Package 'FMStable'

May 7, 2026

Type Package

Title Finite Moment Stable Distributions

Version 0.1-4

Date 2022-06-03

Author Geoff Robinson

Maintainer Daniel Wilson <hmp.R.package@gmail.com>

Description Some basic procedures for dealing
with log maximally skew stable distributions, which are also
called finite moment log stable distributions.

License GPL-3

NeedsCompilation yes

Repository CRAN

Date/Publication 2022-06-06 20:10:26 UTC

Contents

Estable *Extremal or Maximally Skew Stable Distributions*

Description

Density function, distribution function, quantile function and random generation for stable distributions which are maximally skewed to the right. These distributions are called Extremal by Zolotarev (1986).

Usage

```
dEstable(x, stableParamObj, log=FALSE)
pEstable(x, stableParamObj, log=FALSE, lower.tail=TRUE)
qEstable(p, stableParamObj, log=FALSE, lower.tail=TRUE)
tailsEstable(x, stableParamObj)
```

Arguments

x	Vector of quantiles.
stableParamObj	An object of class stableParameters which describes a maximally skew stable distribution. It may, for instance, have been created by setParam or setMomentsFMstable.
p	Vector of tail probabilities.
log	Logical; if TRUE, the log density or log tail probability is returned by functions dEstable and pEstable; and logarithms of probabilities are input to function qEstable.
lower.tail	Logical; if TRUE, the lower tail probability is returned. Otherwise, the upper tail probability.

Details

The values are worked out by interpolation, with several different interpolation formulae in various regions.

Value

dEstable gives the density function; pEstable gives the distribution function or its complement; qEstable gives quantiles; tailsEstable returns a list with the following components which are all the same length as x:

density The probability density function.

F The probability distribution function. i.e. the probability of being less than or equal to x.

righttail The probability of being larger than x.

logdensity The probability density function.

logF The logarithm of the probability of being less than or equal to x.

logrighttail The logarithm of the probability of being larger than x.

References

Chambers, J.M., Mallows, C.L. and Stuck, B.W. (1976). A method for simulating stable random variables. Journal of the American Statistical Association, 71, 340–344.

See Also

If x has an extremal stable distribution then $\exp(-x)$ has a finite moment log stable distribution. The left hand tail probability computed using `pEstable` should be the same as the coresponding right hand tail probability computed using `pFMstable`.

Aspects of extremal stable distributions may also be computed (though more slowly) using `tailsGstable` with `beta=1`.

Functions for generation of random variables having stable distributions are available in package `stabledist`.

Examples

```
tailsEstable(-2:3, setMomentsFMstable(mean=1, sd=1.5, alpha=1.7))

# Compare Estable and FMstable
obj <- setMomentsFMstable(1.7, mean=.5, sd=.2)
x <- c(.001, 1, 10)
pFMstable(x,  obj, lower.tail=TRUE, log=TRUE)
pEstable(-log(x), obj, lower.tail=FALSE, log=TRUE)

x <- seq(from=-5, to=10, length=30)
plot(x, dEstable(x, setMomentsFMstable(alpha=1.5)), type="l", log="y",
  ylab="Log(density) for stable distribution",
  main="log stable distribution with alpha=1.5, mean=1, sd=1" )

x <- seq(from=-2, to=5, length=30)
plot(x, x, ylim=c(0,1), type="n", ylab="Distribution function")
for (i in 0:2)lines(x, pEstable(x,
  setParam(location=0, logscale=-.5, alpha=1.5, pm=i)), col=i+1)
legend("bottomright", legend=paste("S", 0:2, sep=""), lty=rep(1,3), col=1:3)

p <- c(1.e-10, .01, .1, .2, .5, .99, 1-1.e-10)
obj <- setMomentsFMstable(alpha=1.95)
result <- qEstable(p, obj)
pEstable(result, obj) - p

# Plot to illustrate continuity near alpha=1
y <- seq(from=-36, to=0, length=30)
logprob <- -exp(-y)
plot(0, 0, type="n", xlim=c(-25,0), ylim=c(-35, -1),
  xlab="x (M parametrization)", ylab="-log(-log(distribution function))")
for (oneminusalpha in seq(from=-.2, to=0.2, by=.02)){
  obj <- setParam(oneminusalpha=oneminusalpha, location=0, logscale=0, pm=0)
  type <- if(oneminusalpha==0) 2 else 1
  lines(qEstable(logprob, obj, log=TRUE), y, lty=type, lwd=type)
}
```

FMstable *Finite Moment Log Stable Distributions*

Description

Density function, distribution function, and quantile function for a log stable distribution with location, scale and shape parameters. For such families of distributions all moments are finite. Carr and Wu (2003) refer to such distributions as "finite moment log stable processes".

