\theta_1, \ldots, \theta_q)$ are non-negative vectors of parameters, and where $\{\epsilon_k\}$ is a series of *iid* random variables with a common distribution defined by `rand.gen`.

The functions `mar` and `mma` generate MAR(p) and MMA(q) processes respectively. A MAR(p) process $\{X_k\}$ is equivalent to a MARMA(p, 0) process, so that

$$X_k = \max\{\phi_1 X_{k-1}, \ldots, \phi_p X_{k-p}, \epsilon_k\}.$$

A MMA(q) process $\{X_k\}$ is equivalent to a MARMA(0, q) process, so that

$$X_k = \max\{\epsilon_k, \theta_1 \epsilon_{k-1}, \ldots, \theta_q \epsilon_{k-q}\}.$$

Value

A numeric vector of length `n`.

See Also

evmc

Examples

```
marma(100, p = 1, q = 1, psi = 0.75, theta = 0.65)
mar(100, psi = 0.85, n.start = 20)
mma(100, q = 2, theta = c(0.75, 0.8))
```

`mrlplot` *Empirical Mean Residual Life Plot*

Description

The empirical mean residual life plot.

Usage

```
mrlplot(data, tlim, nt = max(100, length(data)), lty = c(2,1,2),
    col = 1, conf = 0.95, main = "Mean Residual Life Plot", xlab =
    "Threshold", ylab = "Mean Excess", ...)
```

Arguments

`data`	A numeric vector.
`tlim`	A numeric vector of length two, giving the limits for the thresholds at which the mean residual life plot is evaluated. If `tlim` is not given, sensible defaults are used.
`nt`	The number of thresholds at which the mean residual life plot is evaluated.
`lty, col`	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.
`main`	Plot title.
`xlab, ylab`	x and y axis labels.
`...`	Other arguments to be passed to `matplot`.

Details

The empirical mean residual life plot is the locus of points

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

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

See Also

fpot, matplot, tcplot

Examples

```
data(portpirie)
mrlplot(portpirie)
```

---

mvevd *Parametric Multivariate Extreme Value Distributions*

Description

Density function, distribution function and random generation for the multivariate logistic and multivariate asymmetric logistic models.

Usage

```
pmvevd(q, dep, asy, model = c("log", "alog"), d = 2, mar = c(0,1,0),
    lower.tail = TRUE)
rmvevd(n, dep, asy, model = c("log", "alog"), d = 2, mar = c(0,1,0))
dmvevd(x, dep, asy, model = c("log", "alog"), d = 2, mar = c(0,1,0),
    log = FALSE)
```

Arguments

x, q	A vector of length d or a matrix with d columns, in which case the density/distribution is evaluated across the rows.
n	Number of observations.
dep	The dependence parameter(s). For the logistic model, should be a single value. For the asymmetric logistic model, should be a vector of length $2^d - d - 1$, or a single value, in which case the value is used for each of the $2^d - d - 1$ parameters (see **Details**).
asy	The asymmetry parameters for the asymmetric logistic model. Should be a list with $2^d - 1$ vector elements containing the asymmetry parameters for each separate component (see **Details**).
model	The specified model; a character string. Must be either "log" (the default) or "alog" (or any unique partial match), for the logistic and asymmetric logistic models respectively.
d	The dimension.
mar	A vector of length three containing marginal parameters for every univariate margin, or a matrix with three columns where each column represents a vector of values to be passed to the corresponding marginal parameter. It can also be a list with d elements, such that each element is either a vector of length three or a matrix with three columns, in which case the ith element represents the marginal parameters on the ith margin.
log	Logical; if TRUE, the log density is returned.
lower.tail	Logical; if TRUE (default), the distribution function is returned; the survivor function is returned otherwise.

Details

Define

$$y_i = y_i(z_i) = \{1 + s_i(z_i - a_i)/b_i\}^{-1/s_i}$$

for $1 + s_i(z_i - a_i)/b_i > 0$ and $i = 1, \ldots, d$, where the marginal parameters are given by (a_i, b_i, s_i), $b_i > 0$. If $s_i = 0$ then y_i is defined by continuity. Let $z = (z_1, z_2, \ldots, z_d)$. In each of the multivariate distributions functions $G(z)$ given below, the univariate margins are generalized extreme value, so that $G(z_i) = \exp(-y_i)$ for $i = 1, \ldots, d$. If $1+s_i(z_i-a_i)/b_i \leq 0$ for some $i = 1, \ldots, d$, the value z_i is either greater than the upper end point (if $s_i < 0$), or less than the lower end point (if $s_i > 0$), of the ith univariate marginal distribution.

`model = "log"` (Gumbel, 1960)

The `d` dimensional multivariate logistic distribution function with parameter `dep` $= r$ is

$$G(z) = \exp\left\{-\left(\sum\nolimits_{i=1}^{d} y_i^{1/r}\right)^r\right\}$$

where $0 < r \leq 1$. This is a special case of the multivariate asymmetric logistic model.

`model = "alog"` (Tawn, 1990)

Let B be the set of all non-empty subsets of $\{1, \ldots, d\}$, let $B_1 = \{b \in B : |b| = 1\}$, where $|b|$ denotes the number of elements in the set b, and let $B_{(i)} = \{b \in B : i \in b\}$. The `d` dimensional multivariate asymmetric logistic distribution function is

$$G(z) = \exp\left\{-\sum\nolimits_{b \in B}\left[\sum\nolimits_{i \in b}(t_{i,b}y_i)^{1/r_b}\right]^{r_b}\right\},$$

where the dependence parameters $r_b \in (0, 1]$ for all $b \in B \setminus B_1$, and the asymmetry parameters $t_{i,b} \in [0, 1]$ for all $b \in B$ and $i \in b$. The constraints $\sum_{b \in B_{(i)}} t_{i,b} = 1$ for $i = 1, \ldots, d$ ensure that the marginal distributions are generalized extreme value. Further constraints arise from the possible redundancy of asymmetry parameters in the expansion of the distribution form. Let $b_{-i_0} = \{i \in b : i \neq i_0\}$. If $r_b = 1$ for some $b \in B \setminus B_1$ then $t_{i,b} = 0$ for all $i \in b$. Furthermore, if for some $b \in B \setminus B_1$, $t_{i,b} = 0$ for all $i \in b_{-i_0}$, then $t_{i_0,b} = 0$.

