Package 'Renext'

July 2, 2014

Type Package

Title Renewal method for extreme values extrapolation

Version 2.1-0

Date 2013-10-03

Author Yves Deville <deville.yves@alpestat.com>, IRSN <renext@irsn.fr>

Maintainer Lise Bardet <lise.bardet@irsn.fr>

Depends R (>= 2.8.0), stats, graphics, evd

Imports numDeriv

Suggests MASS, ismev, XML

Description R package dedicated to some Extreme values problems and allowing the use of the so-called ``methode du renouvellement'' which is popular among french-speaking hydrologists.

License GPL (>= 2)

LazyData yes

URL https://gforge.irsn.fr/gf/project/renext

NeedsCompilation no

Repository CRAN

Date/Publication 2013-10-04 16:14:15

R topics documented:

`Renext-package` *Renewal method for extreme values extrapolation*

Description

This package proposes fits and diagnostics for the so-called *méthode du renouvellement*, an alternative to other "Peaks Over Threshold" (POT) methods. The *méthode du renouvellement* generalises the classical POT by allowing the exceedances over the threshold to follow a probability distribution which can differ from the Generalised Pareto Distribution (GPD). Weibull or gamma exceedances are sometimes preferred to GPD exceedances. The special case of exponential exceedances (which falls in the three families: GPD, Weibull and gamma) has a special interest since it allows exact inference for the (scalar) parameter and for the quantiles form OT data (only).

The package allows the joint use of possibly three kinds of data or information. The first kind is *classical exceedances*, or *"OT data"*, and is always be required (for this version). It can be completed with two kinds of data resulting from a temporal aggregation as is often the case for *historical data*. Both types are optional, and concern periods or *blocks* that must not overlap nor cross the OT period.

- *MAX data* correspond to the case where one knows the r largest observations over each block. The number r of may vary between blocks. This kind of data is often called 'r-largest', or "r-Largest Order Statistics" (r-LOS).
- *OTS data* (for OT Supplementary data) correspond to the case where one knows for each block b all the observations that exceeded a threshold u_b which is greater (usually much greater) than the main threshold u. The number r_b of such observations can be zero, in which case we may say that u_b is an unobserved level. A threshold u_b is sometimes called a *perception threshold*.

Historical data are often available in hydrology (e.g. for river flows) for large periods such as past centuries. An unobserved level can typically be related to a material benchmark.

Maximum likelihood estimation is made possible in this context of heterogeneous data. Inference is based on the asymptotic normality of parameter vector estimate and on linearisation ("delta method") for quantiles or parameter functions.

The package allows the use of "marked-process observations" data (datetime of event and level) where an interevent analysis can be useful. It also allows that the event dates are unknown and replaced by a much broader *block* indication, e.g. a year number. The key point is then that the "effective duration" (total duration of observation periods) is known. Event counts for blocks can be used to check the assumption of Poisson-distributed events.

The package development was initiated, directed and financed by the french *Institut de Radioprotection et de Sûreté Nucléaire* (IRSN). The package is a non-academic tool designed for applied analysis on case studies and investigations or comparisons on classical probabilistic models.

Additional information and material related to this package can be found at the URL https://gforge.irsn.fr/gf/project/renext.

Details

Package:	Renext
Type:	Package
Version:	2.1-0
Date:	2013-10-03
License:	GPL (>= 2)
LazyLoad:	yes

This package contains a function Renouv to fit "renouvellement" models.

Author(s)

Yves Deville <deville.yves@alpestat.com>, IRSN <renext@irsn.fr>

Maintainer: Lise Bardet <lise.bardet@irsn.fr>

References

- Miquel J. (1984) *Guide pratique d'estimation des probabilités de crues*, Eyrolles (coll. EDF DER)
- Coles S. (2001) *Introduction to Statistical Modelling of Extremes Values*, Springer.

See Also

The packages evd, ismev, extRemes, bayesevd, POT.

Examples

```
## Garonne data set
summary(Garonne)
plot(Garonne)

## Weibull exceedances
fG <- Renouv(x = Garonne,
             threshold = 3000,
             distname.y = "weibull",
             main = "Weibull fit for 'Garonne'")

coef(fG)
vcov(fG)
summary(fG)
logLik(fG)
## re-plot if needed
plot(fG)

## classical 'predict' method with usual formal args
predict(fG, newdata = c(100, 150, 200),
        level = c(0.8, 0.9))
```

---

barplotRenouv *Barplot for Renouv "Over Threshold" counts*

Description

Barplot for "Over Threshold" counts in time blocks (usually years)

Usage

```
barplotRenouv(data,
              blockname = colnames(data)[1],
              varname = colnames(data)[2],
              threshold = quantile(data[, varname], 0.2),
              na.block = NULL,
              plot = TRUE,
              main = NULL, xlab = NULL, ylab = NULL,
              mono = FALSE,
              prob.theo = 0.999,
              ...)
```

Arguments

data	A dataframe object containing the variables.
blockname	Name of the "block" variable (column in data). This variable should contain integers, or be of class "factor", but with integer values such as year numbers.
varname	Name of the variable (e.g. "Surge").
threshold	Only obs for which the variable exceeds threshold will be taken into account.
na.block	Values of blocks containing missing values. See the Details section.
plot	If FALSE tests are computed without producing any plot.
main	Character for main title or NULL in which case a default main title is used.
xlab	Character for x axis label or NULL in which case a default lab is used.
ylab	Character for y axis or NULL in which case a default lab is used.
mono	If FALSE barplot will have colors, else grayscale will be used.
prob.theo	The total theoretical probability corresponding to the plotted (theoretical) bars.
...	Further args to be passed to barplot.

Details

Blocks described in the na.block are omitted in the determination of counts. The object given in the na.block is coerced to character and the same is done for values of block before comparing them to the na.block values. If block variable is of class factor with levels representing years (e.g. 1980, 1981, etc.) missing blocks can be specified either as c("1980", "1981") or as numeric c(1980, 1981).

For the chi-square test, counts for neighbouring frequency classes are grouped in order to reach a minimum frequency of 5 in each group. E.g. if we expect respectively 1.0, 3.8 and 7.0 blocks with frequency 0, 1 and 2 for events, the three counts are grouped in one group with frequency 1.0+3.8+7.0=11.8. Note that this strategy of grouping is not unique and is likely to weaken the power of the test. Before grouping, the higher class theretical probability is computed as the probability to obtain a count equal to or greater than the max value.

Value

A list with the following objects.

`freq`	frequency table (matrix) giving observed and theoretical (Poisson) frequencies as well as a group number for the chi-square test.
`overdispersion`	the overdispersion coefficient (variance/mean ratio).
`disp.test`	a list giving results of the (over)dispersion test. See the reference Yagouti and al. in the **References** section.
`chisq.test`	a list giving results for the chis-square test of goodness-of-fit to the Poisson distribution.
`tests`	a matrix with the two tests displayed in two rows.

For both tests, the statistic follows a chi-square distribution under the null hypothesis . The list of results contains the statistic `statistic`, the number of degrees of freedom `df` and the p-value `p.value`.

Note

The two tests: (over-)dispersion and chi-square have *one-sided* (upper tail) p-value. In other words, we do not intend to reject when statistics take "abnormally small" values, but only when abnormally large values are met.

Author(s)

Yves Deville

References

See Yagouti A., Abi-Zeid I., Ouarda, T.B.M.J. and B. Bobée (2001), Revue de processus ponctuels et synthèse de tests statistiques pour le choix d'un type de processus *Revue des Sciences de l'Eau*, **1**, pp. 323-361.

See Also

`plot.Rendata`

Examples

```
## na.block influence for Brest data
opar <- par(mfrow = c(2, 2))

bp1 <- barplotRenouv(data = Brest.years, threshold = 30,
         main = "missing periods ignored")
bp2 <- barplotRenouv(data = Brest.years, threshold = 30,
         na.block = 1992, main = "1992 missing")
bp3 <- barplotRenouv(data = Brest.years, threshold = 30,
         na.block = 1991:1993, main ="1991:1993 missing")
bp4 <- barplotRenouv(data = Brest.years, threshold = 30,
         na.block = Brest.years.missing, main = "all missing periods")
```

```
par(opar)

## threshold influence
opar <- par(mfrow = c(2,2))

thresh <- c(30, 35, 40, 50)

for (i in 1:length(thresh)) {
  bp  <- barplotRenouv(data = Brest.years, threshold = thresh[i],
                   na.block = Brest.years.missing,
                   main = paste("threshold =", thresh[i], "cm at Brest"))
}
par(opar)
```

`Brest` *Surge heights at Brest*

Description

Surge heights near high tide at Brest tide gauge station (France), detailed version

Usage

```
Brest
```

Format

The format is: List of 5

- `$info` : List of 6
  - `$name` : chr `"Brest"`
  - `$shortLab` : chr `"Surge Heights at Bres (France)"`
  - `$longLab` : chr `"Surge Heights near high tide,  Brest (France)"`
  - `$varName` : chr `"Surge"`
  - `$varShortLab` : chr `"Surge"`
  - `$varUnit` : chr `"cm"`
- `$describe` : chr "High tide sea surge over 30 cm at Brest(France)..."
- `$OTinfo` : List of 4
  - `$start` : chr POSIXct[1:1], format: `"1845-12-31 23:59:39"`
  - `$end` : chr POSIXct[1:1], format: `"2009-01-01"`
  - `$effDuration` : num 148
  - `$threshold` : num 30
- `$OTdata` : 'data.frame': 1289 obs. of 2 variables:
  - `$date` : POSIXct[1:1289], format: `"1846-01-13 23:59:39" "1846-01-20 23:59:39"` ...

  - $Surge : num [1:1289] 36 60 46 40 33 ...
- $OTmissing : 'data.frame': 43 obs. of 3 variables:
  - $start : POSIXct[1:43], format: "1845-12-31 23:59:39" "1847-12-31 23:59:39" ...
  - $end : POSIXct[1:43], format: "1846-01-03 23:59:39" "1847-01-20 23:59:39" ...
  - $comment : chr [1:43] NA NA NA NA ...

- attr(*, "class")= chr "Rendata"

Details

Data are provided as a list.

- info gives general information about the data
- OTinfo gives general information about the Over the Threshold part of data. The effective duration (effDuration element) is the total duration for the periods with effective measurements.
- OTdata give OT measurements
- OTmissing gives start and end of the missing periods for OT measurements.

Data come from hourly sea levels measured and predicted by the french *Service Hydrogéographique et Océanographique de la Marine* (SHOM). Observed sea levels are available at the url http://www.sonel.org. Data were processed (declustered) by IRSN in order to provide a series of independent surge heights at high tide. Surge height at high tide is defined as the difference between the observed and the predicted maximal sea levels near high tide. A correction was applied to account for trend in the sea-level over the observation period.

The effective duration given in years is defined up to a small fraction of year due to leap years and leap seconds.

Source

http://www.sonel.org

Examples

```
str(Brest)
Brest$OTinfo$start
plot(Brest)
```

Brest.years *Surge heights at Brest partial data*

Description

Surge heights at Brest (France)

Usage

```
Brest.years
```

Format

A data frame with 954 observations on the following 2 variables.

`year` Year e.g; 1980

`Surge` Surge heights above the threshold of 30 cm.

Details

These data are a simplified version of `Brest`. For each surge event only the year is retained as timestamp. Years with missing periods are available as a vector `Brest.years.missing`.

This dataset is useful for testing since similar data are sometimes met in the analyses.

Examples

```
names(Brest.years)
```

Brest.years.missing *Years with missing periods in 'Brest.year' dataset*

Description

Years with missing periods in the 'Brest.years' dataset

Usage

```
Brest.years.missing
```

Format

The format is: int [1:49] 1846 1847 1852 1857 1858 1859 1860 1861 1862 1863 ...

Details

Vector of years containing missing periods in the `Brest.years` dataset. This years should be ignored when computing yearly statistics such as event rates, since time records are lost.

Examples

```
print(Brest.years.missing)
```

`Dunkerque` *Surge heights at Dunkerque*

Description

Surge heights near high tide at Dunkerque tide gauge station (France)

Usage

```
Dunkerque
```

Format

The format is: List of 7

- `$info` : List of 6
  - `$name` : chr `"Dunkerque"`
  - `$shortLab` : chr `"Surge Heights at Dunkerque (France)"`
  - `$longLab` : chr `"Surge Heights near high tide,  Dunkerque (France)"`
  - `$varName` : chr `"Surge"`
  - `$varShortLab` : chr `"Surge"`
  - `$varUnit` : chr `"cm"`
- `$describe` : chr `"High tide sea surge over 30 cm at Dunkerque... "`
- `$OTinfo` : List of 4
  - `$start` : POSIXct[1:1], format: `"1956-01-01"`
  - `$end` : POSIXct[1:1], format: `"2009-01-01"`
  - `$effDuration: num 38.8`
  - `$threshold  : num "30"`
- `$OTdata` : 'data.frame': 740 obs. of 3 variables:
  - `$date` : POSIXct[1:740], format: `"1956-11-27" "1956-12-03"` ...
  - `$Surge` : num [1:740] `67.9 30.9 51.8 30.8 39.8` ...
  - `$comment` : Class 'AsIs' chr [1:740] `"" "" "" ""` ...
- `$OTmissing` : 'data.frame': 83 obs. of 3 variables:
  - `$start` : POSIXct[1:83], format: `"1956-01-01" "1956-08-08"` ...
  - `$end` : POSIXct[1:83], format: `"1956-06-07" "1956-11-03"` ...
  - `$comment`: Class 'AsIs' chr [1:83] `"" "" "" ""` ...
- `$MAXinfo` : 'data.frame' : 1 obs. of 3 variables:
  - `$start` : POSIXct[1:1], format: `"1705-12-31 23:59:39"`
  - `$end` : POSIXct[1:1], format: `"1956-01-01"`

  - $duration : num 250
- $MAXdata :'data.frame': 1 obs. of 4 variables:
  - $block : int 1
  - $date : POSIXct[1:1], format: "1953-02-01"
  - $Surge : num 213
  - $comment : Class 'AsIs' chr "1"

- attr(*, "class")= chr "Rendata"

Details

See Brest and Garonne datasets with the same list structure.