The finite moment log stable distribution is well-defined for $\alpha = 0$, when the distribution is discrete with probability concentrated at x=0 and at one other point. The distribution function may be computed by `pFMstable.alpha0`.

Usage

```
dFMstable(x, stableParamObj, log=FALSE)
pFMstable(x, stableParamObj, log=FALSE, lower.tail=TRUE)
pFMstable.alpha0(x, mean=1, sd=1, lower.tail=TRUE)
qFMstable(p, stableParamObj, lower.tail=TRUE)
tailsFMstable(x, stableParamObj)
```

Arguments

`x`	Vector of quantiles.
`stableParamObj`	An object of class `stableParameters` which describes a maximally skew stable distribution. It may, for instance, have been created by `setMomentsFMstable` or `fitGivenQuantile`.
`mean`	Mean of logstable distribution.
`sd`	Standard deviation of logstable distribution.
`p`	Vector of tail probabilities.
`log`	Logical; if `TRUE`, the log density or log tail probability is returned by functions `dFMstable` and `pFMstable`; and logarithms of probabilities are input to function `qFMstable`.
`lower.tail`	Logical; if `TRUE`, the lower tail probability is returned. Otherwise, the upper tail probability.

Details

The values are worked out by interpolation, with several different interpolation formulae in various regions.

Value

`dFMstable` gives the density function; `pFMstable` gives the distribution function or its complement; `qFMstable` gives quantiles; `tailsFMstable` returns a list with the following components which are all the same length as `x`:

density The probability density function.

F The probability distribution function. i.e. the probability of being less than or equal to x.

righttail The probability of being larger than x.

logdensity The probability density function.

logF The logarithm of the probability of being less than or equal to x.

logrighttail The logarithm of the probability of being larger than x.

References

Carr, P. and Wu, L. (2003). The Finite Moment Log Stable Process and Option Pricing. Journal of Finance, American Finance Association, vol. 58(2), pages 753-778

See Also

If a random variable X has a finite moment stable distribution then $\log(X)$ has the corresponding extremal stable distribution. The density of $\log(X)$ can be found using dEstable. Option prices can be found using callFMstable and putFMstable.

Examples

```
tailsFMstable(1:10, setMomentsFMstable(3, 1.5, alpha=1.7))

x <- c(-1, 0, 1.e-5, .001, .01, .03, seq(from=.1, to=4.5, length=100))
plot(x, pFMstable(x, setMomentsFMstable(1, 1.5, 2)), type="l" ,xlim=c(0, 4.3),
  ylim=c(0,1), ylab="Distribution function")
for (alpha in c(.03, 1:19/10)) lines(x, pFMstable(x,
  setMomentsFMstable(1, 1.5, alpha)), col=2)
lines(x, pFMstable.alpha0(x, mean=1, sd=1.5), col=3)

p <- c(1.e-10, .01, .1, .2, .5, .99, 1-1.e-10)
obj <- setMomentsFMstable(alpha=1.95)
result <- qFMstable(p, obj)
OK <- result > 0
pFMstable(result[OK], obj) - p[OK]
```

Gstable *General Stable Distributions*

Description

A procedure based on the R function integrate for computing the distribution function for stable distributions which may be skew but have standard location and scale parameters. This computation is not fast.

It is not designed to work for alpha near to 1.