`dep` should be a vector of length $2^d - d - 1$ which contains $\{r_b : b \in B \setminus B_1\}$, with the order defined by the natural set ordering on the index. For example, for the trivariate model, `dep` $= (r_{12}, r_{13}, r_{23}, r_{123})$. `asy` should be a list with $2^d - 1$ elements. Each element is a vector which corresponds to a set $b \in B$, containing $t_{i,b}$ for every integer $i \in b$. The elements should be given using the natural set ordering on the $b \in B$, so that the first d elements are vectors of length one corresponding to the sets $\{1\}, \ldots, \{d\}$, and the last element is a a vector of length d, corresponding to the set $\{1, \ldots, d\}$. `asy` must be constructed to ensure that all constraints are satisfied or an error will occur.

Value

`pmvevd` gives the distribution function, `dmvevd` gives the density function and `rmvevd` generates random deviates, for the multivariate logistic or multivariate asymmetric logistic model.

Note

Multivariate extensions of other bivariate models are more complex. A multivariate extension of the Husler-Reiss model exists, involving a multidimensional integral and one parameter for each bivariate margin. Multivariate extensions for the negative logistic model can be derived but are considerably more complex and appear to be less flexible. The "multivariate negative logistic model" often presented in the literature (e.g. Kotz *et al*, 2000) is not a valid distribution function and should not be used.

The logistic and asymmetric logistic models respectively are simulated using Algorithms 2.1 and 2.2 in Stephenson(2003b).

The density function of the logistic model is evaluated using the representation of Shi(1995). The density function of the asymmetric logistic model is evaluated using the representation given in Stephenson(2003a).

References

Gumbel, E. J. (1960) Distributions des valeurs extremes en plusieurs dimensions. *Publ. Inst. Statist. Univ. Paris*, **9**, 171–173.

Kotz, S. and Balakrishnan, N. and Johnson, N. L. (2000) *Continuous Multivariate Distributions*, vol. 1. New York: John Wiley & Sons, 2nd edn.

Shi, D. (1995) Fisher information for a multivariate extreme value distribution. *Biometrika*, **82**(3), 644–649.

Stephenson, A. G. (2003a) *Extreme Value Distributions and their Application*. Ph.D. Thesis, Lancaster University, Lancaster, UK.

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

Tawn, J. A. (1990) Modelling multivariate extreme value distributions. *Biometrika*, **77**, 245–253.

See Also

rbvevd, rgev

Examples

```
pmvevd(matrix(rep(0:4,5), ncol=5), dep = .7, model = "log", d = 5)
pmvevd(rep(4,5), dep = .7, model = "log", d = 5)
rmvevd(10, dep = .7, model = "log", d = 5)
dmvevd(rep(-1,20), dep = .7, model = "log", d = 20, log = TRUE)

asy <- list(.4, .1, .6, c(.3,.2), c(.1,.1), c(.4,.1), c(.2,.3,.2))
pmvevd(rep(2,3), dep = c(.6,.5,.8,.3), asy = asy, model = "alog", d = 3)
asy <- list(.4, .0, .6, c(.3,.2), c(.1,.1), c(.4,.1), c(.2,.4,.2))
rmvevd(10, dep = c(.6,.5,.8,.3), asy = asy, model = "alog", d = 3)
dmvevd(rep(0,3), dep = c(.6,.5,.8,.3), asy = asy, model = "alog", d = 3)

asy <- list(0, 0, 0, 0, c(0,0), c(0,0), c(0,0), c(0,0), c(0,0), c(0,0),
  c(.2,.1,.2), c(.1,.1,.2), c(.3,.4,.1), c(.2,.2,.2), c(.4,.6,.2,.5))
rmvevd(10, dep = .7, asy = asy, model = "alog", d = 4)
rmvevd(10, dep = c(rep(1,6), rep(.7,5)), asy = asy, model = "alog", d = 4)
```

`ocmulgee` *Maximum Annual Flood Discharges of the Ocmulgee River*

Description

The `ocmulgee` data frame has 40 rows and 2 columns. The columns contain maximum annual flood discharges, in units of 1000 cubed feet per second, from the Ocmulgee River in Georgia, USA at Hawkinsville (upstream) and Macon (downstream), for the years 1910 to 1949. The row names give the years of observation.

Usage

```
data(ocmulgee)
```

Format

This data frame contains the following columns:

hawk A numeric vector containing maximum annual flood discharges at Hawkinsville (upstream).

macon A numeric vector containing maximum annual flood discharges at Macon (downstream).

Source

Gumbel, E. J. and Goldstein, N. (1964) Analysis of empirical bivariate extremal distributions. *J. Amer. Statist. Assoc.*, **59**, 794–816.

`oldage` *Oldest Ages for Swedish Males and Females*

Description

The `oldage` data frame has 66 rows and 2 columns. The columns contain the oldest ages at death for men and women in Sweden, for the period 1905–1970. The row names give the years of observation.

Usage

```
data(oldage)
```

Format

This data frame contains the following columns:

men A numeric vector containing the oldest ages at death for men.

women A numeric vector containing the oldest ages at death for women.

Source

Fransen, A. and Tiago de Oliveira, J. (1984) Statistical choice of univariate extreme models, part II, in *Statistical Extremes and Applications*, J. Tiago de Oliveira ed., 373–394, D. Reidel, Dordrect.

`order` *Distributions of Order Statistics*

Description

Density function, distribution function and random generation for a selected order statistic of a given number of independent variables from a specified distribution.