An 'historical' surge of 213 cm was observed on 1953-02-01 and is considered by experts as having a return period of 250 years.

Examples

```
Dunkerque$info
plot(Dunkerque)
```

EM.mixexp *Expectation-Maximisation for a mixture of exponential distributions*

Description

Experimental function for Expectation-Maximisation (EM) estimation

Usage

```
EM.mixexp(x, m = 2)
```

Arguments

x	Sample vector with values >0.
m	Number of mixture components.

Details

The EM algorithm is very simple for exponential mixtures (as well as for many other mixture models).

According to a general feature of EM, this iterative method leads to successive estimates with increasing likelihood but which may converge to a local maximum of the likelihood.

Value

List with

estimate	Estimated values as a named vector.
logL	Vector giving the log-likelihood for successive iterations.
Alpha	Matrix with m columns giving probability weights for successive iterations. Row with number it contains the m probabilities at iteration it.
Theta	Matrix with m columns giving the estimates of the m expectations for the successive iterations

Note

The estimation is done for expectation (inverse rates) but the estimate vector in the result contains rates for compatibility reasons (e.g with exponential).

Author(s)

Yves Deville

See Also

mom.mixexp2 and ini.mixexp2 for "cheap" estimators when m = 2.

Examples

```
set.seed(1234)
x <- rmixexp2(n = 100, prob1 = 0.5, rate2 = 4)
EM.mixexp(x) -> res
res$estimate
matplot(res$Theta, type = "l", lwd = 2,
        xlab = "iteration", ylab = "theta",
        main = "exponential inverse rates")
```

---

expplot *Classical "exponential distribution" plot*

Description

Plot a vector using "exponential distribution" scales

Usage

```
expplot(x,
        plot.pos = "exp",
        rate = NULL,
        labels = NULL,
        mono = TRUE,
        ...)
```

Arguments

x	The vector to be plotted.
plot.pos	Plotting position for points: either "exp" for *expected* ranks or "med" for a *median* rank approximation (see **Details** below).
rate	Rate parameter for one or several "exponential distribution" lines to be plotted
labels	Text to display in legend when "exponential distribution" lines are specified
mono	Monochrome graph?
...	Arguments to be passed to plot.

Details

This plot shows $-\log[1 - F(x)]$ against x where $F(x)$ at point i is taken as $i/(n + 1)$ if plot.pos is "exp", or as the "median rank" approximation $(i - 0.3)/(n + 0.4)$ if plot.pos is "med".

If the data in x is a sample from an exponential distribution, the points should be roughly aligned. However the largest order statistics have high sampling dispersion.

Note

The log scale for y is emulated via the construction of suitable graduations. So be careful when adding graphical material (points, etc) to this graph with functions of the "add to plot" family (points, lines, ...).

The ML estimate of the rate parameter is the inverse of the sample mean.

Author(s)

Yves Deville

See Also

The weibplot function for a classical "Weibull" plot. The interevt is useful to compute interevents (or "interarrivals") that should follow an exponential distribution in the homogeneous Poisson process context.

Examples

```
x <- rexp(200)
expplot(x, rate = 1/mean(x), labels = "fitted")
```

`fgamma` *ML estimation of the Gamma distribution*

Description

Fast Maximum Likelihood estimation of the Gamma distribution.

Usage

```
fgamma(x, check.loglik = FALSE)
```

Arguments

`x`	Sample vector to be fitted. Should contain only positive non-NA values.
`check.loglik`	If `TRUE`, the log-likelihood is recomputed using `dgamma` function with `log = TRUE`. The result is returned as a list element.

Details

The likelihood is concentrated with respect to the scale parameter. The concentrated log-likelihood is a strictly concave function of the shape parameter which can easily maximised numerically.

Value

A list with the following elements

`estimate`	Parameter ML estimates.
`sd`	Vector of (asymptotic) standard deviations for the estimates.
`loglik`	The maximised log likelihood.
`check.loglik`	The checked log-likelihood.
`cov`	The (asymptotic) covariance matrix computed from theoretical or observed information matrix
`info`	The information matrix.

Note

The distribution is fitted by using the `scale` parameter rather than `rate` (inverse of `scale`).

Author(s)

Yves Deville

See Also

`GammaDist` in the **stats** package.

Examples

```
set.seed(9876)
alpha <- 0.06
beta <- rexp(1)
n <- 30
x <- rgamma(n, shape = alpha, scale = beta)
fit <- fgamma(x, check.loglik = TRUE)

## compare with MASS results
if (require(MASS)) {
   fit.MASS <- fitdistr(x, densfun = "gamma")
   rate <- 1 / fit$estimate["scale"]
   est <- c(fit$estimate, rate = rate)
   der <- rate * rate ## derivative of rate w.r.t scale
   sdest <- c(fit$sd, rate = der * fit$sd["scale"])
   tab <- rbind(sprintf(" %10.8f ", est),
                sprintf("(%10.8f)", sdest))
   colnames(tab) <- c("shape", "scale", "rate")
   rownames(tab) <- c("est", "sd")
   noquote(tab)
}
```

---

flomax *ML estimation of the Lomax distribution*

Description

Fast Maximum Likelihood estimation of the Lomax distribution.

Usage

```
flomax(x,
       info.observed = FALSE,
       plot = FALSE,
       scaleData = TRUE)
```

Arguments

x	Sample vector to be fitted. Should contain only positive non-NA values.
info.observed	Should the observed information matrix be used or the expected one be used?
plot	Logical. If TRUE, a plot will be produced showing the derivative of the concentrated log-likelihood, function of the shape parameter.
scaleData	Logical. If TRUE observations in x (which are positive) are divided by their mean value. The results are in theory not affected by this transformation, but scaling the data could improve the estimation in some cases. The log-likelihood plots are shown using the scaled values so the returned estimate of the scale parameter is not the the abscissa of the maximum shown on the plot.

Details

The likelihood is concentrated with respect to the shape parameter. This function is increasing for small values of the scale parameter β. For large β, the derivative of the concentrated log-likelihood tends to zero, and its sign is that of $(1 - \mathrm{CV}^2)$ where CV is the coefficient of variation, computed using n as denominator in the formula for the standard deviation.

The ML estimate does not exist when the sample has a coefficient of variation CV less than 1 and it may fail to be found when CV is greater than yet close to 1.

Value

A list with the following elements

`estimate`	Parameter ML estimates.
`sd`	Vector of (asymptotic) standard deviations for the estimates.
`loglik`	The maximised log likelihood.
`dloglik`	Gradient of the log-likelihood at the optimum. Its two elements should normally be close to zero.
`cov`	The (asymptotic) covariance matrix computed from theoretical or observed information matrix.
`info`	The information matrix.

Note

The estimates are biased for small or medium sized sample. The bias is positive for the shape parameter, thus the estimated shape tends to be larger than the true unknown value.

Fitting a Lomax distribution to an exponential sample might lead to a divergence since the exponential is the limit of a Lomax distibution with large shape and large scale with constant ratio shape/scale. Fitting this distribution to a sample having a coefficient of variation smaller than 1 is not allowed since it should lead to divergence of the estimation.

Author(s)

Yves Deville

References

Johnson N. Kotz S. and N. Balakrishnan *Continuous Univariate Distributions* vol. 1, Wiley 1994.

D. E. Giles, H. Feng & R. T. Godwin (2013) "On the Bias of the Maximum Likelihood Estimator for the Two-Parameter Lomax Distribution" *Com. in Stat. Theory & Methods*. Vol. 42, n. 11, pp. 1934-1950.

See Also

`Lomax` for the Lomax distribution.

Examples

```
## generate sample
set.seed(1234)
n <- 200
alpha <- 2 + rexp(1)
beta <- 1 + rexp(1)
x <- rlomax(n, scale = beta, shape = alpha)
res <- flomax(x, plot = TRUE)

## compare with a GPD with shape 'xi' and scale 'sigma'
xi <- 1 / alpha; sigma <- beta * xi
res.evd <- evd::fpot(x, threshold = 0, model = "gpd")
xi.evd <- res.evd$estimate["shape"]
sigma.evd <- res.evd$estimate["scale"]
beta.evd <- sigma.evd / xi.evd
alpha.evd <- 1 / xi.evd
cbind(Renext = res$estimate, evd = c(alpha = alpha.evd, beta = beta.evd))
```

`fmaxlo` *ML estimation of a 'maxlo' distribution*

Description

Fast Maximum Likelihood estimation of a 'maxlo' distribution

Usage

```
fmaxlo(x,
       info.observed = FALSE,
       plot = FALSE,
       scaleData = TRUE)
```

Arguments

`x`	Sample vector to be fitted. Should contain only positive non-NA values.
`info.observed`	Should the observed information matrix be used or the expected one be used?
`plot`	Logical. If `TRUE`, a plot will be produced showing the derivative of the concentrated log-likelihood, function of the shape parameter.
`scaleData`	Logical. If `TRUE` observations in `x` (which are positive) are divided by their mean value. The results are in theory not affected by this transformation, but scaling the data could improve the estimation in some cases. The log-likelihood plots are shown using the scaled values so the returned estimate of the scale parameter is not the the abscissa of the maximum shown on the plot.

Details

The 'maxlo' likelihood is concentrated with respect to the shape parameter, thus the function to be maximised has only one one scalar argument: the scale parameter β. For large scale β, the derivative of the concentrated log-likelihood tends to zero, and its sign is that of $(\mathrm{CV}^2 - 1)$ where CV is the coefficient of variation, computed using n as denominator in the formula for the standard deviation.

The ML estimate does not exist when the sample has a coefficient of variation CV greater than 1 and it may fail to be found when CV is smaller than yet close to 1.

The information matrix can be computed by noticing that when the r.v. Y follows the 'maxlo' distribution with shape α and scale β the r.v $V := 1/(1 - Y/\beta)$ follows a Pareto distribution with minimum 1 and and shape parameter α. The information matrix involves the second order moment of V.

Value

A list with the following elements

`estimate`	Parameter ML estimates.
`sd`	Vector of (asymptotic) standard deviations for the estimates.
`loglik`	The maximised log-likelihood.
`dloglik`	Gradient of the log-likelihood at the optimum. Its two elements should normally be close to zero.
`cov`	The (asymptotic) covariance matrix computed from theoretical or observed information matrix.
`info`	The information matrix.

Note

The name of the distribution hence also that of the fitting function are still experimental and might be changed.

Author(s)

Yves Deville

See Also

`Maxlo` for the description of the distribution.

Examples

```
## generate sample
set.seed(1234)
n <- 200
alpha <- 2 + rexp(1)
beta <- 1 + rexp(1)
x <- rmaxlo(n, scale = beta, shape = alpha)
res <- fmaxlo(x, plot = TRUE)
```

```
## compare with a GPD with shape 'xi' and scale 'sigma'
xi <- -1 / alpha; sigma <- -beta * xi
res.evd <- evd::fpot(x, threshold = 0, model = "gpd")
xi.evd <- res.evd$estimate["shape"]
sigma.evd <- res.evd$estimate["scale"]
beta.evd <- -sigma.evd / xi.evd
alpha.evd <- -1 / xi.evd
cbind(Renext = res$estimate, evd = c(alpha = alpha.evd, beta = beta.evd))
```

`fweibull` *ML estimation of classical Weibull distribution*

Description

Fast Maximum Likelihood estimation of the classical two parameters Weibull distribution

Usage

```
fweibull(x, info.observed = FALSE, check.loglik = FALSE)
```

Arguments

`x`	Sample vector to be fitted. Should contain only positive non-NA values.
`info.observed`	Should the observed information matrix be used or the expected one be used?
`check.loglik`	If `TRUE`, the log-likelihood is recomputed using `dweibull` function with `log = TRUE`. The result is returned as a list element.

Details

The ML estimates are obtained thanks to a reparametrisation with $\eta = scale^{1/shape}$ in place of `shape`. This allows the maximisation of a one-dimensional likelihood L since the η parameter can be concentrated out of L. This also allows the determination of the *expected* information matrix for $[shape, \eta]$ rather than the usual *observed* information.