Usage

```
tailsGstable(x, logabsx, alpha, oneminusalpha, twominusalpha,
  beta, betaplus1, betaminus1, parametrization, lower.tail=TRUE)
```

Arguments

x	Value (scalar).
logabsx	Logarithm of absolute value of x. Must be used when x is outside the range over which numbers can be stored. (e.g. 1.e-5000)
alpha	Value of parameter of stable distribution.
oneminusalpha	Value of alpha. This should be specified when alpha is near to 1 so that the difference from 1 is specified accurately.
twominusalpha	Value of 2 - alpha. This should be specified when alpha is near to 2 so that the difference from 2 is specified accurately.
beta	Value of parameter of stable distribution.
betaplus1	Value of beta + 1. This should be specified when beta is near to -1 so that the difference from -1 is specified accurately.
betaminus1	Value of beta - 1. This should be specified when beta is near to 1 so that the difference from 1 is specified accurately.
parametrization	Parametrization: 0 for Zolotarev's M = Nolan S0, 1 for Zolotarev's A = Nolan S1 and 2 for Zolotarev's C = Chambers, Mallows and Stuck.
lower.tail	Logical: Whether the lower tail of the distribution is of primary interest. This parameter affects whether numerical integration is used for the lower or upper tail. The other tail is computed by subtraction.

Value

Returns a list with the following components:

left.tail.prob The probability distribution function. I.e. the probability of being less than or equal to x.

right.tail.prob The probability of being larger than x.

est.error An estimate of the computational error in the previous two numbers.

message A message produced by R's standard integrate routine.

Note

This code is included mainly as an illustration of a way to deal with the problem that different parametrizations are useful in different regions. It is also of some value for checking other code, particularly since it was not used as the basis for the interpolation tables.

For the C parametrization for alpha greater than 1, the parameter beta needs to be set to -1 for the distribution to be skewed to the right.

References

Chambers, J.M., Mallows, C.L. and Stuck, B.W. (1976) A method for simulating stable random variables. Journal of the American Statistical Association 71, 340–344.

Examples

```
# Check relationship between maximally skew and other stable distributions
# in paper by J.M. Chambers, C.L. Mallows and B.W. Stuck
alpha <- 1.9
beta <- -.5
k <- 1- abs(1-alpha)
denom <- sin(pi*k)
p <- (sin(.5*pi*k * (1+beta))/denom)^(1/alpha)
q <- (sin(.5*pi*k * (1-beta))/denom)^(1/alpha)
# Probability that p S1 - q S2 < x
S1 <- setParam(alpha=1.9, location=0, logscale =log(p), pm="C")
S2 <- setParam(alpha=1.9, location=0, logscale =log(q), pm="C")
S3 <- setParam(alpha=1.9, location=0, logscale =0, pm="C")
xgiven <- 1
f <- function(x) dEstable(x, S1) * pEstable(xgiven + x, S2)
print(integrate(f, lower=-Inf, upper=Inf, rel.tol=1.e-12)$value, digits=16)
f <- function(x) dEstable(x, S3) * pEstable((xgiven + p*x)/q, S3)
print(integrate(f, lower=-Inf, upper=Inf, rel.tol=1.e-8)$value, digits=16)
direct <- tailsGstable(x=xgiven, logabsx=log(xgiven),alpha=alpha,
  beta=beta, parametrization=2)
print(direct$left.tail.prob, digits=16)

# Compare Estable and Gstable
# List fractional discrepancies
disc <- function(tol){
  for(pm in pms) for (a in alphas) for(x in xs) {
    lx <- log(abs(x))
    beta <- if(pm==2 && a > 1) -1 else 1
    if(x > 0 || a > 1){
      a1 <- pEstable(x, setParam(alpha=a, location=0, logscale=0, pm=pm))
      a2 <- tailsGstable(x=x, logabsx=lx, alpha=a, beta=beta,
        parametrization=pm)$left.tail.prob
      print(paste("parametrization=", pm, "alpha=", a,"x=", x,
        "Frac disc=", a1/a2-1), quote=FALSE)
    }
  }
}

alphas <- c(.3, .8, 1.1, 1.5, 1.9)
pms <- 0:2
xs <- c(-2, .01, 4.3)
disc()
```

`impliedVolatility` *Computations Regarding Value of Options for Log Normal Distributions*

Description

Computes values of European-style call and put options over assets whose future price is expected to follow a log normal distribution.