Usage

```
dorder(x, densfun, distnfun, ..., distn, mlen = 1, j = 1,
      largest = TRUE, log = FALSE)
porder(q, distnfun, ..., distn, mlen = 1, j = 1, largest = TRUE,
      lower.tail = TRUE)
rorder(n, quantfun, ..., distn, mlen = 1, j = 1, largest = TRUE)
```

Arguments

`x, q`	Vector of quantiles.
`n`	Number of observations.
`densfun, distnfun, quantfun`	Density, distribution and quantile function of the specified distribution. The density function must have a `log` argument (a simple wrapper can always be constructed to achieve this).
`...`	Parameters of the specified distribution.
`distn`	A character string, optionally specified as an alternative to `densfun`, `distnfun` and `quantfun` such that the density, distribution and quantile functions are formed upon the addition of the prefixes `d`, `p` and `q` respectively.
`mlen`	The number of independent variables.
`j`	The order statistic, taken as the `j`th largest (default) or smallest of `mlen`, according to the value of `largest`.
`largest`	Logical; if `TRUE` (default) use the `j`th largest order statistic, otherwise use the `j`th smallest.
`log`	Logical; if `TRUE`, the log density is returned.
`lower.tail`	Logical; if `TRUE` (default) probabilities are P[X <= x], otherwise P[X > x].

Value

`dorder` gives the density function, `porder` gives the distribution function and `qorder` gives the quantile function of a selected order statistic from a sample of size `mlen`, from a specified distibution. `rorder` generates random deviates.

See Also

`rextreme`, `rgev`

Examples

```
dorder(2:4, dnorm, pnorm, mean = 0.5, sd = 1.2, mlen = 5, j = 2)
dorder(2:4, distn = "norm", mean = 0.5, sd = 1.2, mlen = 5, j = 2)
dorder(2:4, distn = "exp", mlen = 2, j = 2)
porder(2:4, distn = "exp", rate = 1.2, mlen = 2, j = 2)
rorder(5, qgamma, shape = 1, mlen = 10, j = 2)
```

---

oxford *Annual Maximum Temperatures at Oxford*

Description

A numeric vector containing annual maximum temperatures, in degrees Fahrenheit, from 1901 to 1980 at Oxford, England.

Usage

```
data(oxford)
```

Format

A vector containing 80 observations.

Source

Tabony, R. C. (1983) Extreme value analysis in meteorology. *The Meteorological Magazine* **112**, 77–98.

---

plot.bvevd *Plot Diagnostics for a Bivariate EVD Object*

Description

Four plots (selectable by `which`) are currently provided: two conditional P-P plots (conditioning on each margin), a density plot and a dependence function plot. Plot diagnostics for the generalized extreme value margins (selectable by `mar` and `which`) are also available.

Usage

```
## S3 method for class 'bvevd':
plot(x, mar = 0, which = 1:4, main = c("Conditional Plot One",
    "Conditional Plot Two", "Density Plot", "Dependence Function"),
    ask = nb.fig < length(which) && dev.interactive(), ci = TRUE,
    jitter = FALSE, grid = 50, legend = TRUE, nplty = 2, blty = 3,
    method = "cfg", convex = FALSE, wf = function(t) t, ...)
```

Arguments

`x`	An object of class `"bvevd"`.
`mar`	If `mar = 1` or `mar = 2` diagnostics are given for the first or second generalized extreme value margin respectively. The values of the remaining parameters are then passed to the plot method `plot.uvevd`.
`which`	If a subset of the plots is required, specify a subset of the numbers `1:4`.
`main`	Title of each plot.
`ask`	Logical; if `TRUE`, the user is asked before each plot.
`ci`	Logical; if `TRUE` (the default), plot simulated 95% confidence intervals for the conditional P-P plots.
`jitter, grid, legend`	Arguments for the density plot. The (possibly transformed) data is plotted with a contour plot of the bivariate density of the fitted model. The density is evaluated at `grid^2` points. If `jitter` is `TRUE`, the data are jittered. This need only be used if the data contains repeated values. If `legend` is `TRUE` and if the fitted data contained a third column of mode `logical`, then a legend is included.
`nplty, blty, method, convex, wf`	Arguments to the dependence function plot. The dependence function for the fitted model is plotted and (optionally) compared to a non-parameteric estimate. See `abvnonpar` for a definition of the dependence function, and for a description of the arguments `method`, `modify` and `wf`, which alter the behaviour of the non-parametric estimator. `nplty` is the line type of the non-parametric estimate. To omit the non-parametric estimate set `nplty` to zero. `blty` is the line type of the triangular border. To omit the border estimate set `blty` to zero.
`...`	Other arguments to be passed through to plotting functions.

Details

The following discussion assumes that the fitted model is stationary. For non-stationary models the data are transformed to stationarity. The plot then corresponds to the distribution obtained when all covariates are zero.

A conditional P-P plot is a P-P plot for the condition distribution function of a bivariate evd object. Let $G(.|.)$ be the conditional distribution of the first margin given the second, under the fitted model. Let $z_1, \ldots, z_m$ be the data used in the fitted model, where $z_j = (z_{1j}, z_{2j})$ for $j = 1, \ldots, m$. The plot that (by default) is labelled Conditional Plot Two, conditioning on the second margin, consists of the points

$$\{(p_i, c_i), i = 1, \ldots, m\}$$

where $p_1, \ldots, p_m$ are plotting points defined by `ppoints` and c_i is the ith largest value from the sample $\{G(z_{j1}|z_{j2}), j = 1, \ldots, m\}$. The margins are reversed for Conditional Plot One, so that $G(.|.)$ is the conditional distribution of the second margin given the first.

See Also

`plot.uvevd`, `contour`, `jitter`, `abvnonpar`

Examples

```
bvdata <- rbvevd(100, dep = 0.6, model = "log")
M1 <- fbvevd(bvdata, model = "log")
## Not run: par(mfrow = c(2,2))
## Not run: plot(M1)
## Not run: plot(M1, mar = 1)
## Not run: plot(M1, mar = 2)
```

`plot.profile.evd` *Plot Profile Deviances and Calculate Profile Confidence Intervals*

Description

Displays profile deviances from a model profiled with `profile.evd`, and invisibly returns profile confidence intervals.