Value

A list

`estimate`	Parameter ML estimates.
`sd`	The (asymptotic) standard deviation for estimate.
`cov`	The (asymptotic) covariance matrix computed from theoretical or observed Information matrix.
`eta`	The estimated value for eta.

Note

A well-accepted verdict is that expected information is better than observed information for problems with no missing data as it is the case here. We suspect in the present estimation context that the expected information matrix has often a better condition number than its observed version.

Author(s)

Yves Deville

See Also

weibplot for Weibull plots.

Examples

```
n <- 1000
shape <- 2 * runif(1)
x <- 100 * rweibull(n, shape = 0.8, scale = 1)
res <- fweibull(x)

## compare with MASS
if (require(MASS)) {
   res2 <- fitdistr(x , "weibull")
   est <- cbind(res$estimate, res2$estimate)
   colnames(est) <- c("Renext", "MASS")
   loglik <- c(res$loglik, res2$loglik)
   est <- rbind(est, loglik)
   est
}

## Weibull plot
weibplot(x,
         shape = c(res$estimate["shape"], res2$estimate["shape"]),
         scale = c(res$estimate["scale"], res2$estimate["scale"]),
         labels = c("Renext 'fweibull'", "MASS 'fitdistr'"),
         mono = TRUE)
```

Garonne *Flow of the french river La Garonne*

Description

Flow of the french river La Garonne at le Mas d'Agenais

Usage

```
Garonne
```

Format

The format is: List of 7

- $info : List of 6
  - $name : chr "Garonne"
  - $shortLab : chr "La Garonne at Le Mas d'Agenais"
  - $longLab : chr "River flow of La Garonne at Le Mas d'Agenais"
  - $varName : chr "Flow"
  - $varShortLab : chr "Flow"
  - $varUnit : chr "m3/s"
- $describe : chr "Flow of the french river La Garonne ..."
- $OTinfo :List of 4
  - $start : POSIXct[1:1], format: "1913-01-01"
  - $end : POSIXct[1:1], format: "1978-01-01"
  - $effduration : num 65
  - $threshold : num 2500
- $OTdata : 'data.frame': 151 obs. of 3 variables:
  - $date : POSIXct[1:151], format: "1913-04-08" "1913-04-25" ...
  - $Flow : num [1:151] 2600 2800 2700 4579 3400 ...
  - comment : Class 'AsIs' chr [1:151] "" "" "" "" ...
- $OTmissing : NULL
- $MAXinfo :'data.frame': 1 obs. of 3 variables:
  - $start : POSIXct[1:1], format: "1769-12-31 23:59:39"
  - $end : POSIXct[1:1], format: "1913-01-01"
  - $duration : num 143
- $MAXdata :'data.frame': 12 obs. of 4 variables:
  - $block : num [1:12] 1 1 1 1 1 1 1 1 1 1 ...
  - date : POSIXct[1:12], format: NA NA ...
  - $Flow : num [1:12] 7500 7400 7000 7000 7000 6600 6500 6500 6400 6300 ...
  - $comment : class 'AsIs' chr [1:12] "1 (1875)" "2 (1770)" "3 (1783)" "4 (1855)" ...

- attr(*, "class")= chr "Rendata"

Details

The data concern the french river *La Garonne* at the gauging station named *Le Mas d'Agenais* where many floods occurred during the past centuries.

The data consist in OT data and historical data. The variable is the river flow in cube meter per second $(\mathrm{m}^3/\mathrm{s})$ as estimated from the river level using a rating curve. The precision is limited and many ties are present among the flow values.

The OT data or "OTdata" contain flows values over the threshold $u = 2500\,\mathrm{m}$ for the 65 years period 1913-1977. The historical data or "MAXdata" is simply the $r = 12$ largest flows for the period of 143 years 1770-1912. The exact dates of these events are not known with precision but the years are known and given as comments.

Source

The data were taken from the book by Miquel.

References

Miquel J. (1984) *Guide pratique d'estimation des probabilités de crues*, Eyrolles (coll. EDF DER).

Parent E. and Bernier J. (2003) Bayesian POT modeling for Historical data. *Journal of Hydrology* vol. 274, pp. 95-108.

Examples

```
plot(Garonne)
```

`gof.date` *Goodness-of-fit for the distribution of dates*

Description

Goodness-of-fit diagnostics for the distribution of event dates in a (assumed) Poisson process

Usage

```
gof.date(date,
         start = NULL,
         end = NULL,
         plot = TRUE,
         main = NULL,
         skip = NULL,
         plot.type = "skip")
```

Arguments

`date`	Object of class `POSIXct` (or that can be coerced to this class) giving the dates to be tested. Must be in strictly increasing order.
`start`	The beginning of the interval, a `POSIXct` object. If `NULL`, the first event in `date` is used.
`end`	Object of class `POSIXct` the end of the interval. If `NULL`, the last event in `date` is used.
`plot`	Should a plot be shown?
`main`	Character giving the main title of the plot. The default `NULL` stands for a default main describing the period.
`skip`	Optional data.frame with columns `start` and `end` indicating start and end of skipped periods. The two columns need to be coerced to POSIXct objects. They can be POSIXct or character with POSIX datetime format.

`plot.type` Character indicating the type of plot to produce when a `skip` data.frame is given. With `plot.type = "skip"` the plot shows missing periods as greyed rectangles and the displays the results of a Kolmogorov-Smirnov (KS) test performed on the events. For the `"omit"` case the missing periods are collapsed into vertical lines on the plot and the displayed results are for an "effective" KS test of uniformity performed omitting the missing periods.

Details

In the homogeneous Poisson process, events occur on a time interval in a uniform fashion. More precisely, for a given time interval the distribution of the event dates conditional to their number n is the distribution of the order statistics of a sample of size n of the uniform distribution on this interval.

When the interval has limits taken at events the uniformity statement remains true, but for *inner* events. This behaviour is met when `start` and `end` are not given and taken as the first and last events in `date`.

Value

A list

`effKS.statistic, KS.statistic`
Kolmogorov-Smirnov global test statistic for uniformity (bilateral test) omitting slipped periods or not.

`effKS.pvalue, KS.pavalue`
Critical probability in the KS test omitting skipped periods or not.

`effnevt, nevt` Number of events omitting skipped periods or not.

`effduration, duration`
Effective duration i.e. total duration of non-skipped periods. In years, omitting skipped periods or not.

`effrate, rate` Occurrence rate in **number of events by year**, omitting skipped periods or not.

`effduration, duation`
Total duration in **years**, omitting missing periods or not.

`noskip` Data.frame object giving indications on the periods that are NOT skipped over (hence usually non-missing periods). These are : `start`, `end` (POSIX), `duration` (in years) `rate` (in number of events by year) and Kolmogorov test statistic and p-value. This data.frame is only available when a suitable `skip` has been given.

When the number of events corresponding to the indications of args is `0`, the function returns `NULL` with a warning. When the number of events is less than `6` a warning is shown.

Warning

When skipped periods exist the number of events, duration, rate the global KS test must be computed by omitting the skipped periods in the duration and retaining only valid interevents. The indication given in `nevt` `rate` and `duration` should be used only when no skipped period exist (`skip = NULL` on input) and replaced by `effnevt`, `effrate` and `effduration` otherwise.

Note

In practical contexts missing periods are often met in the datasets. The diagnostic should therefore be applied on *every period with no missing data*. Even if the event dates seem reasonably uniform, it is a good idea to check that the rates do not differ significantly over intervals.

When some events are missing and no suitable information is given via the `skip` argument, the global `rate`, `KS.statistic` and `KS.pvalue` are of little interest. Yet the graph might be instructive.

Author(s)

Yves Deville

See Also

`interevt` function for the determination of interevents ans subsequent diagnostics.

Examples

```
## Use "Brest" dataset
## simple plot. Kolmogorov-Smirnov is not useful
gof1 <- gof.date(date = Brest$OTdata$date)

## consider missing periods. Much better!
gof2 <- gof.date(date = Brest$OTdata$date,
         skip = Brest$OTmissing,
         start = Brest$OTinfo$start,
         end = Brest$OTinfo$end)

print(gof2$noskip)

## Second type of graph
gof3 <- gof.date(date = Brest$OTdata$date,
         skip = Brest$OTmissing,
         start = Brest$OTinfo$start,
         end = Brest$OTinfo$end,
         plot.type = "omit")

## non-skipped periods at Brest
ns <- skip2noskip(skip = Brest$OTmissing,
                  start = Brest$OTinfo$start,
                  end = Brest$OTinfo$end)

## say 9 plots/diagnostics
oldpar <- par(mar = c(3, 4, 3, 2), mfcol = c(3, 3))

for (i in 1:9) {
  GOF <- gof.date(date = Brest$OTdata$date,
           start = ns$start[i],
           end = ns$end[i])
}

par(oldpar)
```

gofExp.test *Goodness-of-fit test for exponential distribution*

Description

Bartlett's goodness-of-fit test for exponential distribution

Usage

```
gofExp.test(x)
```

Arguments

x	sample

Value

A list with elements

statistic	Statistic
p.value	Critical value

Author(s)

Yves Deville

References

See Yagouti A., Abi-Zeid I., Ouarda, T.B.M.J. and B. Bobée (2001), Revue de processus ponctuels et synthèse de tests statistiques pour le choix d'un type de processus *Revue des Sciences de l'Eau*, **1**, pp. 323-361.

See Also

Among other goodness-of-fit tests ks.test in the stats package. See expplot for a graphical diagnostic.

Examples

```
## a sample of size 30
x <- rexp(30)
res <- gofExp.test(x)

## ns samples: p.values should look as uniform on (0, 1)
ns <- 100
xmat <- matrix(rexp(30*ns), nrow = ns, ncol = 30)
p.values <- apply(xmat, 1, function(x) gofExp.test(x)$p.value)
plot(sort(p.values), type = "p", pch = 16)
```

`ini.mixexp2` *Simple estimation for the mixture of two exponential distributions*

Description

Compute a simple (preliminary) estimation for the tree parameters of the mixture of two exponential distributions

Usage

```
ini.mixexp2(x, plot = FALSE)
```

Arguments

`x`	Sample: numerical vector with elements `>0`.
`plot`	Should a graphic be displayed?

Details

This function gives estimators using several methods if necessary. The goal is to find the rates `rate1`, `rate2` and the mixing probability `prob1` with the 'feasibility' constraints `0 < rate1 < rate2` and `0 < prob1 < 1`.

First the method of moments is used. If the estimates are feasible they are returned with `method = "moments"`. If not, the estimates are derived using two linear regressions. A regression without constant using only the smallest values gives an estimator of the mean rate. A regression using only the largest values gives `rate1` and `prob1`. Yet the constraints must be fulfilled. If they are, the estimates are returned (together with `method = "Hreg"` suggesting a cumulative hazard regression). If not, a (poor) default estimate is returned with `method = "arbitrary"`.

Value

A list

`estimate`	A vector with named elements `"prob1"`, `"rate1"` and `"rate2"`.
`method`	The method that really produced the estimators.

Note

The method of moments is implemented in `mom.mixexp2`. Further investigations are needed to compare the estimators (moments or Hreg) and select the best strategy.

Note that this function returns the estimate within a list and no longer as a vector with named elements as was the case before.

Author(s)

Yves Deville

See Also

See MixExp2, mom.mixexp2.

Examples

```
set.seed(1234)
x <- rmixexp2(n = 100, prob1 = 0.5, rate2 = 4)
res <- ini.mixexp2(x, plot = TRUE)
```

---

interevt *Interevents (or interarrivals) from events dates*

Description

Compute intervent durations from events dates

Usage

```
interevt(date,
         skip = NULL, noskip = NULL)
```

Arguments

date	A POSIXct vector containing the date(time) of the events.
skip	A data.frame containing two POSIXct columns start and end describing the periods to skip over.
noskip	A data.frame like skip but where the periods define the NON skipped part of the events.

Details

Interevents are the time differences between successive dates. When the date argument contains occurrence times T_i for successive events of an homogeneous Poisson process, interevents $T_i - T_{i-1}$ are mutually independent with the same exponential distribution.

When some time intervals are skipped independently from the event point process, we may consider the interevents $T_i - T_{i-1}$ between two non-skipped events such that the time interval (T_{i-1}, T_i) does not contains any skipped interval. These interevents still are mutually independent with the same exponential distribution. When skip or noskip is not NULL the computation therefore only retains couples of two successive datetimes "falling" in the same non-skipped period, which number can therefore be associated with the interevent.

Value

A list mainly containing a `interevt` data.frame.

`interevt`	Data.frame. Each row describes a retained interevent through a `period` integer giving the "noskip" period, a `start` and `end` POSIX and a `duration` in **days**.
`noskip`	Only hen `skip` or `noskip` args have been given. A data.frame containing broadly the same information as the `noskip` arg is it was given or the information deduced from the `skip` arg if given.
`axis`	When needed, a list with some material to build an axis with uneven ticks as in the `gof.date` with `skip.action = "omit"`.

Note

Only one of the two arguments `skip` and `noskip` should be given in the call. In each case, the rows of the returned data.frame objects describe periods in chronological order. That is: `start` at row `2` must be after the `end` value of row `1` and so on.