Usage

```
BSOptionValue(spot, strike, expiry, volatility,
  intRate=0, carryCost=0, Call=TRUE)
ImpliedVol(spot, strike, expiry, price, intRate=0, carryCost=0,
  Call=TRUE, ImpliedVolLowerBound=.01, ImpliedVolUpperBound=1, tol=1.e-9)
lnorm.param(mean, sd)
```

Arguments

`spot`	The current price of a security.
`strike`	The strike price for an option.
`expiry`	The time when an option may be exercised. (We are only dealing with European options which have a single date on which they may be exercised.)
`volatility`	The volatility of the price of a security per unit time. This is the standard deviation of the logarithm of price.
`price`	The price for an option. This is used as an input parameter when computing the implied volatility.
`intRate`	The interest rate.
`carryCost`	The carrying cost for a security. This may be negative when a security is expected to pay a dividend.
`Call`	Logical: Whether the option for which a `price` is given is a call option.
`ImpliedVolLowerBound`	Lower bound used when searching for the inplied volatility.
`ImpliedVolUpperBound`	Upper bound used when searching for the inplied volatility.
`tol`	Tolerance specifying accuracy of search for implied volatility.
`mean`	The mean of a quantity which has a lognormal distribution.
`sd`	The standard deviation of a quantity which has a lognormal distribution.

Details

The lognormal distribution is the limit of finite moment log stable distributions as `alpha` tends to 2. The function `lnorm.param` finds the mean and standard deviation of a lognormal distribution on the log scale given the mean and standard deviation on the raw scale. The function `BSOptionValue` finds the value of a European call or put option. The function `ImpliedVol` allows computation of the implied volatility, which is the volatility on the logarithmic scale which matches the value of an option to a specified price.

Value

`impVol` returns the implied volatility when the value of options is computed using a finite moment log stable distribution. `approx.impVol` returns an approximation to the implied volatility. `lnorm.param` returns the mean and standard deviation of the underlying normal distribution.

See Also

Option prices computed using the log normal model can be compared to those computed for the finite moment log stable model using `putFMstable` and `callFMstable`.

Examples

```
lnorm.param(mean=5, sd=.8)
BSOptionValue(spot=4, strike=c(4, 4.5), expiry=.5, volatility=.15)
ImpliedVol(spot=4, strike=c(4, 4.5), expiry=.5, price=c(.18,.025))
```

---

moments *Convolutions of Finite Moment Log Stable Distributions and the Moments of such Distributions*

Description

If $X_1, \ldots, X_n$ are independent random variables with the same stable distribution then $X_1 + \cdots + X_n$ has a stable distribution with the same alpha. The function `iidcombine` allows the parameters of the resulting stable distribution to be computed. Because stable distributions are infinitely divisible, it is also easy to find the parameters describing the distribution of X_1 from the parameters describing the distribution of $X_1 + \cdots + X_n$.

Convolutions of maximally skew stable distributions correspond to products of logstable distributions. The raw moments of these distributions (i.e. moments about zero, not moments about the mean) can be readily computed using the function `moments`. Note that the raw moments of the convolution of two independent distributions are the products of the corresponding moments of the component distributions, so the accuracy of `iidcombine` can be checked by using `moments`.

Usage

```
iidcombine(n, stableParamObj)
moments(powers, stableParamObj, log=FALSE)
```

Arguments

n	Number of random variables to be convoluted. May be any positive number.
powers	Raw moments of logstable distributions to be computed.
stableParamObj	An object of class stableParameters which describes a maximally skew stable distribution.
log	Logical; if TRUE, the logarithms of moments are returned.

Value

The value returned by iidcombine is another object of class stableParameters. The value returned by moments is a numeric vector giving the values of the specified raw moments.

See Also

Objects of class stableParameters can be created using functions such as setParam. The taking of convolutions is sometimes associated with the computing of values of options using functions such as callFMstable.

Examples

```
yearDsn <- fitGivenQuantile(mean=1, sd=2, prob=.7, value=.1)
upper <- exp(-yearDsn$location) # Only sensible for alpha<.5
x <- exp(seq(from=log(.0001), to=log(upper), length=50))
plot(x, pFMstable(x, yearDsn), type="l", ylim=c(.2,1), lwd=2, xlab="Price",
  ylab="Distribution function of future price")
half <- iidcombine(.5, yearDsn)
lines(x, pFMstable(x, half), lty=2, lwd=2)
quarter <- iidcombine(.25, yearDsn)
lines(x, pFMstable(x, quarter), lty=3, lwd=2)
legend("bottomright", legend=paste(c("1","1/2","1/4"),"year"), lty=c(1,2,3),
  lwd=c(2,2,2))
moments(1:2, yearDsn)
moments(1:2, half)
moments(1:2, quarter)

# Check logstable against lognormal
iidcombine(2, setMomentsFMstable(.5, .2, alpha=2))
p <- lnorm.param(.5, .2)
2*p$meanlog # Gives the mean
log(p$sdlog) # Gives the logscale
```

---

optionValues *Values of Options over Finite Moment Log Stable Distributions*

Description

Computes values of European-style call and put options over assets whose future price is expected to follow a finite moment log stable distribution.