Usage

```
## S3 method for class 'profile.evd':
plot(x, which = names(x), main = NULL,
    ask = nb.fig < length(which) && dev.interactive(), ci = 0.95,
    clty = 2, ...)
```

Arguments

`x`	An object of class `"profile.evd"`.
`which`	A character vector giving the parameters for which the profile deviance is plotted, and for which profile confidence intervals are calculated. By default all profiled parameters in `x` are used.
`main`	Title of each plot; a character vector, the same length as `which`.
`ask`	Logical; if `TRUE`, the user is asked before each plot.
`ci`	A numeric vector. For each parameter in `which` profile confidence intervals are calculated, for each confidence coefficient in `ci` (but see **Warning**). The intervals are returned invisibly as a list of vectors/matrices. Each plot then (by default) includes horizonal lines that represent each interval.
`clty`	The line type of the horizontal lines that represent the profile confidence intervals. To omit the lines set `clty` to zero.
`...`	Other graphics parameters.

Value

Profile devainces are plotted for each parameter in `which`. A list with one element for each parameter in `which` is also returned invisibly. Each element is a vector of length two or a matrix with two columns, giving the lower and upper limits of each confidence interval.

Warning

The profile confidence intervals may not have confidence coefficient `ci`, because the usual asymptotic properties of maximum likelihood estimators may not hold. For the GEV model, the usual asymptotic properties hold when the shape parameter is greater than -0.5 (Smith, 1985). Fortunately, this is usually the case.

References

Smith, R. L. (1985) Maximum likelihood estimation in a class of non-regular cases. *Biometrika*, **72**, 67–90.

See Also

`plot.profile2d.evd`, `profile.evd`, `profile2d.evd`

Examples

```
uvdata <- rgev(100, loc = 0.13, scale = 1.1, shape = 0.2)
M1 <- fgev(uvdata)
## Not run: M1P <- profile(M1)
## Not run: par(mfrow = c(2,2))
## Not run: cint <- plot(M1P, ci = c(0.95, 0.99))
## Not run: cint
```

`plot.profile2d.evd` *Plot Joint Profile Devainces*

Description

Displays an image plot of the joint profile deviance from a model profiled with `profile.evd` and `profile2d.evd`.

Usage

```
## S3 method for class 'profile2d.evd':
plot(x, main = NULL,
    ci = c(0.5, 0.8, 0.9, 0.95, 0.975, 0.99, 0.995),
    col = heat.colors(8), intpts = 75, xaxs = "r", yaxs = "r", ...)
```

Arguments

`x`	An object of class `"profile2d.evd"`.
`main`	Title of plot; a character string.
`ci`	A numeric vector whose length is one less than the length of `col`. The colours of the image plot, excluding the background colour, represent confidence sets with confidence coefficients `ci` (but see **Warning**).
`col`	A list of colors such as that generated by `rainbow`, `heat.colors`, `topo.colors`, `terrain.colors` or similar functions.
`intpts`	If the package **akima** is available, interpolation is performed using `intpts` points for each parameter. The function is interpolated at `intpts^2` points in total.
`xaxs,yaxs`	Graphics parameters (see `par`). The default, `"r"`, overrides the default set by `image`.
`...`	Other parameters to be passed to `image`.

Warning

The sets represented by different colours may not be confidence sets with confidence coefficients `ci`, because the usual asymptotic properties of maximum likelihood estimators may not hold. For the GEV model, the usual asymptotic properties hold when the shape parameter is greater than -0.5 (Smith, 1985). Fortunately, this is usually the case.

References

Smith, R. L. (1985) Maximum likelihood estimation in a class of non-regular cases. *Biometrika*, **72**, 67–90.

See Also

`plot.profile.evd`, `profile.evd`, `profile2d.evd`

Examples

```
uvdata <- rgev(100, loc = 0.13, scale = 1.1, shape = 0.2)
M1 <- fgev(uvdata)
## Not run: M1P <- profile(M1)
## Not run: M1JP <- profile2d(M1, M1P, which = c("scale", "shape"))
## Not run: plot(M1JP)
```

`plot.uvevd` *Plot Diagnostics for a Univariate EVD Object*

Description

Four plots (selectable by `which`) are currently provided: a P-P plot, a Q-Q plot, a density plot and a return level plot.

Usage

```
## S3 method for class 'uvevd':
plot(x, which = 1:4, main = c("Probability Plot",
    "Quantile Plot", "Density Plot", "Return Level Plot"),
    ask = nb.fig < length(which) && dev.interactive(),
    ci = TRUE, adjust = 1, jitter = FALSE, nplty = 2, ...)
```

Arguments

`x`	An object that inherits from class `"uvevd"`.
`which`	If a subset of the plots is required, specify a subset of the numbers `1:4`.
`main`	Title of each plot.
`ask`	Logical; if `TRUE`, the user is asked before each plot.
`ci`	Logical; if `TRUE` (the default), plot simulated 95% confidence intervals for the P-P, Q-Q and return level plots.

`adjust, jitter, nplty`
: Arguments to the density plot. The density of the fitted model is plotted with a rug plot and (optionally) a non-parameteric estimate. The argument `adjust` controls the smoothing bandwidth for the non-parametric estimate (see `density`). `jitter` is logical; if `TRUE`, the (possibly transformed) data are jittered to produce the rug plot. This need only be used if the data contains repeated values. `nplty` is the line type of the non-parametric estimate. To omit the non-parametric estimate set `nplty` to zero.

`...`
: Other parameters to be passed through to plotting functions.

Details

The following discussion assumes that the fitted model is stationary. For non-stationary generalized extreme value models the data are transformed to stationarity. The plot then corresponds to the distribution obtained when all covariates are zero.

