Package ‘stabledist’

August 17, 2024

Version 0.7-2

Date 2024-08-14

Title Stable Distribution Functions

Maintainer Martin Maechler <maechler@stat.math.ethz.ch>

Depends R (>= 3.1.0)

Imports stats

Suggests Matrix, fBasics, FMStable, RUnit, Rmpfr, sfsmisc , libstable4u

SuggestsNote 'libstable4u', based on 'libstableR', is said to be uniformly better both accuracy and speed wise, at least in parts.

Description Density, Probability and Quantile functions, and random number generation for (skew) stable distributions, using the parametrizations of Nolan.

License GPL (>= 2)

URL https://r-forge.r-project.org/scm/viewvc.php/pkg/stabledist/?root=rmetrics

NeedsCompilation no

Author Diethelm Wuertz [aut] (original code), Martin Maechler [aut, cre] (checks/tests; fixes, .., <https://orcid.org/0000-0002-8685-9910>), Yohan Chalabi [ctb] (namespace; admin)

Repository CRAN

Date/Publication 2024-08-17 04:50:02 UTC

Contents

StableDistribution *Stable Distribution Function*

Description

Compute density, distribution and quantile function and to generate random variates of the stable distribution.

The four functions are:

[dpqr]stable	the (skewed) stable distribution.

Three parametrizations via pm = 0, 1, or 2 *differ* in their meaning of delta and gamma, see 'Details' below. Notably the special cases of the Gaussian / normal distribution for alpha = 2 and Cauchy distribution for alpha = 1 and beta = 0 are easily matched for pm = 2.

Usage

```
dstable(x, alpha, beta, gamma = 1, delta = 0, pm = 0,
        log = FALSE,
        tol = 64*.Machine$double.eps, zeta.tol = NULL,
        subdivisions = 1000)
pstable(q, alpha, beta, gamma = 1, delta = 0, pm = 0,
        lower.tail = TRUE, log.p = FALSE, silent = FALSE,
        tol = 64*.Machine$double.eps, subdivisions = 1000)
qstable(p, alpha, beta, gamma = 1, delta = 0, pm = 0,
        lower.tail = TRUE, log.p = FALSE,
        tol = .Machine$double.eps^0.25, maxiter = 1000, trace = 0,
        integ.tol = 1e-7, subdivisions = 200)
rstable(n, alpha, beta, gamma = 1, delta = 0, pm = 0)
```

Arguments

alpha, beta, gamma, delta
: value of the index parameter alpha in the interval= $(0, 2]$; skewness parameter beta, in the range $[-1, 1]$; scale parameter gamma; and location (or 'shift') parameter delta.

n
: sample size (integer).

p
: numeric vector of probabilities.

pm
: parameterization, an integer in 0, 1, 2; by default pm=0, the 'S0' parameterization.

x, q
: numeric vector of quantiles.

log, log.p
: logical; if TRUE, probabilities p are given as log(p).

lower.tail
: logical; if TRUE (default), probabilities are $P[X \le x]$ otherwise, $P[X > x]$.

silent
: logical indicating that e.g., warnings should be suppressed when NaN is produced (because of numerical problems).

`integ.tol` positive number, the tolerance used for numerical integration, see `integrate`.

`tol` numerical tolerance,

dstable(), pstable(): used for numerical integration, see `integ.tol` above. Note that earlier versions had tighter tolerances – which seem too tight as default values.

qstable(): used for rootfinding, see `uniroot`.

`zeta.tol` (`dstable`) numerical tolerance for checking if `x` is close to $\zeta(\alpha, \beta)$. The default, `NULL` depends itself on (α, β).
As it is *experimental* and not guaranteed to remain in the future, its use is not recommended in production code. Rather e-mail the package maintainer about it.

`subdivisions` maximal number of intervals for integration, see `integrate`.

`maxiter, trace` maximal number of iterations and verboseness in `uniroot`, see there.

Details

Skew Stable Distribution:

The function uses the approach of J.P. Nolan for general stable distributions. Nolan (1997) derived expressions in form of integrals based on the characteristic function for standardized stable random variables. For `dstable` and `pstable`, these integrals are numerically evaluated using R's `integrate()` function.
"S0" parameterization [pm=0]: based on the (M) representation of Zolotarev for an alpha stable distribution with skewness beta. Unlike the Zolotarev (M) parameterization, gamma and delta are straightforward scale and shift parameters. This representation is continuous in all 4 parameters, and gives an intuitive meaning to gamma and delta that is lacking in other parameterizations.
Switching the sign of `beta` *mirrors* the distribution at the vertical axis $x = \delta$, i.e.,

$$f(x, \alpha, -\beta, \gamma, \delta, 0) = f(2\delta - x, \alpha, +\beta, \gamma, \delta, 0),$$

see the graphical example below.