Note that there are usually less interevents than dates since two successive dates will be retained for an interevent only when they are not separated by missing period. As a limit case, there can be no interevents if the `noskip` periods contain only one date from the `date` vector.

Author(s)

Yves Deville

See Also

`gof.date` for goodness-of-fit diagnostics for dates of events `expplot` for diagnostics concerning the exponential distribution.

Examples

```
## Use Brest data
ie <- interevt(date = Brest$OTdata$date, skip = Brest$OTmissing)

expplot(ie$interevt$duration, rate = 1 / mean(ie$interevt$duration),
  main = "No threshold")

## keep only data over a threshold
ind1 <- Brest$OTdata$Surge >= 35
ie1 <- interevt(Brest$OTdata$date[ind1], skip = Brest$OTmissing)
expplot(ie1$interevt$duration, main = "Threshold = 35")

## increase threshold
ind2 <- Brest$OTdata$Surge >= 55
ie2 <- interevt(date = Brest$OTdata$date[ind2], skip = Brest$OTmissing)
expplot(ie2$interevt$duration, main = "Threshold = 55 cm")
```

`logLik.Renouv` *Log-likelihood of a "Renouv" object.*

Description

Log-likelihood, AIC, BIC and number of observations of an object of class "Renouv".

Usage

```
## S3 method for class 'Renouv'
AIC(object, ..., k = 2)
## S3 method for class 'Renouv'
BIC(object, ...)
## S3 method for class 'Renouv'
logLik(object, ...)
## S3 method for class 'Renouv'
nobs(object, ...)
```

Arguments

`object`	Object of class `"Renouv"`.
`k`	See `AIC`.
`...`	Other arguments passed to the default method.

Caution

Comparing log-likelihoods, AIC or BIC for different `Renouv` objects makes sense only when these share the same data and the same threshold.

Note

`logLik`, `AIC` and `BIC` can be used with an object of class `"Renouv"` which makes use of historical data. In this case, the number of observations may be misleading since a single historical observation may concern dozens of years and thus have a much greater impact on the estimation of the tail than an "ordinary" observation.

Author(s)

Yves Deville

See Also

The `AIC`, `nobs` generic functions.

Lomax *Lomax distribution*

Description

Density function, distribution function, quantile function and random generation for the Lomax distribution.

Usage

```
dlomax(x, scale = 1.0, shape = 4.0, log = FALSE)
plomax(q, scale = 1.0, shape = 4.0, lower.tail = FALSE)
qlomax(p, scale = 1.0, shape = 4.0)
rlomax(n, scale = 1.0, shape = 4.0)
```

Arguments

x, q	Vector of quantiles.
p	Vector of probabilities.
n	Number of observations.
scale, shape	Shift and shape parameters. Vectors of length > 1 are not accepted.
log	Logical; if TRUE, the log density is returned.
lower.tail	Logical; if TRUE (default), probabilities are Pr$[X <= x]$, otherwise, Pr$[X > x]$.

Details

The Lomax distribution function with shape $\alpha > 0$ and scale $\beta > 0$ has survival function

$$S(y) = [1 + y/\beta]^{-\alpha} \qquad (y > 0)$$

This distribution has increasing hazard and decreasing mean residual life (MRL). The coefficient of variation decreases with α, and tends to 1 for large α. The default value $\alpha = 4$ corresponds to CV $= \sqrt{2}$.

Value

dlomax gives the density function, plomax gives the distribution function, qlomax gives the quantile function, and rlomax generates random deviates.

Note

This distribution is sometimes called *log-exponential*. It is a special case of Generalised Pareto Distibution (GPD) whith positive shape $\xi > 0$, scale σ and location $\mu = 0$. The Lomax and GPD parameters are related according to

$$\alpha = 1/\xi, \qquad \beta = \sigma/\xi.$$

The Lomax distribution can be used in POT to describe exceedances following GPD with shape $\xi > 0$ thus with decreasing hazard and increasing Mean Residual Life.

Note that the exponential distribution with rate ν is the limit of a Lomax distibution having large scale β and large shape α, with the constraint on the shape/scale ratio $\alpha/\beta = \nu$.

References

Johnson N. Kotz S. and N. Balakrishnan *Continuous Univariate Distributions* vol. 1, Wiley 1994.

Lomax distribution in Wikipedia

See Also

`flomax` to fit the Lomax distibution by Maximum Likelihood.

Examples

```
shape <- 5; scale <- 10
xl <- qlomax(c(0.00, 0.99), scale = scale, shape = shape)
x <- seq(from = xl[1], to = xl[2], length.out = 200)
f <- dlomax(x, scale = scale, shape = shape)
plot(x, f, type = "l", main = "Lomax distribution density")
F <- plomax(x, scale = scale, shape = shape)
plot(x, F, type ="l", main ="Lomax distribution function")
```

Maxlo *'maxlo' distribution*

Description

Density function, distribution function, quantile function and random generation for the 'maxlo' distribution.

Usage

```
dmaxlo(x, scale = 1.0, shape = 4.0, log = FALSE)
pmaxlo(q, scale = 1.0, shape = 4.0, lower.tail = FALSE)
qmaxlo(p, scale = 1.0, shape = 4.0)
rmaxlo(n, scale = 1.0, shape = 4.0)
```

Arguments

`x, q`	Vector of quantiles.
`p`	Vector of probabilities.
`n`	Number of observations.
`scale, shape`	Shift and shape parameters. Vectors of length > 1 are not accepted.
`log`	Logical; if `TRUE`, the log density is returned.
`lower.tail`	Logical; if `TRUE` (default), probabilities are $\Pr[X <= x]$, otherwise, $\Pr[X > x]$.

Details

The 'maxlo' distribution function with shape $\alpha > 0$ and scale $\beta > 0$ is a special case of Generalised Pareto (GPD) with *negative shape* $\xi < 0$ and location at zero. This is the finite upper endpoint case of the GPD. Its name is TEMPORARY and was chosen to suggest some form of symmetry with respect to the Lomax distribution.

The survival function is

$$S(y) = [1 - y/\beta]^{\alpha} \qquad 0 < y < \beta$$

This distribution has a coefficient of variation smaller than 1.

Value

`dmaxlo` gives the density function, `pmaxlo` gives the distribution function, `qmaxlo` gives the quantile function, and `rmaxlo` generates random deviates.

Note

The 'maxlo' and GPD parameters are related according to

$$\alpha = -1/\xi, \qquad \beta = -\sigma/\xi.$$

where σ is the scale parameter of the GPD. Since only GPD with $\xi > -0.5$ seem to be used in practice, this distribution should be used with $\alpha > 2$.

This distribution can be used in POT to describe bounded exceedances following GPD with shape $\xi < 0$. The scale parameter β then represents the upper end-point of the exceedances, implying the finite upper end-point $u + \beta$ for the levels, where u is the threshold. It can be used in `Renouv` with a fixed scale parameter, thus allowing a control of the upper end-point.

This distribution is simply a rescaled version of a beta distribution and also a rescaled version of a Kumaraswamy distribution. The name "maxlo" is used here to suggest a form of symmetry to Lomax distribution.

See Also

`fmaxlo` to fit such a distibution by Maximum Likelihood.

Examples

```
xs <- rmaxlo(500, shape = 2.2, scale = 1000)
hist(xs, main = "'maxlo' distribution"); rug(xs)

xs <- rmaxlo(500, shape = 4, scale = 1000)
hist(xs, main = "'maxlo' distribution"); rug(xs)

x <- seq(from = -10, to = 1010, by = 2)
plot(x = x, y = dmaxlo(x, shape = 4, scale = 1000),
     type = "l", ylab = "dens",
     col = "orangered", main = "dmaxlo and dgpd")
abline(h = 0)
lines(x = x, y = dgpd(x, shape = -1/4, scale = 250),
     type = "l",
```

```
        col = "SpringGreen3", lty = "dashed")
```

MixExp2 *Mixture of two exponential distributions*

Description

Probability functions associated to the mixture of two exponential distributions.

Usage

```
dmixexp2(x, prob1,
         rate1 = 1.0, rate2 = rate1 + delta, delta,
         log = FALSE)
pmixexp2(q, prob1,
         rate1 = 1.0, rate2 = rate1 + delta, delta,
         log = FALSE)
qmixexp2(p, prob1,
         rate1 = 1.0, rate2 = rate1 + delta, delta)
rmixexp2(n, prob1,
         rate1 = 1.0, rate2 = rate1 + delta, delta)
hmixexp2(x, prob1,
         rate1 = 1.0, rate2 = rate1 + delta, delta)
Hmixexp2(x, prob1,
         rate1 = 1.0, rate2 = rate1 + delta, delta)
```

Arguments

x, q	Vector of quantiles.
p	Vector of probabilities.
n	Number of observations.
log	Logical; if TRUE, the log density is returned.
prob1	Probability weight for the "number 1" exponential density.
rate1	Rate (inverse expectation) for the "number 1" exponential density.
rate2	Rate (inverse expectation) for the "number 2" exponential density. Should in most cases be > rate1. See *Details*.
delta	Alternative parametrisation delta = rate2 - rate1.

Details

The density function is the mixture of two exponential densities

$$f(x) = \alpha_1 \lambda_1 \, e^{-\lambda_1 x} + (1 - \alpha_1) \lambda_2 \, e^{-\lambda_2 x} \qquad x > 0$$

where α_1 is the probability given in `prob1` while λ_1 andλ_2 are the two rates given in `rate1` and `rate2`.

A 'naive' identifiability constraint is

$$\lambda_1 < \lambda_2$$

i.e. `rate1 < rate2`, corresponding to the simple constraint `delta > 0`. The parameter `delta` can be given instead of `rate2`.

The mixture distribution has a decreasing hazard, increasing Mean Residual Life (MRL) and has a thicker tail than the usual exponential. However the hazard, MRL have a finite non zero limit and the distribution behaves as an exponential for large return levels/periods.

The quantile function is not available in closed form and is computed using a dedicated numerical method.

Value

`dmiwexp2`, `pmiwexp2`, `qmiwexp2`, evaluates the density, the distribution and the quantile functions. `dmixexp2` generates a vector of `n` random draws from the distribution. `hmixep2` gives hazard rate and `Hmixexp2` gives cumulative hazard.

Examples

```
rate1 <- 1.0
rate2 <- 4.0
prob1 <- 0.8
qs <- qmixexp2(p = c(0.99, 0.999), prob1 = prob1,
               rate1 = rate1, rate2 = rate2)
x <- seq(from = 0, to = qs[2], length.out = 200)
F <- pmixexp2(x, prob1 = prob1, rate1 = rate1, rate2 = rate2)
plot(x, F, type = "l", col = "orangered", lwd = 2,
     main = "Mixexp2 distribution and quantile for p = 0.99")
abline(v = qs[1])
abline(h = 0.99)
```

---

mom.mixexp2 *Moment estimation for the mixture of two exponential distributions*

Description

Compute the moment estimation for the tree parameters of the mixture of two exponential distributions

Usage

```
mom.mixexp2(x)
```

Arguments

x	Sample. Vector containing values >0.

Details

The three parameters (probability and the two rates) are computed from the first three moments (theoretical and sample). It can be shown that the inverse rates are obtained solving a quadratic equation. However the roots can be negative or complex and the estimates are not valid ones.

Value

A list with elements

estimate	A vector with named elements "prob1", "rate1" and "rate2". When the moment estimators are not valid (negative or complex rates), a vector of three NA is returned.
method	Character "moments".

Note

The theoretical coefficient of variation (CV) of a mixture of two exponential distributions always exceed 100%. When the sample CV is over 100%, no valid estimates exist since the two first moments can not be matched.

Author(s)

Yves Deville

References

Paul R. Rider. The Method of Moments Applied to a Mixture of Two Exponential Distributions. *Ann. Math. Statist.* Vol. 32, Number 1 (1961), 143-147.

See Also

See ini.mixexp2 for a more versatile initial estimation.

Examples

```
x <- rmixexp2(n = 100, prob1 = 0.5, rate1 = 1.0, rate2 = 3.0)
est <- mom.mixexp2(x)
```

mom2par *Parameters from moments*

Description

Compute parameters from (theoretical) moments

Usage

```
mom2par(densfun = "exponential",
        mean,
        sd = NA)
```

Arguments

densfun	Name of the distribution. This can be at present time: "exponential", "weibull", "gpd", "gamma", "negative binomial".
mean	Theoretical mean (expectation) of the distribution. Can be a vector, in which case the parameters will be vectors.
sd	Standard deviation.

Details

For some distributions like Weibull, it is necessary to find a numerical solution since the parameters have no closed form expression involving the moments.

Value

A named list containing the parameters values e.g. with names shape and scale. When mean or sd is vector the list contains vectors.

Note

The name of the formal argument densfun is for compatibility with fitdistr from the MASS package. However, unlike in fitdistr this formal can not be given a density value, i.e. an object of the class "function" such as dnorm.