The P-P plot consists of the points

$$\{(G_n(z_i), G(z_i)), i = 1, \ldots, m\}$$

where G_n is the empirical distribution function (defined using `ppoints`), G is the model based estimate of the distribution (generalized extreme value or generalized Pareto), and $z_1, \ldots, z_m$ are the data used in the fitted model, sorted into ascending order.

The Q-Q plot consists of the points

$$\{(G^{-1}(p_i), z_i), i = 1, \ldots, m\}$$

where G^{-1} is the model based estimate of the quantile function (generalized extreme value or generalized Pareto), $p_1, \ldots, p_m$ are plotting points defined by `ppoints`, and $z_1, \ldots, z_m$ are the data used in the fitted model, sorted into ascending order.

The return level plot for generalized extreme value models is defined as follows.

Let G be the generalized extreme value distribution function, with location, scale and shape parameters a, b and s respectively. Let z_t be defined by $G(z_t) = 1 - 1/t$. In common terminology, z_t is the return level associated with the return period t.

Let $y_t = -1/\log(1 - 1/t)$. It follows that

$$z_t = a + b(y_t^s - 1)/s.$$

When $s = 0$, z_t is defined by continuity, so that

$$z_t = a + b\log(y_t).$$

The curve within the return level plot is z_t plotted against y_t on a logarithmic scale, using maximum likelihood estimates of (a, b, s). If the estimate of s is zero, the curve will be linear. For large values of t, y_t is approximately equal to the return period t. It is usual practice to label the x-axis as the return period.

The points on the plot are

$$\{(-1/\log(p_i), z_i), i = 1, \ldots, m\}$$

where $p_1, \ldots, p_m$ are plotting points defined by `ppoints`, and $z_1, \ldots, z_m$ are the data used in the fitted model, sorted into ascending order. For a good fit the points should lie "close" to the curve.

The return level plot for peaks over threshold models is defined as follows.

Let G be the generalized Pareto distribution function, with location, scale and shape parameters u, b and s respectively, where u is the model threshold. Let z_m denote the m period return level (see `fpot` and the notation therein). It follows that

$$z_m = u + b((pmN)^s - 1)/s.$$

When $s = 0$, z_m is defined by continuity, so that

$$z_m = u + b \log(pmN).$$

The curve within the return level plot is z_m plotted against m on a logarithmic scale, using maximum likelihood estimates of (b, s, p). If the estimate of s is zero, the curve will be linear.

The points on the plot are

$$\{(1/(pN(1 - p_i)), z_i), i = 1, \ldots, m\}$$

where $p_1, \ldots, p_m$ are plotting points defined by `ppoints`, and $z_1, \ldots, z_m$ are the data used in the fitted model, sorted into ascending order. For a good fit the points should lie "close" to the curve.

See Also

`plot.bvevd`, `density`, `jitter`, `rug`, `ppoints`

Examples

```
uvdata <- rgev(100, loc = 0.13, scale = 1.1, shape = 0.2)
M1 <- fgev(uvdata)
## Not run: par(mfrow = c(2,2))
## Not run: plot(M1)

uvdata <- rgpd(100, loc = 0, scale = 1.1, shape = 0.2)
M1 <- fpot(uvdata, 1)
## Not run: par(mfrow = c(2,2))
## Not run: plot(M1)
```

---

portpirie — *Annual Maximum Sea Levels at Port Pirie*

Description

A numeric vector containing annual maximum sea levels, in metres, from 1923 to 1987 at Port Pirie, South Australia.

Usage

```
data(portpirie)
```

Format

A vector containing 65 observations.

Source

Tawn, J. A. (1993) Extreme sea-levels, in *Statistics in the Environment*, 243–263, eds. V. Barnett and F. Turkman, Wiley.

References

Coles, S. G. (2001) *An Introduction to Statistical Modeling of Extreme Values.* London: Springer-Verlag.

`profile.evd` *Method for Profiling EVD Objects*

Description

Calculate profile traces for fitted models.

Usage

```
## S3 method for class 'evd':
profile(fitted, which = names(fitted$estimate), conf = 0.999,
    mesh = fitted$std.err[which]/4, xmin = rep(-Inf, length(which)),
    xmax = rep(Inf, length(which)), convergence = FALSE, method = "BFGS",
    control = list(maxit = 500), ...)
```

Arguments

`fitted`	An object of class `"evd"`.
`which`	A character vector giving the model parameters that are to be profiled. By default, all parameters are profiled.
`conf`	Controls the range over which the parameters are profiled. The profile trace is constructed so that (assuming the usual asymptotic properties hold) profile confidence intervals with confidence coefficients `conf` or less can be derived from it.
`mesh`	A numeric vector containing one value for each parameter in `which`. The values represent the distance between the points profiled. By default `mesh` is one quarter of the standard errors. If the fitted object does not contain standard errors the argument must be specified. The argument should also be specified when an estimator is on or close to a parameter boundary, since the approximated "standard error" will then be close to zero.
`xmin, xmax`	Numeric vectors containing one value for each parameter in `which`. Each value represents the theoretical lower/upper bound of the corresponding parameter. The arguments are needed only when a parameter has a lower/upper bound at which the likelihood is non-zero. Do not use these arguments to specify plotting ranges in a subsequent plot (as they are used in the calculation of profile confidence intervals); to do this use `xlim` in the call to `plot`.
`convergence`	Logical; print convergence code after each optimization? (A warning is given for each non-zero convergence code, irrespective of the value of `convergence`.)

method	The optimization method.
control	Passed to optim. See optim for details.
...	Ignored.

Value

An object of class "profile.evd", which is a list with an element for each parameter being profiled. The elements are matrices. The first column contains the values of the profiled parameter. The second column contains profile deviances. The remaining columns contain the constrained maximum likelihood estimates for the remaining model parameters.

See Also

profile2d.evd, plot.profile.evd

Examples

```
uvdata <- rgev(100, loc = 0.13, scale = 1.1, shape = 0.2)
M1 <- fgev(uvdata)
## Not run: M1P <- profile(M1)
## Not run: par(mfrow = c(2,2))
## Not run: cint <- plot(M1P)
## Not run: cint
```

---

profile2d.evd *Method for Profiling EVD Objects*

Description

Calculate joint profile traces for fitted models.

