Package ‘ParetoPosStable’

October 12, 2022

Type Package
Title Computing, Fitting and Validating the PPS Distribution
Version 1.1
Date 2015-09-02
Maintainer Antonio Jose Saez-Castillo <ajsaez@ujaen.es>
Description Statistical functions to describe a Pareto Positive Stable (PPS) distribution and fit it to real data. Graphical and statistical tools to validate the fits are included.
License GPL-2 | GPL-3
LazyData true
NeedsCompilation no
Author Antonio Jose Saez-Castillo [aut, cre],
Faustino Prieto [aut],
Jose Maria Sarabia [aut]
Repository CRAN
Date/Publication 2015-09-02 15:52:04

R topics documented:

    coef.PPSfit ................................. 2
    forbes400 .................................. 3
    GoF .......................................... 3
    logLik.PPSfit ............................... 4
    pareto.fit ................................... 5
    ParetoPosStable ............................ 6
    plot.PPSfit .................................. 6
    PPS ........................................... 7
    PPS.fit ....................................... 9
**Description**

`coef` returns the parameter estimates in a `PPSfit` Object

**Usage**

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

**Arguments**

- `object` A `PPSfit` object, typically from `PPS.fit()`.
- `...` Other arguments.

**Value**

A list with the parameter estimates.

**See Also**

`PPS.fit`.

**Examples**

```r
x <- rPPS(50, 1.2, 100, 2.3)
fit <- PPS.fit(x)
coef(fit)
```
Description

The Forbes 400 or 400 Richest Americans is a list published by Forbes magazine of the wealthiest 400 Americans, ranked by net worth. The dataset presents results about 2012.

Format

A data frame with 400 observations on the following 2 variables.

- Name: members of the list names.
- NetWorth: net worth in 2012 in $ billion.

GoF

Goodness of fit tests for the Pareto Positive Stable (PPS) distribution

Description

Kolmogorov-Smirnov, Anderson-Darling and PPS goodness of fit tests to validate a PPS fit (typically from PPS.fit()).

Usage

GoF(PPSfit, k = 2000, parallel = TRUE, ncores = 2, ...)
logLik.PPSfit

Value
A list with the values of the tests statistics and the approximated p-values.

References

See Also
PPS.fit, plot.PPSfit

Examples
```r
x <- rPPS(50, 1.2, 100, 2.3)
fit <- PPS.fit(x)
GoF(fit, k = 50)
```

---

logLik.PPSfit

Log-likelihood value of a PPSfit Object

Description
It returns the log-likelihood value of a PPSfit Object

Usage
```r
## S3 method for class 'PPSfit'
logLik(object, ...)
```

Arguments
- `object`: A PPSfit Object.
- `...`: Other arguments.

Value
The log-likelihood.

References

See Also
PPS.fit
Examples

```r
x <- rPPS(50, 1.2, 100, 2.3)
fit <- PPS.fit(x)
logLik(fit)
```

---

**Description**

It is an auxiliary function for fitting a Pareto distribution as a particular case of a Pareto Positive Stable distribution, allowing the scale parameter to be held fixed if desired.

**Usage**

```r
pareto.fit(x, estim.method = "MLE", sigma = NULL, start, ...)
```

**Arguments**

- `x`: The vector of observations.
- `estim.method`: The estimation method, "MLE" or "OLS".
- `sigma`: The value of the scale parameter, if it is known; if the value is NULL, the parameter is estimated.
- `start`: Unused argument from `PPS.fit`.
- `...`: Other arguments.

**Details**

This function is called by `PPS.fit()` when Pareto argument is TRUE.

**Value**

A PPSfit Object.

**References**


**See Also**

`PPS.fit`, `coef.PPSfit`, `print.PPSfit`, `plot.PPSfit`, `GoF`
ParetoPosStable  Computing, Fitting and Validating the PPS Distribution

Description
Statistical functions to describe a Pareto Positive Stable (PPS) distribution and fit it to real data. Graphical and statistical tools to validate the fits are included.

Details
The main function you’re likely to need from ParetoPosStable is \texttt{PPS.fit}, in order to obtain a PPS fit from data. Validation can be obtained with \texttt{GoF} and \texttt{plot}.

References

plot.PPSfit  Plots to validate a Pareto Positive Stable (PPS) fit

Description
Plots to validate a PPS fit (typically from \texttt{PPS.fit()}) with different comparisons between empirical and theoretical functions.

Usage
## S3 method for class ‘PPSfit’
plot(x, which = 1:4, ask = prod(par("mfcol")) <
length(which) && dev.interactive(), ylim, breaks, ...)