Author(s)

Yves Deville

Examples

```
## Weibull
mom2par(densfun = "weibull", mean = 1, sd = 2)
## Genrealised Pareto
mom2par(densfun = "gpd", mean = 1, sd = 2)
## Gamma
mom2par(densfun = "gamma", mean = 1:10, sd = 1)
```

NBlevy *Negative Binomial Levy process*

Description

Negative Binomial Lévy process estimation from partial observations (counts)

Usage

```
NBlevy(N,
       gamma = NA,
       prob = NA,
       w = rep(1, length(N)),
       sum.w = sum(w),
       interval = c(0.01, 1000),
       optim = TRUE,
       plot = FALSE, ...)
```

Arguments

N	Vector of counts, one count by time period.
gamma	The gamma parameter if known (NOT IMPLEMENTED YET).
prob	The prob parameter if known (NOT IMPLEMENTED YET).
w	Vector of time length (durations).
sum.w	NOT IMPLEMENTED YET. The effective duration. If sum.w is strictly inferior to sum(w), it is to be understood that missing periods occur within the counts period. This can be taken into account with a suitable algorithm (Expectation Maximisation, etc.)
interval	Interval giving min and max values for gamma.
optim	If TRUE a one-dimensional optimisation is used. Else the zero of the derivative of the (concentrated) log-likelihood is searched for instead.
plot	Should a plot be drawn? MAY BE REMOVED IN THE FUTURE.
...	Arguments passed to plot.

Details

The vector $\mathbf{N}$ contains counts for events occurring on non-overlapping time periods with lengths given in $\mathbf{w}$. Under the NB Lévy process assumptions, the observed counts (i.e. elements of $\mathbf{N}$) are independent random variables, each following a negative binomial distribution. The size parameter r_k for N_k is $r_k = \gamma w_k$ and the probability parameter p is prob. The vector $\boldsymbol{\mu}$ of the expected counts has elements

$$\mu_k = \mathrm{E}(N_k) = \frac{1-p}{p}\,\gamma\,w_k$$

The parameters γ and p (prob) are estimated by Maximum Likelihood using the likelihood concentrated with respect to the prob parameter.

Value

A list with the results

`estimate`	Parameter estimates.
`sd`	Standard deviation for the estimate.
`score`	Score vector at the estimated parameter vector.
`info`	Observed information matrix.
`cov`	Covariance matrix (approx.).

Note

The Negative Binomial Lévy process is an alternative to the Homogeneous Poisson Process when counts are subject to overdispersion. In the NB process, all counts share the same index of dispersion (variance/expectation ratio), namely `1/prob`. When `prob` is close to 1, the counts are nearly Poisson-distributed.

Author(s)

Yves Deville

References

Podgórsky K. (2008) *Negative Binomial Lévy process -genesis, properties and applications*. Lund University, Sweden.

See Also

`NegBinomial` for the negative binomial distribution, `glm.nb` from the MASS package for fitting Generalised Linear Model of the negative binomial family.

Examples

```
## known parameters
nint <- 100
gam <- 6; prob <- 0.20

## draw random w, then the counts N
w <- rgamma(nint, shape = 3, scale = 1/5)
N <- rnbinom(nint, size = w * gam, prob = prob)
mu <- w * gam * (1 - prob) / prob
Res <- NBlevy(N = N, w = w)

## Use example data 'Brest'

## compute the number of event and duration
## of the non-skipped periods
gof1 <- gof.date(date = Brest$OTdata$date,
                 skip = Brest$OTmissing,
                 start = Brest$OTinfo$start,
                 end = Brest$OTinfo$end,
```

```
                plot.type = "omit")
ns1 <- gof1$noskip
## fit the NBlevy
fit1 <- NBlevy(N = ns1$nevt, w = ns1$duration)

## use a higher threshold
OT2 <- subset(Brest$OTdata, Surge > 50)
gof2 <- gof.date(date = OT2$date,
                skip = Brest$OTmissing,
                start = Brest$OTinfo$start,
                end = Brest$OTinfo$end,
                plot.type = "omit")
ns2 <- gof2$noskip
## the NBlevy prob is now closer to 1
fit2 <- NBlevy(N = ns2$nevt, w = ns2$duration)

c(fit1$prob, fit2$prob)
```

`OTjitter` *Add a small amount of noise to a numeric vector*

Description

Add a small amount of noise to a numeric vector keeping all the values above the given threshold.

Usage

```
OTjitter(x, threshold = NULL)
```

Arguments

`x`	The numeric vector to which *jitter* should be added.
`threshold`	A threshold above which all elements of the modified vector must stay.

Value

A vector with the same length and nearly the same values as `x`. As in `jitter`, a small amount of noise is added to each value of `x`. The noise level is adjusted so that every noisy value remains above the specified threshold. When the a value is very close to the threshold, only a very small amount of negative noise can be added.

Note

The aim of this function is to remove possible ties in experimental OT data. Ties cause problems or warnings in some goodness-of-fit tests such as Kolmogorov-Smirnov.

Author(s)

Yves Deville

See Also

jitter

Examples

```
## Garonne data (heavily rounded)
x <- Garonne$OTdata$Flow
min(x)
xmod <- OTjitter(x, threshold = 2500)
length(x)
nlevels(as.factor(x))
nlevels(as.factor(xmod))
max(abs(x-xmod))
```

---

plot.Rendata *Plot a Rendata object*

Description

Plot 'Rendata' datasets with OT and historical data

Usage

```
    ## S3 method for class 'Rendata'
plot(x,
     textOver = quantile(x$OTdata[, x$info$varName], probs = 0.99),
     showHist = TRUE,
                ...)
```

Arguments

x	Rendata object i.e. a list object as read with the readXML function.
textOver	Mark values of the variable in the OTdata part of x. Values above the textOver value (if any) will be marked with the character version of the block, typically a year
showHist	If TRUE, the historical periods (is any) are shown on the plot.
...	further args to be passed to plot function.

Details

The plot shows the main data of the object x (the OTdata part) as well as historical data MAXdata or OTSdata if any. Different colours are used on the background. This function is not intended to produce nice plots to be printed.

Note

This function is mainly a companion function of readXML. Its goal is to check the content of the data read.

Author(s)

Yves Deville

See Also

readXML

Examples

```
if (require(XML)) {
   ## use 'index.xml' file shipped with Renext
   dir1 <- system.file("Rendata", package = "Renext")
   BrestNew <- readXML(name = "Brest", dir = dir1)
   plot(BrestNew)
   GaronneNew <- readXML(name = "Garonne", dir = dir1)
   plot(GaronneNew)
   test1 <- readXML(name = "test1", dir = dir1)
   plot(test1)
}
```

---

plot.Renouv *Plot an object of class "Renouv"*

Description

Plot an object of class "Renouv". The plot is a return level plot with some supplementary elements to display historical data.

Usage

```
## S3 method for class 'Renouv'
plot(x,
    pct.conf = x$pct.conf,
    show = list(OT = TRUE, quant = TRUE, conf = TRUE,
                MAX = TRUE, OTS = TRUE),
    mono = TRUE,
    predict = FALSE,
    par = NULL,
    legend = TRUE,
    legendEnvir = NULL,
    label = NULL,
    problim = NULL,
    Tlim = NULL,
    main = NULL, xlab = "periods", ylab = "level",
    ...)
## S3 method for class 'Renouv'
lines(x,
    pct.conf = x$pct.conf,
```

```
    show = NULL,
    mono = TRUE,
    predict = FALSE,
    par = NULL,
    legend = FALSE,
    legendEnvir = NULL,
    label = NULL,
    ...)
```

Arguments

x	Object of class "Renouv".
pct.conf	Percents for confidence limits (lower and upper). These levels should be found within those computed in the object x. By default, all computed levels will be used.
show	A list with named elements specifying which parts of the return level plot must be drawn. Element OT is for the the sample points (Over the Threshold data), quant is for the quantile curve (or Return Level curve), conf is for the confidence limits. Finally MAX and OTS are for the two possible types of historical data. When the element conf is TRUE, only the percent levels given in pct.conf are drawn. Moreover, the levels not already computed in the object given in x will be drawn only if the predictions are recomputed with predict set to TRUE.
mono	Logical, TRUE for a monochrome plot.
predict	Logical. When TRUE, predictions are re-computed from the model before plotting. One effect is that the points used to draw the curves are designed to cover the whole range (if specified by the user). One other effect is that the confidence limits are recomputed in order to include the percent levels given on entry.
par	A list such as returned by the RLpar function.
legend	Logical. If TRUE, a legend is built and drawn on the graph.
legendEnvir	An optional environment in which legend information will be stored for later use. This information will be shown by using RLlegend.show. This environment is used only when legend is FALSE, and the legend is drawn directly otherwise.
label	A character label used to build the labels used in the legend.
problim	Limits for the x-axis in probability scale. Can be used as an alternative to Tlim.
Tlim	Limits for the x-axis in return period scale. The values are given as a numeric vector of length 2, containing values ≥ 1. The first element (minimal return period can be 0 in which case it will be replaced by a very small positive value.
xlab	Label of the x-axis (time periods, with log scale).
main	Main title (character).
ylab	Label of the y-axis (labels).
...	Other arguments passed to the default plot function e.g., ylim to adjust the y-axis.

Details

Historical data embeded in the x object are list(MAX = TRUE)

- If the MAX element is TRUE and if x embeds historical data of type MAX, then these will be shown with a symbol differing from the one for ordinary points.
- If OTS element is TRUE and is x embeds historical data of type OTS, then these will be shown with a symbol differing from the one for ordinary points. An exception is when one or several OTS block have no data. Then each such block is shown as an horizontal segment; its right end-point shows the effective duration of the block and the ordinate shows the OTS threshold for this block. No data exceeded the threshold within the block.

This function acts on a list variable named .RLlegend in the legendEnvir environment. This variable is used to build legends for plots produced with multiple commands. See the RLlegend help page.

Value

No value returned.

Caution

Remaind that the methods plot and lines may change the value of the variable .RLlegend in the environment legendEnvir. This variable describes the material to be used in the legend at the next call of RLlegend.show.

Note

The return level plot is of exponential type i.e. uses a log-scale for return periods. This contrasts with the Gumbel plot which is also used in similar contexts.

Author(s)

Yves Deville

See Also

The RLlegend page for the legend construction and RLpar to specify the grahical parameters (colors, line types, ...) of the elements.

Examples

```
## two fits for the Garonne data
fit.exp <- Renouv(x = Garonne, plot = FALSE)
fit.gpd <- Renouv(x = Garonne, distname.y = "gpd", plot = FALSE)

## simple plot (legend is TRUE here)
plot(fit.exp,
     main = "Two fits overlapped",
     label = "",
     ## Tlim = c(1, 5000),
```

```
     predict = TRUE)

## Now try 'lines' and RLlegend.xxx functions
plot(fit.exp,
     main = "Fancy legend",
     show = list(OT = FALSE, quant = FALSE, conf = FALSE,
                 OTS = FALSE, MAX = FALSE),
     legend = FALSE,
     Tlim = c(1, 5000))

legEnv <- new.env()
RLlegend.ini(x = "bottomright", bg = "lightyellow", envir = legEnv) ## initialise legend

lines(fit.exp,
      show = list(quant = FALSE, conf = FALSE, OT = TRUE, MAX = TRUE),
      label = "expon",
      par = RLpar(quant.col = "orange",
        OT.pch = 21, OT.cex = 1.2, OT.col = "SeaGreen", OT.bg = "yellow",
        MAX.block1.col = "purple", MAX.block1.bg = "mistyrose",
        MAX.block1.lwd = 1.4),
      legendEnvir = legEnv
      )

lines(fit.gpd,
      pct.conf = c(95, 70),
      show = list(quant = TRUE, conf = TRUE),
      label = "GPD",
      par = RLpar(quant.col = "darkcyan", conf.conf1.col = "red"),
      legendEnvir = legEnv
      )

RLlegend.show(envir = legEnv) ## now draw legend
```

`predict.Renouv` *Compute return levels and confidence limits for a "Renouv" object*

Description

Compute return levels and confidence limits for an object of class "Renouv".

Usage

```
   ## S3 method for class 'Renouv'
predict(object,
        newdata = c(10, 20, 50, 100, 200, 500, 1000),
        cov.rate = FALSE,
        level = c(0.95, 0.7),
        trace = 1, eps = 1e-06,
        ...)
```

Arguments

object	An object of class "Renouv" typically created by using the Renouv function.
newdata	The return period at which return levels and confidence bounds are wanted.
cov.rate	If FALSE, the delta method will not take into account the uncertainty on the event rate lambda of the Poisson process. NOT IMPLEMENTED YET. At the time only FALSE is possible.
level	Confidence levels as in other 'predict' methods (not percentages).
trace	Some details are printed when trace is not zero.
eps	Level of perturbation used to compute the numerical derivatives in the delta method.
...	Further arguments passed to or from other methods.

Details

Unless in some very special cases, the confidence limits are approximated ones computed by using the delta method with numerical derivatives.

Value

A data.frame with the expected return levels (col. named "quant") at the given return periods, and confidence limits. The returned object has an infer.method attribute describing the method used to compute the confidence limits.