"S" or "S1" parameterization [pm=1]: the parameterization used by Samorodnitsky and Taqqu in the book Stable Non-Gaussian Random Processes. It is a slight modification of Zolotarev's (A) parameterization.
"S*" or "S2" parameterization [pm=2]: a modification of the S0 parameterization which is defined so that (i) the scale gamma agrees with the Gaussian scale (standard dev.) when alpha=2 and the Cauchy scale when alpha=1, (ii) the mode is exactly at delta. For this parametrization, `stableMode(alpha,beta)` is needed.
"S3" parameterization [pm=3]: an internal parameterization, currently not available for these functions. The scale is the same as the "S2" parameterization, the shift is $-\beta * g(\alpha)$, where $g(\alpha)$ is defined in Nolan(1999).

Value

All values for the `*stable` functions are numeric vectors: `d*` returns the density, `p*` returns the distribution function, `q*` returns the quantile function, and `r*` generates random deviates.

Tail Behavior

The asymptotic behavior for large x, aka "tail behavior" for the cumulative $F(x) = P(X \le x)$ is (for $x \to \infty$)

$$1 - F(x) \sim (1+\beta) C_\alpha x^{-\alpha},$$

where $C_\alpha = \Gamma(\alpha)/\pi \sin(\alpha\pi/2)$; hence also

$$F(-x) \sim (1-\beta) C_\alpha x^{-\alpha}.$$

Differentiating $F()$ above gives

$$f(x) \sim \alpha(1+\beta) C_\alpha x^{-(1+\alpha)}.$$

Note

In the case $\beta = 1$, the distributions are "maximally skewed to the right" or simply "*extremal* stable" (Zolotarev). In that case, the package **FMStable** provides `dpq*` functions which are faster and more accurate than ours (if accuracy higher than about 6 digits is needed), see, `pEstable`.

When `alpha` is close to 1 or close to 0 ("close", e.g., meaning distance $d < 0.01$), the computations typically are numerically considerably more challenging, and the results may not be accurate.
As we plan to improve on this, *and* as it is unknown when exactly the numerical difficulties arise, we mainly only warn here in the documentation, and only in some cases, e.g. when the root finding with `uniroot` fails, signal explicit `warning()`s and may return `NaN` then.

Author(s)

Diethelm Wuertz for the original Rmetrics R-port. Many numerical improvements by Martin Maechler.

References

Chambers J.M., Mallows, C.L. and Stuck, B.W. (1976) *A Method for Simulating Stable Random Variables*, J. Amer. Statist. Assoc. **71**, 340–344.

John P. Nolan (2020) *Univariate Stable Distributions - Models for Heavy Tailed Data* Springer Series in Operations Research and Financial Engineering; doi:10.1007/9783030529154 Much earlier version of chapter 1 available at `https://edspace.american.edu/jpnolan/stable/`, see "Introduction to Stable Distributions"

Nolan J.P. (1997) Numerical calculation of stable densities and distribution functions. *Stochastic Models* **13**(4), 759–774.
Also available as '`density.ps`' from Nolan's web page.

Samoridnitsky G., Taqqu M.S. (1994); *Stable Non-Gaussian Random Processes, Stochastic Models with Infinite Variance*, Chapman and Hall, New York, 632 pages.

Weron, A., Weron R. (1999); *Computer Simulation of Levy alpha-Stable Variables and Processes*, Preprint Technical University of Wroclaw, 13 pages.

Royuela-del-Val, J., Simmross-Wattenberg, F., and Alberola-López, C. (2017) libstable: Fast, Parallel, and High-Precision Computation of α-Stable Distributions in R, C/C++, and MATLAB. *Journal of Statistical Software* **78**(1), 1–25. doi:10.18637/jss.v078.i01

See Also

the stableSlider() function from package **fBasics** for displaying densities and probabilities of these distributions, for educational purposes.

Royuela del Val et al. (2017) partly show to be uniformly better both accuracy and speed wise than our computations; While their package **libstableR** is no longer on CRAN, there is **libstable4u** derived from their implementation.