Arguments
\begin{itemize}
\item \texttt{x}  a PPSfit Object.
\item \texttt{which}  values from 1 to 4 indicating the type of plot.
\item \texttt{ask}  an argument to control the plot window.
\item \texttt{ylim}  optional argument to control the y limits of the histogram. It is included to prevent non-desired scales on the y-axis.
\item \texttt{breaks}  optional argument to control the breakpoints of the histogram. See \texttt{hist} help for the details. It is included to prevent non-desired scales on the y-axis.
\item \texttt{...}  other arguments.
\end{itemize}
Details

The plots return:
1. The histogram of the observations and the fitted PPS density (which = 1). Optional ylim and breaks arguments are provided to prevent frequent imbalances between density and histogram scales in real data: they work as the analogue arguments of the default hist function. 2. The empirical distribution function of data and the cumulative distribution function of the fitted model (which = 2). 3. A rank-size plot in log-log scale to check the Pareto or power-law behaviour of data (which = 3). In the X-axis the log of the observations appears; in the Y-axis, the log of the empirical survival function. If the scatter-plot is around a straight line, then the observations exhibit a power law behaviour. The plot also includes the curve with the theoretical survival function of the model specified in the first argument class PPSfit: only when nu is 1, that curve is going to be a straight line. 4. A plot in a double log-log scale to check the adequacy of data to the PPS model (which = 4).

References


See Also

PPS.fit

Examples

data(forbes400)
f <- PPS.fit(forbes400$NetWorth)
par(mfrow=c(2,2))
plot(f)
dev.off()
plot(f, which = 1, breaks = seq(0, 60, length.out = 60))

PPS

The Pareto Positive Stable (PPS) distribution

Description

Density, distribution function, hazard function, quantile function and random generation for the Pareto Positive Stable (PPS) distribution with parameters lam, sc and v.
Usage

dPPS(x, lam, sc, v, log = FALSE)
hPPS(x, lam, sc, v)
pPPS(x, lam, sc, v, lower.tail = TRUE, log.p = FALSE)
qPPS(p, lam, sc, v, lower.tail = TRUE, log.p = FALSE)
rPPS(n, lam, sc, v)

Arguments

x vector of quantiles.
lam vector of (non-negative) first shape parameters.
sc vector of (non-negative) scale parameters.
v vector of (non-negative) second shape parameters.
log logical; if TRUE, probabilities/densities p are returned as log(p).
lower.tail logical; if TRUE (default), probabilities are \( P[X \leq x] \), otherwise, \( P[X > x] \).
log.p logical; if TRUE, probabilities/densities p are returned as log(p).
p vector of probabilities.
n number of random values to return.

Details

The PPS distribution has density

\[ f(x) = \lambda \nu [(\log(x/\sigma))^{\nu - 1}] \exp(-\lambda [\log(x/\sigma)]^\nu) / x, \]

cumulative distribution function

\[ F(x) = 1 - \exp(-\lambda [(\log(x/\sigma))^\nu]), \]

quantile function

\[ Q(p) = \sigma \exp\left([-1/(\lambda \log(1 - p))^{1/\nu}]\right) \]

and hazard function

\[ \lambda \nu (\log(x/\sigma))^{\nu - 1} x^{\nu - 1}. \]

See Sarabia and Prieto (2009) for the details about the numbers random generation.

Value

dPPS gives the (log) density, pPPS gives the (log) distribution function, qPPS gives the quantile function, and rPPS generates random samples. Invalid parameters will result in return value NaN, with a warning. The length of the result is determined by n for rPPS, and is the common length of the numerical arguments for the other functions.
References


Examples

```r
print(x <- sort(rPPS(10, 1.2, 100, 2.3)))
dPPS(x, 1.2, 100, 2.3)
pPPS(x, 1.2, 100, 2.3)
qPPS(pPPS(x, 1.2, 100, 2.3), 1.2, 100, 2.3)
hPPS(x, 1.2, 100, 2.3)
```

---

**PPS.fit**  
*Fitting the Pareto Positive Stable (PPS) distribution*

**Description**

PPS.fit() returns the fit of a PPS distribution to real data, allowing the scale parameter to be held fixed if desired.

**Usage**

```r
PPS.fit(x, estim.method = "MLE", sigma = NULL, start, Pareto = FALSE, ...)
```

**Arguments**

- `x`: a vector of observations
- `estim.method`: the method of parameter estimation. It may be "MLE", "iMLE", "OLS", or "LMOM".
- `sigma`: the value of the scale parameter, if it is known; if the value is NULL, the parameter is estimated.
- `start`: a list with the initial values of the parameters for some of the estimation methods.
- `Pareto`: a logical argument to constrain the PPS fit to a Pareto fit when the value is TRUE.
- `...`: other arguments.