Note

Despite of its name, this method does not compute true predictions. A return period is to be interpreted as an average interevent time rather than the duration of a specific period of time. For instance, the expected return level for a given return period with length 100 years is the level that would be on average exceeded once every 100 years (assuming that the model description in object is correct).

Author(s)

Yves Deville

References

Coles S. (2001) *Introduction to Statistical Modelling of Extremes Values*, Springer.

See Also

Renouv to fit Renouv model.

Examples

```
## Use Brest data
fit <- Renouv(Brest)
pred <- predict(fit, newdata = c(100, 125, 150, 175, 200),
                level = c(0.99, 0.95))
```

`readXML` *Read data using an XML index file*

Description

Read one or several dataset(s) using an XML index file specifying the data sets to read and the structure of each

Usage

```
readXML(name,
        dir,
        index = "index.xml",
        trace = 0)
```

Arguments

`name`	Name for the dataset that will be matched against the `name` attribute of datasets as they are given in the index file.
`dir`	Path to the directory where the index file and all data files should be found.
`index`	Name (short) of the index file. This file must be in the directory given by `dir`.
`trace`	Integer or logical to trace the successive steps by short indications.

Details

The XML index file is parsed within R. Then according to the indications within the index file, other files are read (e.g. csv files). In this way, data returned as a list can contain heterogeneous data: Over Threshold (OT) data, missing periods, MAX data, etc. Various pieces of information are also stored in list elements with name containing the `"info"` string.

This function requires the CRAN package XML.

Value

A list with the data read.

`info`	General information about the data: `varName`, `varShorlLab` and `varUnit` give the variable name unit and short label.
`OTinfo`	Information for the Over the Threshold (OT).

OTdata	Over the Threshold (OT) data.
OTmissing	Missing periods within the OTdata period.
MAXinfo	Information for the MAX (r-largest) supplement data.
MAXdata	MAX supplement data.
OTSinfo	Information for the Over the Threshold Supplement (OTS) data.
OTSdata	Over the Threshold (OT) supplement data.

Note

The flat files (usually .csv files) can also be read in a more conventional way e.g. through read.table. However, conform to the index.xml examples or to the index.xsd schema to see how to adjust the reading of parameters such as sep, etc.

Author(s)

Yves Deville

See Also

See Brest for an example of such a list.

See URL https://gforge.irsn.fr/gf/project/renext for data related to Renext.

Examples

```
## Not run:
if (require(XML)) {
   ## use 'index.xml' file shiped with Renext
   dir1 <- system.file("Rendata", package = "Renext")
   BrestNew1 <- readXML(name = "Brest", dir = dir1)
   test1 <- readXML(name = "test1", dir = dir1)
}

## End(Not run)
```

Renouv *Fit a 'Renouvellement' model*

Description

Fit a 'renouvellement' POT model using Over the Threshold data and possibly historical data of two kinds.

Usage

```
Renouv(x,
       threshold = NULL,
       effDuration = NULL,
       distname.y = "exponential",
       MAX.data = NULL,
       MAX.effDuration = NULL,
       OTS.data = NULL,
       OTS.effDuration = NULL,
       OTS.threshold = NULL,
       fixed.par.y = NULL,
       start.par.y = NULL,
       force.start.H = FALSE,
       numDeriv = TRUE,
       trans.y = NULL,
       jitter.KS = TRUE,
       pct.conf = c(95, 70),
       rl.prob = NULL,
       prob.max = 1.0-1e-04 ,
       pred.period = NULL,
       suspend.warnings = TRUE,
       control = list(maxit = 300, fnscale = -1),
       control.H = list(maxit = 300, fnscale = -1),
       trace = 0,
       plot = TRUE,
       ...)
```

Arguments

`x`	Either a numeric vector or an object of the class `"Rendata"`. In the first case, `x` contains all the levels above the threshold for a variable of interest. In the second case, most formal arguments take values in accordance with the object content, and can be by-passed by giving the formal explicitly.
`threshold`	Value of the threshold for the OT data.
`effDuration`	Effective duration, i.e. duration of the OT period .
`distname.y`	Name of the distribution for the exceedances over the threshold. See **Details** below.
`MAX.data`	Either a numeric vector or a list of numeric vectors representing historical data r-max by blocks. When *a vector* is given, there is only one block, and the data are the corresponding r-max observed levels where r is the vector length; the block duration is given in `MAX.effDuration`. When *a list* is given, each list element contains the data for one block, and the effective duration are in `MAX.effDuration`
`MAX.effDuration`	Vector of (effective) durations, one by block MAX data.
`OTS.data`	A numeric vector or a list of numeric vectors representing supplementary Over Threshold data in blocks. When a *vector* is given, there is only one block, and

the data contain all the 'historical' levels over the corresponding threshold given in OTS.threshold. The block duration is given in OTS.effDuration. When a *list* is given, each list element contains the data for one block, and the threshold and effective duration are in OTS.threshold and OTS.effDuration.

OTS.effDuration
: A numeric vector giving the (effective) durations for the OTS blocks.

OTS.threshold
: A vector giving the thresholds for the different OTS blocks. The given values must be greater than or equal to the value of threshold.

fixed.par.y
: Named list of known (or fixed) parameter values for the y-distribution.

start.par.y
: Named list of parameter initial values for the y-distribution. Only used when the distribution does not belong to the list of special distributions.

force.start.H
: Logical. When TRUE, the values in start.par.y (which must then be correct) will be used also as starting values in the maximisation of the global likelihood : OT data and historical data. This is useful e.g. when the historical data fall outside of the support for the distribution fitted without historical data. See below the **Details** section.

numDeriv
: Logical: should the hessian be computed using the numDeriv package (value TRUE) or should it be taken from the results of optim?

trans.y
: Transformation of the levels *before thresholding* (if not NULL). This is only possible with the "exponential" value distname.y. The two allowed choices are "square" and "log" meaning that the fitted (exponentially distributed) values are x.OT^2 -threshold^2 and log(x.OT) -log(threshold) respectively. The corresponding distributions for x.OT may be called "square-exponential" and "log-exponential".

jitter.KS
: Logical. When set to TRUE, a small amount of noise is added to the "OT" data used in the Kolmogorov-Smirnov test in order to remove ties. This is done using the OTjitter function.

pct.conf
: Character or numeric vector specifying the percentages for the confidence (bilateral) limits on quantiles.

rl.prob
: Vector of probabilities for the computation of return levels. These are used in plots (hence must be dense enough) and appear on output in the data.frame ret.lev.

prob.max
: Max value of probability for return level and confidence limits evaluations. This argument 'shortens' the default prob vector: values > prob.max in the default prob vector are omitted. Ignored when a prob argument is given.

pred.period
: A vector of "pretty" periods at which return level and probability will be evaluated and returned in the pred data.frame.

suspend.warnings
: Logical. If TRUE, the warnings will be suspended during optimisation steps. This is useful when the parameters are subject to constraints as is usually the case.

control
: A named list used in optim for the no-history stage (if any). Note that fnscale = -1 says that maximisation is required (not minimisation) and must not be changed!

control.H
: A named list used in optim for the historical stage (if any).

trace
: Level of verbosity. Value 0 prints nothing.

plot
: Draw a return level plot?

...
: Arguments passed to plot.Renouv, e.g. main, ylim.

Details

The model is fitted using Maximum Likelihood (ML).

Some distributions listed below and here called "special" are considered in a special manner. For these distributions, it is not necessary to give starting values nor parameter names which are unambiguous.

distribution	parameters
`exponential`	`rate`
`weibull`	`shape`, `scale`
`gpd`	`scale`, `shape`
`lomax`	`scale`, `shape`
`maxlo`	`scale`, `shape`
`log-normal`	`meanlog`, `sdlog`
`gamma`	`shape`, `scale`
`mixexp2`	`prob1`, `rate1`, `delta`

Other distributions can be used. Because the probability functions are then used in a "black-box" fashion, these distributions should respect the following *formal requirements*:

1. The name for the *density*, *distribution* and *quantile* functions must obey to the *classical "prefixing convention"*. Prefixes must be respectively: `"d"`, `"p"`, `"q"`. This rules applies for distribution of the `stats` package and those of many other packages such `evd`.
2. *The first (main) argument must be vectorisable* in all three functions, i.e. a vector of `x`, `q` or `p` must be accepted by the density, the distribution and the quantile functions.
3. *The density must have a* `log` *formal* argument. When `log` is `TRUE`, the log-density is returned instead of the density.

For such a distribution, it is necessary to give arguments names in `start.par.y`. The arguments list must have exactly the required number of parameters for the family (e.g. `2` for `gamma`). Some parameters can be fixed (known); then the parameter set will be the reunion of those appearing in `start.par.y` and those in `fixed.par.y`. Anyway, in the present version, *at least one parameter must be unknown* for the `y` part of the model.

Mathematical requirements exist for a correct use of ML. They are refered to as "regularity conditions" in ML theory. Note that the support of the distribution must be the set of non-negative real numbers.

The estimation procedure differs according to the existence of historical (MAX) data.

1. When no historical data is given, the whole set of parameters contains orthogonal subsets: a "point process" part concerning the process of events, and an "observation" part concerning the threshold exceedances part. The parameters can in this case be estimated separately. The rate of the Poisson process is estimated by the empirical rate, i.e. the number of events divided by the total duration given in `effDuration`. The "Over the Threshold" parameters are estimated from the exceedances computed as `x.OT` minus the threshold.
2. When historical data is given, the two parameter vectors must be coped with together in maximising the global likelihood. In this case, we begin the estimation ignoring the historical data and then use the estimates as starting values for the maximisation of the global likelihood. In some circumstances, the estimates obtained in the first stage can not be used with historical

data because some of these fall outside the support of the distribution fitted. This can happen e.g. with a `distname.y = "gpd"` when historical data exceed `threshold - scale/shape` for the values of `shape` and `scale` computed in the first stage.

Value

An object with class `"Renouv"`. This is mainly a list with the various results.

`est.N`	Estimate(s) for the count `"N"` part. This estimate does not use the historical data, even if is available.
`est.y`	Estimate(s) for the exceedance `"y"` part. This estimate does not use the historical data, even if available.
`cov.N, cov.y`	The (co-)variances for the estimates above.
`estimate`	Estimate(s) for the whole set of parameters based on OT data **and on historical data** if available.
`ks.test`	Kolmogorov-Smirnov goodness-of-fit test.
`ret.lev`	A data.frame containing return levels and confidence limits. The corresponding probabilities are either provided by user or taken as default values.
`pred`	A data.frame similar to `ret.lev`, but with "pretty" return periods. These are taken as the provided values `pred.period` if any or are chosen as "round" multiples of the time unit (taken from `effDuration`). The periods are chosen in order to cover periods ranging from 1/10 to 10 time units.

Other results are available. Use `names(result)` to see their list.

Except in the the special case where `distname.y` is `"exponential"` and where no historical data are used, the inference on quantiles is obtained with the *delta method* and using numerical derivatives. Confidence limits are unreliable for return levels much greater than the observation-historical period.

Due to the presence of estimated parameters, the Kolmogorov-Smirnov test is unreliable when less than 30 observations are available.

Warning

With some distributions or in presence of historical data, the estimation can fail due to some problem during the optimisation. Even when the optimisation converges, the determination of the (numerical) hessian can be impossible: This can happen if *one or more parameter is too small* to compute a finite difference approximation of gradient. For instance the 'rate' parameter of the exponential distribution (= inverse mean) will be small when the mean of exceedances is large.

A possible solution is then to **rescale the data** e.g. dividing them by 10 or 100. As a rule of thumb, an acceptable scaling leads to data (exceedances) of a few units to a few hundreds, but **an order of magnitude of thousands or more should be avoided and reduced by scaling**. The rescaling is recommanded for the square exponential distribution (obtained with `trans = "square"`) since the observations are squared.

Another possible way to solve the problem is to change the `numDeriv` value.

This problem will be solved in a future version.

Note

The model only concerns the "Over the Threshold" part of the distribution of the observations. When historical data is used, observations should all be larger than the threshold.

The name of the elements in the returned list is indicative, and is likely to be changed in future versions. At the time, the effect of historical data on estimation (when such data exist) can be evaluated by comparing `c(res$est.N, res$est.y)` and `res$estimate` where `res` is the results list.

Some warnings may indicate that missing values are met during the optimisation process. This is due to the evaluation of the density at tail values. At the time the ML estimates are computed using an unconstrained optimisation, so invalid parameter values can be met during the maximisation or even be returned as (invalid) estimates.

Validity tests for the estimation in presence of historical data have been limited at the time. Therefore this possibility should be regarded as experimental.

Author(s)

Yves Deville

References

Miquel J. (1984) *Guide pratique d'estimation des probabilités de crues*, Eyrolles (coll. EDF DER).

See Also

`rRenouv` to simulate "renouvellement" data, `RLplot` for the *return level plot*. See `optim` for the tuning of the optimisation. The `RenouvNoEst` can be used to create an object with S3 class `"Renouv"` from known parameters.