Usage

```
## S3 method for class 'evd':
profile2d(fitted, prof, which, pts = 20, convergence =
    FALSE, method = "Nelder-Mead", control = list(maxit = 5000), ...)
```

Arguments

fitted	An object of class "evd".
prof	An object of class "profile.evd", created using profile.evd with argument fitted. The object must contain the (marginal) profile traces for the two parameters specified in which.
which	A character vector of length two containing the original model parameters that are to be jointly profiled.
pts	The number of distinct values used for each profiled parameter in which. There are pts^2 optimizations performed in total.
convergence	Logical; print convergence code after each optimization? (A warning is given for each non-zero convergence code, irrespective of the value of convergence.)
method	The optimization method.
control	Passed to optim. See optim for details.
...	Ignored.

Value

An object of class `"profile2d.evd"`, which is a list with three elements. The first element, a matrix named `trace`, has the same structure as the elements of an object of class `"profile.evd"`. The last two elements give the distinct values used for each profiled parameter in `which`.

See Also

`profile.evd`, `plot.profile2d.evd`

Examples

```
uvdata <- rgev(100, loc = 0.13, scale = 1.1, shape = 0.2)
M1 <- fgev(uvdata)
## Not run: M1P <- profile(M1)
## Not run: M1JP <- profile2d(M1, M1P, which = c("scale", "shape"))
## Not run: plot(M1JP)
```

`rweibull` *The Reversed Weibull Distribution*

Description

Density function, distribution function, quantile function and random generation for the reversed Weibull distribution with location, scale and shape parameters.

Usage

```
drweibull(x, loc=0, scale=1, shape=1, log = FALSE)
prweibull(q, loc=0, scale=1, shape=1, lower.tail = TRUE)
qrweibull(p, loc=0, scale=1, shape=1, lower.tail = TRUE)
rrweibull(n, loc=0, scale=1, shape=1)
```

Arguments

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

Details

The reversed Weibull distribution function with parameters `loc` $= a$, `scale` $= b$ and `shape` $= s$ is

$$G(z) = \exp\left\{-\left[-\left(\frac{z-a}{b}\right)\right]^{s}\right\}$$

for $z < a$ and one otherwise, where $b > 0$ and $s > 0$.

Value

`drweibull` gives the density function, `prweibull` gives the distribution function, `qrweibull` gives the quantile function, and `rrweibull` generates random deviates.

Note

Within extreme value theory the reversed Weibull distibution is usually referred to as the Weibull distribution. I make a distinction to avoid confusion with the three-parameter distribution used in survival analysis, which is related by a change of sign to the distribution given above.

See Also

`rfrechet`, `rgev`, `rgumbel`

Examples

```
drweibull(-5:-3, -1, 0.5, 0.8)
prweibull(-5:-3, -1, 0.5, 0.8)
qrweibull(seq(0.9, 0.6, -0.1), 2, 0.5, 0.8)
rrweibull(6, -1, 0.5, 0.8)
p <- (1:9)/10
prweibull(qrweibull(p, -1, 2, 0.8), -1, 2, 0.8)
## [1] 0.1 0.2 0.3 0.4 0.5 0.6 0.7 0.8 0.9
```

---

sask — *Maximum Annual Flood Discharges of the North Saskachevan River*

Description

A numeric vector containing maximum annual flood discharges, in units of 1000 cubic feet per second, of the North Saskachevan River at Edmonton, over a period of 47 years. Unfortunately, the data are ordered from largest to smallest.

Usage

```
data(sask)
```

Format

A vector containing 47 observations.

Source

van Montfort, M. A. J. (1970) On testing that the distribution is of type I when type II is the alternative. *J. Hydrology*, **11**, 421–427.

sealevel — *Annual Sea Level Maxima at Dover and Harwich*

Description

The `sealevel` data frame has 81 rows and 2 columns. The columns contain annual sea level maxima from 1912 to 1992 at Dover and Harwich respectively, two sites on the coast of Britain. The row names give the years of observation. There are 39 missing values.

Usage

```
data(sealevel)
```

Format

This data frame contains the following columns:

dover A numeric vector containing annual sea level maxima at Dover, including 9 missing values.

harwich A numeric vector containing sea annual level maxima at Harwich, including 30 missing values.

Source

Coles, S. G. and Tawn, J. A. (1990) Statistics of coastal flood prevention. *Phil. Trans. R. Soc. Lond., A* **332**, 457–476.

sealevel2 — *Annual Sea Level Maxima at Dover and Harwich with Indicator*

Description

The `sealevel2` data frame has 81 rows and 3 columns. The first two columns contain annual sea level maxima from 1912 to 1992 at Dover and Harwich respectively, two sites on the coast of Britain. The third column is a logical vector denoting whether or not the maxima in a given year are assumed to have derived from the same storm event; this assumption is made if the times of obsevation of the maxima are at most 48 hours apart. The row names give the years of observation. There are 39 missing data values. There are only nine non-missing logical values.

Usage

```
data(sealevel2)
```

Format

This data frame contains the following columns:

dover A numeric vector containing annual sea level maxima at Dover, including 9 missing values.

harwich A numeric vector containing sea annual level maxima at Harwich, including 30 missing values.

case A logical vector denoting whether or not the maxima are assumed to have derived from the same storm event.

Source

Coles, S. G. and Tawn, J. A. (1990) Statistics of coastal flood prevention. *Phil. Trans. R. Soc. Lond., A* **332**, 457–476.