**Details**

The maximum likelihood method implemented by the direct optimization of the log-likelihood is given by `estim.method = "MLE"`. The numerical algorithm to search the optimum is the “Nelder-Mead” method implemented in the `optim` function, considering as initial values those given in the start argument or, if it is missing, those provided by the OLS method.

A different approximation of the maximum likelihood estimates is given by `estim.method = "iMLE"`; it is an iterative methodology where `optimize()` function provides the optimum scale parameter value, while the `uniroot()` function solve normal equations for that given scale parameter.
The regression estimates ("OLS") searches an optimum scale value (in a OLS criterion) by the \texttt{optimize()} function. Then the rest of the parameters are estimated also by OLS, as appears in Sarabia and Prieto (2009).

In the L-moments method ("LMOM") estimates are obtained searching parameters that equal the first three sample and theoretical L-moments by means of the “Nelder-Mead” algorithm implemented in \texttt{optim()}; the initial values are given in the \texttt{start} argument or, if it is missing, provided by the "iMLE".

### Value

A \texttt{PPSfit} Object, a list with

- \texttt{estimate} parameter estimates.
- \texttt{loglik} the log-likelihood value.
- \texttt{n} the number of observations.
- \texttt{obs} the observations.
- \texttt{obsName} the name of the variable with the observations.
- \texttt{estim.method} the method of parameter estimation.

When this last value is "LMOM" the function also returns details about the convergence of the numerical method involved (convergence value).

### References


### See Also

\texttt{coef.PPSfit}, \texttt{print.PPSfit}, \texttt{plot.PPSfit}, \texttt{GoF}

### Examples

```r
data(turkey)
fit <- PPS.fit(turkey$Pop2000)
print(fit)
coef(fit)
se(fit, k = 100, parallel = FALSE)
logLik(fit)
par(mfrow = c(2,2))
plot(fit)
GoF(fit, k = 100, parallel = FALSE)
```
**print.PPSfit**

*Printing a PPSfit Object*

**Description**

It prints its argument (typically from `PPS.fit()`), returning some of the most important aspects.

**Usage**

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

**Arguments**

- `x`: a `PPSfit` object.
- `digits`: the number of digits to be printed.
- `...`: other arguments.

**References**


**See Also**

`PPS.fit`

**Examples**

```r
x <- rPPS(50, 1.2, 100, 2.3)
fit <- PPS.fit(x)
print(fit)
```

**se**

*Approximated standard errors of Pareto Positive Stable (PPS) parameter estimates*

**Description**

It approximates the standard errors of PPS parameter estimates by bootstrapping.

**Usage**

```r
se(PPSfit, k = 2000, parallel = TRUE, ncores = 2, ...)
```
Arguments

- **PPSfit**: a `PPSfit` object, typically from `PPS.fit()`.
- **k**: the number of steps in the bootstrapping procedure.
- **parallel**: a logical argument specifying if parallelization is allowed in the bootstrapping procedure.
- **ncore**: is the number of cores that we use if parallel is `TRUE` and `parallel` is `TRUE`.
- **...**: other arguments.

Details

The function simulates `k` samples from the model given in the `PPSfit` argument, fits them with the same method of estimation and uses the parameter estimates to approximate the standard errors.

Value

A list with the standard errors.

References


See Also

`PPS.fit`

Examples

```r
x <- rPPS(50, 1.2, 100, 2.3)
fit <- PPS.fit(x)
coef(fit)
se(fit, k = 50)
```

---

turkey

*Population of Turkish cities and towns*

Description

Census population of Turkish cities and towns of more than 20,000 inhabitants in 1990 and 2000.

Format

A data frame with 280 observations on the following 4 variables.

- Name: cities and towns names.
- Adm.: abbreviated province name of cities and towns.
Source

http://www.citypopulation.de/Turkey-C20.html
Index

ccoef.PPSfit, 2, 5, 10

dPPS (PPS), 7

forbes400, 3

GoF, 3, 5, 6, 10

hPPS (PPS), 7

logLik.PPSfit, 4

pareto.fit, 5
ParetoPosStable, 6
ParetoPosStable-package
(ParetoPosStable), 6

plot, 6
plot.PPSfit, 4, 5, 6, 10
pPPS (PPS), 7
PPS, 7
PPS.fit, 2, 4–7, 9, 11, 12
print.PPSfit, 5, 10, 11
print.se (se), 11

qPPS (PPS), 7

rPPS (PPS), 7

se, 11

turkey, 12