Examples

```
## Garonne data. Use a "Rendata" object as 'x'. Historical data are used!
fit <- Renouv(x = Garonne, distname = "weibull", trace = 1,
              main = "'Garonne' data")
summary(fit)

## generates a warning because of the ties
fit2 <- Renouv(x = Garonne, distname = "gpd",
               jitter.KS = FALSE,
               threshold = 2800, trace = 1,
               main = "'Garonne' data with threshold = 2800 and GPD fit")

## use a numeric vector as 'x'
fit3 <-
    Renouv(x = Garonne$OTdata$Flow,
           threshold = 2500,
           effDuration = 100,
           distname = "gpd",
           OTS.data = list(numeric(0), c(6800, 7200)),
           OTS.effDuration = c(100, 150),
```

```
            OTS.threshold = c(7000, 6000),
            trace = 1,
            main = "'Garonne' data with artificial \"OTS\" data")
## Add historical (fictive) data
fit4 <- Renouv(x = Garonne$OTdata$Flow,
               threshold = 2500,
               effDuration = 100,
               distname = "weibull",
               fixed.par.y = list(shape = 1.1),
               OTS.data = list(numeric(0), c(6800, 7200)),
               OTS.effDuration = c(100, 150),
               OTS.threshold = c(7000, 6000),
               trace = 0,
               main = "'Garonne' data with artificial \"OTS\" data")
```

RenouvNoEst *Define a 'renouvellement' model without estimation*

Description

Build a 'renouvellement' model using parameters given by the user.

Usage

```
RenouvNoEst(threshold,
            estimate = NULL,
            distname.y = "exponential",
            fixed.par.y = NULL,
            trans.y = NULL,
            pct.conf = c(95, 70),
            rl.prob = NULL,
            prob.max = 1 - 1e-04,
            pred.period = NULL,
            cov = NULL,
            nb.OT = NULL,
            infer.method = NULL)
```

Arguments

threshold	The threshold.
estimate	Numeric named vector containing the estimates for the parameters. It must be compatible with the distribution chosen, and must contain in first position an element named "lambda" representing an estimated event rate in *events by year*.
distname.y	Character giving the name of the distribution.
fixed.par.y	Numeric named vector containing values for vectors which are considered as fixed (and not estimated).

trans.y	Transformation as in Renouv. Used only when distname.y is equal to "exponential".
pct.conf	Vector of percents for confidence limits.
rl.prob	Probability used in the return level computations. These values are used for instance in return level plots produced with the plot.Renouv method. When NULL a default vector is used.
prob.max	Maximal probability for which computations are done.
pred.period	Vector of periods for which predicted return levels will be computed.
cov	Covariance matrix for the provided estimated parameters. Must have rownames and colnames in accordance with those of estimate. This covariance matrix is used to build confidence limits on parameters and on return levels using the *delta method*.
nb.OT	Number of data over the threshold used for estimation. This will be used only when distname.y is equal to "exponential".
infer.method	Inference method. Will normally be the *delta method*.

Details

This function is used for plotting or comparing models with known parameter estimates but with no data available.

The parameters estimates should be accompanied with a covariance matrix assuming an approximatively normal joint distribution of these. This matrix is usually obtained by computing the numerical derivatives of the log-likelihood at the second order at the estimates. This covariance is used to compute approximate confidence limits for the return levels of the unknown true distribution that was estimated.

Value

An object of class "Renouv" representing a 'renouvellement' model similar to those built with Renouv. This is mainly a list. Note however that some list elements found in Renouv objects built by Renouv can not be found here. For instance, the returned objects embeds no goodness-of-fit results since the object is created without making use of any data.

Author(s)

Yves Deville

See Also

Renouv to estimate such models.

Examples

```
##=====================================================================
## Example from S. Coles' book, page 86 'rainfall data'
##=====================================================================
estimate <- c(lambda = 152/48, scale = 7.44, shape = 0.184)
cov <- matrix(c(4.9e-7,  0.0000,  0.0000,
                0.0000,  0.9180, -0.0655,
```

```
                0.0000, -0.0655,  0.0102),
              nrow = 3)
colnames(cov) <- rownames(cov) <- names(estimate)
renNE <- RenouvNoEst(threshold = 30, distname.y = "gpd",
                     pct.conf = c(95, 70),
                     estimate = estimate,
                     nb.OT = 152, cov = cov)
summary(renNE)
plot(renNE, main = "Daily rainfall data SW England", ylim = c(0, 400) )
```

RLlegend *Legend management for return level plots*

Description

Legend management for return level plots produced with the `plot` and `lines` method of the `"Renouv"` class.

Usage

```
RLlegend.ini(x = "topleft", bty = "n", envir, ...)
RLlegend.show(envir)
```

Arguments

`x`	A possible value for the `x` argument of `legend`. This will usually be a character giving the position e.g, `"topleft"` or `"bottomleft"`. See the `legend` function help.
`bty`	As in `legend`. The default value `"n"` differs from the default value of `legend`.
`envir`	An environment in which a variable `.RLlegend` will be stored.
`...`	Other arguments to be kept in the list and passed later to `legend`. These arguments should chosen among those of `legend` modifying the global legend appearance (e.g., `bg`) but not among those modifying the legend content (e.g. `col` `pt.bg`, `legend`, ...) since the content is here built semi-automatically.

Details

This function is to be used in conjunction with `plot.Renouv` and `lines.Renouv` methods. It allows the construction of a legend in a semi-automatic fashion, using the value of the `par` argument of the `plot` and `lines` methods to specify the legend construction.

Each call to the `plot.Renouv` or `lines.Renouv` changes the content of a list variable named `.RLlegend` in the environment defined by the `envir` formal. This list is re-created when `RLlegend.ini` is called, and is used later to draw a legend on the active device when `RLlegend.draw` is called. Between these two calls, the `plot` and `lines` methods should be used with their arg `legend` set to `FALSE`.

The list variable used to store the needed information can be assigned into the global environment at the user responsibility by a suitable choice of `envir`.

Value

`RLlegend.ini` returns a copy of the variable which is set.

`RLlegend.show` returns nothing.

Note

The size of symbols (i.e, *plotting characters*) can be set by using the `RLpar` function and the `par` argument of the methods `plot.Renouv` and `lines.Renouv`. However it can not be changed in the legend.

Author(s)

Yves Deville

See Also

`plot.Renouv` and `lines.Renouv` for and the `RLpar` function to change the graphical parameters of the plot and the legend by using the `par` argument.

Examples

```
## use Garonne data
xG <- Garonne$OTdata$Flow
## use spetial "exponential" distribution
fit1 <- Renouv(x = xG, threshold = 2500, distname.y = "exponential",
               effDuration = 65, plot = FALSE)

## use 'exp' in black box fashion, hence with delta method
fit2 <- Renouv(x = xG, , threshold = 2500, distname.y = "exp",
               effDuration = 65, start.par.y = c(rate = 1), plot = FALSE)

legEnv <- new.env()
RLlegend.ini(envir = legEnv) ## initialise legend

## sample points only
plot(fit1, main = "Two types to confidence lims",
     show = list(OT = TRUE, quant = FALSE, conf = FALSE),
     label = "",
     legend = FALSE)
## quant and confidence lims
lines(fit1,
     show = list(OT = FALSE, quant = TRUE, conf = TRUE),
     label = "exact",
     legend = TRUE,
     legendEnvir = legEnv)
## quant (overplot) and confidence lims
lines(fit2,
```

```
        show = list(OT = FALSE, quant = TRUE, conf = TRUE),
        par = RLpar(quant.lty = 2, quant.col = "SpringGreen2",
          conf.conf1.col = "orangered", conf.conf1.lwd = 3,
          conf.conf2.col = "orangered", conf.conf2.lwd = 3),
        label = "delta",
        legend = TRUE,
        legendEnvir = legEnv)

RLlegend.show(envir = legEnv) ## now draw legend
```

RLpar *Graphical parameters for Return Level plots*

Description

Build a hierarchical list of graphical parameters that can be used in the methods plot or lines for the class "Renouv".

Usage

```
RLpar(mono = TRUE, ...)
```

Arguments

mono	Logical. The default TRUE is for plots possibly using colors but that can be printed in grayscale. With the value FALSE, curves or symbols will appear distinctly on a color device but not necessarily when printed in grayscale.
...	Arguments with names corresponding to the herarchical structure and the graphical parameter to be changed.

Details

The formals are in correspondence with the list hierarchy using a column "." as separator to define the tree. Thus a quant.col formal argument can be used to specify the color of the quantile (or return level) curve, while conf.conf1.col will be used for the first confidence limits (lower and upper).

Value

A list containing lists in a hierarchical fashion. At the root level, an element concerns a single curve (e.g. the return level curve), a single scatterplot (e.g. sample used in POT), a group of curves (e.g. the confidence limits) or a group of scatterplots (e.g. the collection of MAX historical blocks). For single elements (curve or scatterplot) the list contains graphical elements with values as they would be given in plot or lines calls. For group elements, each element is a list of such lists.

Note

A list of default parameter values is built first using the model suitable for the mono value. Then the values provided by the user overwrite the existing. Thus a curve can be coloured even if mono = TRUE, if a colour specification is given for the corresponding element.

When the same parameter name is used several times in RLpar, a warning is thrown.

Author(s)

Yves Deville

See Also

plot.Renouv and lines.Renouv with which RLpar is to be sued.

Examples

```
## change color for quantile curve and type for confidence
## limits #1 (with largest confidence level).
newRLpar <- RLpar(quant.col = "red", conf.conf1.lty = "dashed")
newRLpar$quant

## show the names of all possible editable parameters
names(unlist(RLpar()))
```

RLplot *Return level plot*

Description

Return level plot for "Renouvellement" data.

Usage

```
RLplot(data,
       x = NULL,
       duration = 1,
       lambda,
       conf.pct = 95,
       mono = TRUE,
       mark.rl = 100,
       mark.labels = mark.rl,
       mark.col = NULL,
       main = NULL,
       ylim = NULL,
          ...)
```

Arguments

data	A data.frame object with a column named quant
x	Optional vector of observed levels
duration	The (effective) duration corresponding to x if this argument is used.
lambda	Rate (in accordance with duration.
conf.pct	a vector (character or integer) giving confidence levels. See **Details** below.
mono	If TRUE colours are replaced by black.
mark.rl	Return levels to be marked on the plot.
mark.labels	Labels shown at positions in mark.rl.
mark.col	Colours for marked levels.
main	Main title for the return level plot (defaults to empty title).
ylim	Limits for the y axis (defaults to values computed from the data).
...	Further args to be passed to plot. Should be removed in future versions.

Details

Percents should match column names in the data.frame as follows. The upper and lower limits are expected to be U.95 and L.95 respectively. For a 70% confidence percentage, columns should have names "U.70" and "L.70".

The plot is comparable to the return level described in Coles'book and related packages, but the return level is here in log-scale while Coles uses a loglog-scale. A line corresponds here to a one parameter exponential distribution, while Coles'plot corresponds to Gumbel, however the two plots differ only for small return periods. This plot is identical to an expplot but with x and y scales changed: only axis tick-marks differ. The convexity of the scatter plot is therefore opposed in the two plots.

Note

Confidence limits correspond to *two-sided symmetrical intervals*. This means that the (random) confidence interval may be under or above the true unknown value with the same probabilities. E.g. the probability that the unknown quantile falls above U.95 is 2.5%. The two bounds are yet generally not symmetrical with respect to quant; such a behaviour follows from the use of "delta" method for approximate intervals.

It is possible to add graphical material (points, lines) to this plot using log(returnlev) and quantile coordinates. See **Examples** section.

Author(s)

Yves Deville

References

Coles S. (2001) *Introduction to Statistical Modelling of Extremes Values*, Springer.

See Also

See expplot for a classical exponential plot. See Also as Renouv to fit "Renouvellement" models. The return.level function in the extRemes package.

Examples

```
## Typical probability vector
prob <- c(0.0001,
  seq(from = 0.01, to = 0.09, by = 0.01),
  seq(from = 0.10, to = 0.80, by = 0.10),
  seq(from = 0.85, to = 0.99, by = 0.01),
  0.995, 0.996, 0.997, 0.998, 0.999, 0.9995)

## Model parameters rate = #evts by year, over nyear
lambda <- 4
nyear <- 30
theta.x <- 4

## draw points
n.x <- rpois(1, lambda = lambda*nyear)
x <- rexp(n.x, rate = 1/theta.x)

## ML estimation (exponential)
lambda.hat <- n.x / nyear
theta.x.hat <- mean(x)

## Compute bounds (here exact)
alpha <- 0.05

quant <- qexp(p = prob, rate = 1/theta.x.hat)

theta.L <- 2*n.x*theta.x.hat / qchisq(1 - alpha/2, df = 2*n.x)
theta.U <- 2*n.x*theta.x.hat / qchisq(alpha/2, df = 2*n.x)

L.95 <- qexp(p = prob, rate = 1/theta.L)
U.95 <- qexp(p = prob, rate = 1/theta.U)

## store in data.frame object
data <- data.frame(prob = prob, quant = quant, L.95 = L.95, U.95 = U.95)

RLplot(data = data, x = x, lambda = lambda.hat,
       duration = nyear,
       main = "Poisson-exponential return levels")

RLplot(data = data, x = x, lambda = lambda.hat, duration = nyear,
       mark.rl = 10, mark.labels = "10 ans", mono = FALSE, mark.col = "SeaGreen",
       main = "Poisson-exponential return levels")

points(x = log(50), y = 25, pch = 18, cex = 1.4, col = "purple")
text(x = log(50), y = 25, col ="purple", pos = 4, labels = "special event")
```

roundPred *Round quantiles in a pseudo-prediction table*

Description

Round the quantiles of a pseudo prediction table such that computed by `predict.Renouv`.