---

tcplot *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, tlim, model = c("gpd","pp"), 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.
`tlim`	A numeric vector of length two, giving the limits for the thresholds at which the model is fitted.
`model`	The model; either `"gpd"` (the default) or `"pp"`, for the Generalized Pareto or Point Process representations respectively.
`cmax`	Logical; if `FALSE` (the default), the models are fitted using all exceedences over the thresholds. If `TRUE`, the models are fitted using cluster maxima, using clusters of exceedences derived from `clusters`.
`r, ulow, rlow`	Arguments used for the identification of clusters of exceedences (see `clusters`). 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, so that `npar = 2` when `model = "gpd"` (the default) and `npar = 3` when `model = "pp"`.
`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 overplotted 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 `fpot`.

Details

For each of the `nt` thresholds a peaks over threshold model is fitted using the function `fpot`. When `model = "gpd"` (the default), 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. When `model = "pp"` the maximum likelihood estimates for the location, scale and shape parameters are plotted against the thresholds. (The modified scale parameter in the `"gpd"` case is equivalent to the scale parameter in the `"pp"` case.) 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

See Also

`fpot`, `mrlplot`, `clusters`

Examples

```
data(portpirie)
tlim <- c(3.6, 4.2)
## Not run: tcplot(portpirie, tlim)
## Not run: tcplot(portpirie, tlim, nt = 100, lwd = 3, type = "l")
## Not run: tcplot(portpirie, tlim, model = "pp")
```

`uccle` *Rainfall Maxima at Uccle, Belguim*

Description

The `uccle` data frame has 35 rows and 4 columns. The columns contain annual rainfall maxima (in milimetres) from 1938 to 1972 at Uccle, Belguim, over the durations of one day, one hour, ten minutes and one minute. The row names give the years of observation.

Usage

```
data(uccle)
```

Format

This data frame contains the following columns:

day Annual daily rainfall maxima.

hour Annual hourly rainfall maxima.

tmin Annual rainfall maxima over ten minute durations.

min Annual rainfall maxima over one minute durations.

Source

Sneyers, R. (1977) L'intensité maximale des précipitations en Belgique. *Inst. Royal Météor. Belgique, B* **86**.

`venice` *Largest Sea Levels in Venice*

Description

The `venice` data frame has 51 rows and 10 columns. The jth column contains the jth largest sea levels in Venice, for the years 1931–1981. Only the largest six measurements are available for the year 1935; the corresponding row contains four missing values. The years for each set of measurements are given as row names.

Usage

```
data(venice)
```

Format

A data frame with 51 rows and 10 columns.

Source

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

References

Coles, S. G. (2001) *An Introduction to Statistical Modeling of Extreme Values*. London: Springer-Verlag.

Index

∗Topic **datasets**
failure, 23
fox, 36
lisbon, 44
ocmulgee, 49
oldage, 50
oxford, 52
portpirie, 58
sask, 62
sealevel, 63
sealevel2, 63
uccle, 65
venice, 66
∗Topic **distribution**
abvpar, 4
atvpar, 8
bvevd, 10
evmc, 19
extreme, 22
frechet, 39
gev, 40
gpd, 41
gumbel, 43
marma, 44
mvevd, 47
order, 51
rweibull, 61
∗Topic **hplot**
chiplot, 14
mrlplot, 46
plot.bvevd, 52
plot.profile.evd, 54
plot.profile2d.evd, 55
plot.uvevd, 56
tcplot, 64
∗Topic **internal**
evd-internal, 19
∗Topic **manip**
clusters, 17
exi, 20
∗Topic **models**
anova.evd, 6
fbvevd, 23
fbvpot, 27
fextreme, 30
fgev, 32
forder, 34
fpot, 36
profile.evd, 59
profile2d.evd, 60
∗Topic **nonparametric**
abvnonpar, 1
atvnonpar, 7

abvalog *(evd-internal)*, 19
abvaneglog *(evd-internal)*, 19
abvbilog *(evd-internal)*, 19
abvct *(evd-internal)*, 19
abvhr *(evd-internal)*, 19
abvlog *(evd-internal)*, 19
abvnegbilog *(evd-internal)*, 19
abvneglog *(evd-internal)*, 19
abvnonpar, 1, *5*, *8*, *53*
abvpar, *2*, 4, *4*, *10*, *14*, *16*, *25*, *30*
AIC, *25*, *29*, *31*, *33*, *35*, *38*
anova.evd, 6, *27*, *30*, *31*, *34*, *35*, *39*
atvalog *(evd-internal)*, 19
atvlog *(evd-internal)*, 19
atvnonpar, *4*, 7, *10*
atvpar, *5*, 8, *8*

bvcpp *(evd-internal)*, 19
bvdens *(evd-internal)*, 19
bvdepfn *(evd-internal)*, 19
bvdp *(evd-internal)*, 19
bvevd, 10
bvpost.optim *(evd-internal)*, 19
bvstart.vals *(evd-internal)*, 19
bvtpost.optim *(evd-internal)*, 19

ccop *(evd-internal)*, 19
chiplot, 14
clusters, 17, *21*, *37*, *64*, *65*
contour, *53*

dbvalog *(evd-internal)*, 19
dbvaneglog *(evd-internal)*, 19

dbvbilog *(evd-internal)*, 19
dbvct *(evd-internal)*, 19
dbvevd *(bvevd)*, 10
dbvhr *(evd-internal)*, 19
dbvlog *(evd-internal)*, 19
dbvnegbilog *(evd-internal)*, 19
dbvneglog *(evd-internal)*, 19
dens *(evd-internal)*, 19
density, *57*, *58*
deviance, *25*, *29*, *31*, *33*, *35*, *38*
dextreme *(extreme)*, 22
dfrechet *(frechet)*, 39
dgev *(gev)*, 40
dgpd *(gpd)*, 41
dgumbel *(gumbel)*, 43
dmvalog *(evd-internal)*, 19
dmvevd *(mvevd)*, 47
dmvlog *(evd-internal)*, 19
dorder *(order)*, 51
drweibull *(rweibull)*, 61