Examples

```
## stable -

## Plot stable random number series
   set.seed(1953)
   r <- rstable(n = 1000, alpha = 1.9, beta = 0.3)
   plot(r, type = "l", main = "stable: alpha=1.9 beta=0.3",
        col = "steelblue")
   grid()

## Plot empirical density and compare with true density:
   hist(r, n = 25, probability = TRUE, border = "white",
        col = "steelblue")
   x <- seq(-5, 5, by=1/16)
   lines(x, dstable(x, alpha = 1.9, beta = 0.3, tol= 1e-3), lwd = 2)

## Plot df and compare with true df:
   plot(ecdf(r), do.points=TRUE, col = "steelblue",
        main = "Probabilities:  ecdf(rstable(1000,..)) and true  cdf F()")
   rug(r)
   lines(x, pstable(q = x, alpha = 1.9, beta = 0.3),
         col="#0000FF88", lwd= 2.5)

## Switching  sign(beta)  <==> Mirror the distribution around  x == delta:
curve(dstable(x, alpha=1.2, beta =  .8, gamma = 3, delta = 2), -10, 10)
curve(dstable(x, alpha=1.2, beta = -.8, gamma = 3, delta = 2),
      add=TRUE, col=2)
## or the same
curve(dstable(2*2-x, alpha=1.2, beta = +.8, gamma = 3, delta = 2),
      add=TRUE, col=adjustcolor("gray",0.2), lwd=5)
abline(v = 2, col = "gray", lty=2, lwd=2)
axis(1, at = 2, label = expression(delta == 2))

## Compute quantiles:
   x. <- -4:4
   px <- pstable(x., alpha = 1.9, beta = 0.3)
  (qs <- qstable(px, alpha = 1.9, beta = 0.3))
stopifnot(all.equal(as.vector(qs), x., tol = 1e-5))

## Special cases: --- 1. Gaussian  alpha = 2 -----------
x. <- seq(-5,5, by=1/16)
stopifnot(
    all.equal(
```

```
        pnorm  (x.,     m=pi,    sd=1/8),
        pstable(x., delta=pi, gamma=1/8, alpha = 2, beta = 0, pm = 2) )
   ,
##                --- 2. Cauchy  alpha = 1 -----------
    all.equal(
        pcauchy(x.),
        pstable(x., delta=0, gamma=1, alpha = 1, beta = 0, pm = 2) )
)
```

StableMode *Mode of the Stable Distribution Function*

Description

Computes the mode of the stable distribution, i.e., the maximum of its density function in the "0" parametrization, i.e., the maximum x_0 of `dstable(x, alpha, beta, gamma = 1, delta = 0, pm = 0)`.

Finds the maximum of `dstable` numerically, using `optimize`.

Usage

```
stableMode(alpha, beta,
           beta.max = 1 - 1e-11,
           tol = .Machine$double.eps^0.25)
```

Arguments

`alpha, beta`	numeric parameters: value of the index parameter `alpha` in the range $(0, 2]$, and the skewness parameter `beta`, in the range $[-1, 1]$.
`beta.max`	for numerical purposes, values of beta too close to 1, are set to `beta.max`. Do not modify unless you know what you're doing.
`tol`	numerical tolerance for `optimize()`.

Value

returns a numeric value, the location of the stable mode.

Author(s)

Diethelm Wuertz for the Rmetrics R-port; minor cleanup by Martin Maechler.

See Also

For definition and the “dpqr”-functions, `StableDistribution`, also for the references.

Examples

```
## beta = 0  <==>  symmetric  <==>  mode = 0
all.equal(stableMode(alpha=1, beta=0), 0)
al.s <- c(1e-100, seq(0,2, by = 1/32)[-1])
stopifnot(vapply(al.s, function(alp)
                 stableMode(alpha=alp, beta=0), 1.) == 0)

## more interesting: asymmetric (beta != 0):
stableMode(alpha=1.2, beta=0.1)

if(stabledist:::doExtras()) { # takes 2.5 seconds
 sm0.5 <- vapply(al.s, function(AA)
                 stableMode(alpha=AA, beta= 0.5), 1.)
 plot(al.s, sm0.5, type = "o", col=2, xlab = quote(alpha), ylab="mode",
      main = quote("Mode of stable"*{}(alpha, beta == 0.5, pm==0)))
}
```

Index