Usage

```
roundPred(pred, dig.quant = NA)
```

Arguments

`pred`	The data.frame containing the predicted quantiles and return levels.
`dig.quant`	Number of digits. Guessed if not provided.

Details

Only the columns that can be considered as quantiles are rounded. These are assumed to have names `"quant"` for the expected return level and `"L."` or `"U."` followed by a percentage for lower and upper confidence limits (e.g. `"L.95"` and `"U.95"` for 95% percent confidence limits. The number of digits guessed is experimental.

Value

A data.frame with the same structure as that given, but with some columns rounded.

rRenouv *Simulation of "Renouvellement" data*

Description

Simulation of "Renouvellement" data, i.e. observations Over a Threshold and their counts on blocks (e.g. years).

Usage

```
rRenouv(densfun.y = "exponential",
        par.y = list(rate = 1),
        densfun.N = "poisson",
        par.N = list(lambda = 6),
        threshold = 0,
        aggreg = TRUE,
        nb = 50,
        labb = seq(to = 2009, by = 1, length = nb),
        w = rep(1, nb))
```

Arguments

densfun.y	A character string specifying the distribution of the exceedances over the threshold. At the time only "exponential", "weibull" and "gpd" are available.
par.y	Named list giving parameter values for the "y"-distribution. See examples below.
densfun.N	Character string specifying the distribution for the counts. At the time only "poisson" and "negative binomial" are allowed.
par.N	Named list giving parameter values for the N-distribution for counts. The names are lambda in the Poisson case ("poisson"), and size and prob in the negative binomial case ("negative binomial" case).
threshold	Threshold value.
aggreg	Only TRUE is possible at the time.
nb	Number of blocks (or time intervals).
labb	Numeric vector of length nb that will be used in replacement of the block numbers vector. Typically it can contain year numbers. Use NULL to obtain blocks from 1 to nb.
w	Vector of blocks (time interval) length, i.e. duration.

Value

A list with the following objects

x	Vector of x values i.e. threshold plus exceedances.
N	Vector of counts.
block	Vector of length length(x) giving the block number for the corresponding element in x. When coerced to a factor block has length(N) levels.

Note

At the present time, the random drawings of the gpd distribution are generated with the the rgpd of the evd package.

In future versions, this function might return an object with a special class (with name such as "renouv"). Then classical simulate, plot, ... methods could be defined for that class.

Author(s)

Yves Deville

See Also

Renouv to fit Renouvellement models.

Examples

```
## Not run:
test1 <- rRenouv(nb = 100,
                 threshold = 40,
                 par.N = list(lambda = 2),
                 densfun.y = "gpd",
                 par.y = mom2par("gpd", mean = 30, sd = 36))

## End(Not run)

test2 <- rRenouv(nb = 100,
                 threshold = 40,
                 par.N = list(lambda = 2),
                 densfun.y = "weibull",
                 par.y = mom2par("weibull", mean = 30, sd = 36))

test3 <- rRenouv(nb = 100,
                 threshold = 40,
                 densfun.N = "negative binomial",
                 par.N = list(gamma = 10, prob = 0.7),
                 densfun.y = "weibull",
                 par.y = mom2par("weibull", mean = 30, sd = 36))
```

---

skip2noskip *Fix non-skipped periods from skipped ones*

Description

Compute non-skipped periods form start and end of skipped periods.

Usage

```
skip2noskip(skip = NULL,
            start = NULL,
            end = NULL)
```

Arguments

skip	A data.frame object with start and end columns that can be coerced to POSIXct. Other columns can be present (and will be ignored). Each row describes a missing period. Rows must be sorted in chronological order and periods should not overlap. Validity checks are at the time very limited.
start	Beginning of the whole period, to be used in as.POSIXct.
end	End of the whole period to be used in as.POSIXct.

Details

In a 'normal' use of this function `start` and `end` are given, and are respectively *before the beginning* of the first `skip` period and *after the end* of the last `skip` period. Thus the returned dataframe will have `nrow(skip)+1` rows. However, `start` and `end` can be `NULL` in which case only the `nrows(skip)-1` "inner" non-skipped periods will be returned. If `start` and `end` are `NULL` and `skip` has only one row, the returned result is `NULL`.

Value

A data.frame object with two POSIXct columns named `start` and `end`. Each row corresponds to a non-skipped period

Author(s)

Yves Deville

See Also

`readXML` for reading data from XML and csv files.

Examples

```
## Brest data embeds a description of the gaps

ns <- skip2noskip(skip = Brest$OTmissing)

ns2 <- skip2noskip(skip = Brest$OTmissing,
                   start = Brest$OTinfo$start,
                   end = Brest$OTinfo$end)

## check durations. dur2 should be equal to the effective
## duration (with an error of a fraction of day)
dur <- as.numeric(sum(ns$end-ns$start))/365.25
dur2 <- as.numeric(sum(ns2$end-ns2$start))/365.25
```

SLTW *Shifted Left Truncated Weibull (SLTW) distribution*

Description

Density function, distribution function, quantile function and random generation for the Shifted Left Truncated Weibull distribution.

Usage

```
dSLTW(x, delta = 1.0, shape = 1.0, scale = 1.0, log = FALSE)
pSLTW(q, delta = 1.0, shape = 1.0, scale = 1.0, lower.tail = FALSE)
qSLTW(p, delta = 1.0, shape = 1.0, scale = 1.0)
rSLTW(n, delta = 1.0, shape = 1.0, scale = 1.0)
```

Arguments

`x, q`	Vector of quantiles.
`p`	Vector of probabilities.
`n`	Number of observations.
`delta, shape, scale`	Shift, shape and scale parameters. Vectors of length > 1 are not accepted.
`log`	Logical; if `TRUE`, the log density is returned.
`lower.tail`	Logical; if `TRUE` (default), probabilities are $\Pr[X \leq x]$, otherwise, $\Pr[X > x]$.

Details

The SLTW distribution function with shape $\alpha > 0$, scale $\beta > 0$ and shift $\delta > 0$ has survival function

$$S(y) = \exp\left\{-\left[\left(\frac{y+\delta}{\beta}\right)^{\alpha} - \left(\frac{\delta}{\beta}\right)^{\alpha}\right]\right\} \qquad (y > 0)$$

This distribution is that of $Y := X - \delta$ conditional to $X > \delta$ where X follows a Weibull distribution with shape α and scale β.

The hazard and mean residual life (MRL) are monotonous functions with the same monotonicity as their Weibull equivalent (with the same shape and scale). The moments or even expectation do not have simple expression.

This distribution is sometimes called *power exponential*. It is occasionally used in POT with the shift `delta` taken as the threshold as it should be when the distribution for the level X (and not for the exceedance Y) is known to be the standard Weibull distribution.

Value

`dSLTW` gives the density function, `pSLTW` gives the distribution function, `qSLTW` gives the quantile function, and `rSLTW` generates random deviates.

See Also

`Lomax` for the Lomax distribution which is a limit case of `SLTW`.

Examples

```
shape <- rexp(1)+1
delta = 10

xl <- qSLTW(c(0.001, 0.99), delta = delta, shape = shape)
x <- seq(from = xl[1], to = xl[2], length.out = 200)
```

```
f <- dSLTW(x, delta = delta, shape = shape)
plot(x, f, type = "l", main = "SLTW density")

F <- pSLTW(x, delta = delta, shape = shape)
plot(x, F, type = "l", main = "SLTW distribution")

X <- rSLTW(5000, delta = delta, shape = shape)
## Should be close to uniform repartition
plot(ecdf(pSLTW(X, delta = delta, shape = shape)))
```

`summary.Rendata` *Summary method for "Rendata" objects*

Description

Summary method for "Rendata" objects representing data to be used in renouvellement models.

Usage

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

  ## S3 method for class 'summary.Rendata'
print(x, ...)
```

Arguments

`object`	An object with class `"Rendata"`.
`x`	An object of class `"summary.Rendata"`, i.e. a result of a call to `summary.Rendata`.
`...`	Further arguments passed to or from other methods.

Author(s)

Yves Deville

Examples

```
## Brest example: no historical data
summary(Brest)

## Garonne example:  with historical data
summary(Garonne)
```

summary.Renouv *Summary method for "Renouv" objects*

Description

Summary method for "Renouv" objects representing 'Renouvellement' (POT) fitted models.

Usage

```
   ## S3 method for class 'Renouv'
summary(object,
        correlation = FALSE,
        symbolic.cor = FALSE,
        ...)

   ## S3 method for class 'summary.Renouv'
print(x,
      coef = TRUE,
      pred = TRUE,
      probT = FALSE,
      digits = max(3, getOption("digits") - 3),
      symbolic.cor = x$symbolic.cor,
      signif.stars = getOption("show.signif.stars"),
      ...)

   ## S3 method for class 'summary.Renouv'
format(x,
      ...)
```

Arguments

object	An object with class "Renouv".
x	An object of class "summary.Renouv", i.e. a result of a call to summary.Renouv.
correlation	Logical; if TRUE, the correlation matrix of the estimated parameters is returned and printed.
coef	Logical. If FALSE, the table of coefficients and t-ratios' will not be printed.
pred	Logical. If FALSE, the table of return periods/levels will not be printed.
probT	If FALSE, the p-values for the t-tests will not be printed nor displayed.
digits	the number of significant digits to use when printing.
symbolic.cor	logical. If TRUE, print the correlations in a symbolic form (see symnum) rather than as numbers.
signif.stars	logical. If TRUE, 'significance stars' are printed for each coefficient.
...	Further arguments passed to or from other methods.

Details

`print.summary.Renouv` tries to be smart about formatting the coefficients, standard errors, return levels, etc. `format.summary.Renouv` returns as a limited content as a character string. It does not embed coefficients values nor predictions.

Value

The function `summary.Rendata` computes and returns a list of summary statistics concerning the object of class `"Rendata"` given in `object`. The returned list is an object with class `"summary.Renouv"`.

The function `print.summary.Rendata` does not returns anything.

See Also

The model fitting function `Renouv` (to build `"Renouv"` model objects), `summary`.

Examples

```
## use Brest data
fit <- Renouv(Brest)
summary(fit)
```

---

vcov.Renouv *Variance-covariance matrix of the estimates of a "Renouv" object.*

Description

Variance-covariance matrix of the estimates of a "Renouv" object.

Usage

```
## S3 method for class 'Renouv'
vcov(object, ...)
```

Arguments

`object`	Object of class `"Renouv"`.
`...`	Not used at the time.

Value

A variance-covariance matrix. The rows an columns correspond to the parameters of the Renouv object. The are the rate `"lambda"` for the Poisson process, and the parameters of the distribution for the exceedances, with names depending on the chosen distribution.

Author(s)

Yves Deville

See Also

The vcov generic.

`weibplot` *Classical Weibull distribution plot*

Description

Plots a vector using Weibull distribution scales

Usage

```
weibplot(x,
         plot.pos = "exp",
         shape = NULL, scale = NULL,
         labels = NULL,
         mono = TRUE,
         ...)
```

Arguments

`x`	The vector to be plotted.
`plot.pos`	plotting position for points: either "exp" for *expected* ranks or "med" for a *median* rank approximation (see **Details** below).
`shape`	Shape parameter for one or several Weibull lines to be plotted.
`scale`	Scale parameter for one or several Weibull lines to be plotted.
`labels`	Text to display in legend when Weibull lines are specified.
`mono`	Monochrome graph.
`...`	Arguments to be passed to `plot`.

Details

This plot shows $\log\{-\log[1 - F(x)]\}$ against $\log(x)$ where $F(x)$ at point i is taken as $i/(n + 1)$ if `plot.pos` is `"exp"`, or as the "median rank" approximation $(i - 0.3)/(n + 0.4)$ if `plot.pos` is `"med"`.

Note

The graph displayed uses a log scale for x. The log-log scale for y is emulated via the construction of suitable graduations. So be careful when adding graphical material (points, etc) to this graph with functions of the "add to plot" family (`points`, `lines`, ...).

Author(s)

Yves Deville

See Also

The expplot function for an "exponential distribution" plot (dedicated to the shape = 1 case), and the fweibull function for ML estimation of the parameters.

Examples

```
x <- rweibull(200, shape = 1.2, scale = 1)
weibplot(x, main = "Classical Weibull plot")
## Weibull lines
weibplot(x, shape = c(0.9, 1.3), scale = 1)
weibplot(x, shape = c(0.9, 1.3), scale = 1,
         labels = c("before", "after"))
weibplot(x, shape = c(0.9, 1.3), scale = 1,
         labels = c("before", "after"),
         mono = TRUE)
```

Index