evd-internal, 19
evmc, 19, *45*
exi, *18*, 20
extreme, 22

failure, 23
fbvalog *(evd-internal)*, 19
fbvaneglog *(evd-internal)*, 19
fbvbilog *(evd-internal)*, 19
fbvcalog *(evd-internal)*, 19
fbvcaneglog *(evd-internal)*, 19
fbvcbilog *(evd-internal)*, 19
fbvcct *(evd-internal)*, 19
fbvchr *(evd-internal)*, 19
fbvclog *(evd-internal)*, 19
fbvcnegbilog *(evd-internal)*, 19
fbvcneglog *(evd-internal)*, 19
fbvcpot *(evd-internal)*, 19
fbvct *(evd-internal)*, 19
fbvevd, *5*, *6*, *17*, 23, *29*, *30*
fbvhr *(evd-internal)*, 19
fbvlog *(evd-internal)*, 19
fbvnegbilog *(evd-internal)*, 19
fbvneglog *(evd-internal)*, 19
fbvpbilog *(evd-internal)*, 19
fbvpct *(evd-internal)*, 19
fbvplog *(evd-internal)*, 19
fbvpnegbilog *(evd-internal)*, 19
fbvpneglog *(evd-internal)*, 19
fbvpot, *16*, *17*, 27
fbvppot *(evd-internal)*, 19
fextreme, *6*, 30, *35*

fgev, *2*, *4*, *6*, *8*, 32, *41*
fgev.norm *(evd-internal)*, 19
fgev.quantile *(evd-internal)*, 19
fitted, *25*, *29*, *31*, *33*, *35*, *38*
fitted.evd *(fgev)*, 32
fitted.values, *25*, *29*, *31*, *33*, *35*, *38*
forder, *6*, *31*, 34
fox, 36
fpot, 36, *42*, *47*, *58*, *65*
fpot.norm *(evd-internal)*, 19
fpot.quantile *(evd-internal)*, 19
frechet, 39

gev, 40
gpd, 41
gumbel, 43

image, *7*, *9*, *10*

jitter, *53*, *58*

lisbon, 44
logLik, *25*, *29*, *31*, *33*, *35*, *38*
logLik.evd *(fgev)*, 32

mar *(marma)*, 44
marma, *20*, 44
matplot, *17*, *47*
mma *(marma)*, 44
mrlplot, *39*, 46, *65*
mtransform *(evd-internal)*, 19
mvalog.check *(evd-internal)*, 19
mvevd, 47

na.vals *(evd-internal)*, 19
nsloc.transform *(evd-internal)*, 19

ocmulgee, 49
oldage, 50
optim, *24*, *26–35*, *37–39*, *60*
order, 51
oxford, 52

par, *55*
pbvalog *(evd-internal)*, 19
pbvaneglog *(evd-internal)*, 19
pbvbilog *(evd-internal)*, 19
pbvct *(evd-internal)*, 19
pbvevd *(bvevd)*, 10
pbvhr *(evd-internal)*, 19
pbvlog *(evd-internal)*, 19
pbvnegbilog *(evd-internal)*, 19
pbvneglog *(evd-internal)*, 19
pcint *(evd-internal)*, 19

pextreme *(extreme)*, 22
pfrechet *(frechet)*, 39
pgev *(gev)*, 40
pgpd *(gpd)*, 41
pgumbel *(gumbel)*, 43
plot.bvevd, *26*, *27*, 52, *58*
plot.bvpot *(fbvpot)*, 27
plot.profile.evd, 54, *56*, *60*
plot.profile2d.evd, 55, *55*, *61*
plot.uvevd, *34*, *39*, *53*, 56
pmvalog *(evd-internal)*, 19
pmvevd *(mvevd)*, 47
pmvlog *(evd-internal)*, 19
porder *(order)*, 51
portpirie, 58
pp *(evd-internal)*, 19
ppoints, *53*, *57*, *58*
print.bvevd *(fbvevd)*, 23
print.bvpot *(fbvpot)*, 27
print.evd *(fgev)*, 32
print.pot *(fpot)*, 36
profile.evd, *27*, *34*, *39*, *54–56*, 59, *60*, *61*
profile2d *(profile2d.evd)*, 60
profile2d.evd, *27*, *34*, *39*, *55*, *56*, 60, *60*
prweibull *(rweibull)*, 61

qextreme *(extreme)*, 22
qfrechet *(frechet)*, 39
qgev *(gev)*, 40
qgpd *(gpd)*, 41
qgumbel *(gumbel)*, 43
qq *(evd-internal)*, 19
qrweibull *(rweibull)*, 61

rbvalog *(evd-internal)*, 19
rbvaneglog *(evd-internal)*, 19
rbvbilog *(evd-internal)*, 19
rbvct *(evd-internal)*, 19
rbvevd, *5*, *20*, *24*, *27*, *28*, *30*, *49*
rbvevd *(bvevd)*, 10
rbvhr *(evd-internal)*, 19
rbvlog *(evd-internal)*, 19
rbvnegbilog *(evd-internal)*, 19
rbvneglog *(evd-internal)*, 19
rextreme, *51*
rextreme *(extreme)*, 22
rfrechet, *41*, *43*, *62*
rfrechet *(frechet)*, 39
rgev, *14*, *22*, *40*, *42*, *43*, *49*, *51*, *62*
rgev *(gev)*, 40
rgpd *(gpd)*, 41
rgumbel, *40*, *41*, *62*
rgumbel *(gumbel)*, 43
rl *(evd-internal)*, 19
rmvalog *(evd-internal)*, 19
rmvevd, *9*, *10*, *14*
rmvevd *(mvevd)*, 47
rmvlog *(evd-internal)*, 19
rorder, *22*
rorder *(order)*, 51
rrweibull, *40*, *41*, *43*
rrweibull *(rweibull)*, 61
rug, *58*
rweibull, 61

sask, 62
sealevel, 63
sealevel2, 63
sep.bvdata *(evd-internal)*, 19
std.errors, *25*, *29*, *31*, *33*, *35*, *38*
std.errors *(fgev)*, 32
subsets *(evd-internal)*, 19

tcplot, *39*, *47*, 64
tvdepfn *(evd-internal)*, 19

uccle, 65

venice, 66