Usage

```
putFMstable(strike, paramObj, rel.tol=1.e-10)
callFMstable(strike, paramObj, rel.tol=1.e-10)
optionsFMstable(strike, paramObj, rel.tol=1.e-10)
```

Arguments

strike	The strike price for an option.
paramObj	An object of class stableParameters which describes a maximally skew stable distribution. This is the distribution which describes possible prices for the underlying security at the time of expiry.
rel.tol	The relative tolerance used for numerical integration for finding option values.

Value

optionsFMstable returns a list containing the values of put options and the values of call options.

Note

When comparing option values based on finite moment log stable distributions with ones based on log normal distributions, remember that the interest rate and holding cost have been ignored here.

Rather than using functions putFMstable and callFMstable for options that are extremely in-the-money (i.e. the options are almost certain to be exercised), the values of such options can be computed more accurately by first computing the value of the out-of-the-money option and then using the relationship spot + put = call + strike. This is done by function optionsFMstable.

See Also

An example of how an object of class stableParameters may be created is by setParam. Procedures for dealing with the log normal model for options pricing include BSOptionValue.

Examples

```
paramObj <- setMomentsFMstable(mean=10, sd=1.5, alpha=1.8)
putFMstable(c(10,7), paramObj)
callFMstable(c(10,7), paramObj)
optionsFMstable(8:12, paramObj)
# Note that call - put = mean - strike

# Values of some extreme put options
paramObj <- setMomentsFMstable(mean=1, sd=1.5, alpha=0.02)
putFMstable(1.e-200, paramObj)
putFMstable(1.e-100, paramObj)
pFMstable(1.e-100, paramObj)
putFMstable(1.e-50, paramObj)

# Asymptotic behaviour
logmlogx <- seq(from=2, to=6, length=30)
logx <- -exp(logmlogx)
```

```
x <- exp(logx)
plot(logmlogx , putFMstable(x, paramObj)/(x*pFMstable(x, paramObj)), type="l")

# Work out the values of some options using FMstable model
spot <- 20
strikes <- c(15,18:20, 20:24, 18:20, 20:23)
isCall <- rep(c(FALSE,TRUE,FALSE,TRUE), c(4,5,3,4))
expiry <- rep(c(.2, .5), c(9,7))
# Distributions for 0.2 and 0.5 of a year given distribution describing
#   multiplicative change in price over a year:
annual <- fitGivenQuantile(mean=1, sd=.2, prob=2.e-4, value=.01)
timep2 <- iidcombine(.2, annual)
timep5 <- iidcombine(.5, annual)
imp.vols <- prices <- rep(NA, length(strikes))
use <- isCall & expiry==.2
prices[use] <- callFMstable(strikes[use]/spot, timep2) * spot
use <- !isCall & expiry==.2
prices[use] <- putFMstable(strikes[use]/spot, timep2) * spot
use <- isCall & expiry==.5
prices[use] <- callFMstable(strikes[use]/spot, timep5) * spot
use <- !isCall & expiry==.5
prices[use] <- putFMstable(strikes[use]/spot, timep5) * spot
# Compute implied volatilities.
imp.vols[isCall] <- ImpliedVol(spot=spot, strike=strikes[isCall],
  expiry=expiry[isCall], price=prices[isCall], Call=TRUE)
imp.vols[!isCall] <- ImpliedVol(spot=spot, strike=strikes[!isCall],
  expiry=expiry[!isCall], price=prices[!isCall], Call=FALSE)

# List values of options
cbind(strikes, expiry, isCall, prices, imp.vols)

# Can the distribution be recovered from the values of the options?
discrepancy <- function(alpha, cv){
  annual.fit <- setMomentsFMstable(mean=1, sd=cv, alpha=alpha)
  timep2.fit <- iidcombine(.2, annual.fit)
  timep5.fit <- iidcombine(.5, annual.fit)
  prices.fit <- rep(NA, length(strikes))
  use <- isCall & expiry==.2
  prices.fit[use] <- callFMstable(strikes[use]/spot, timep2.fit) * spot
  use <- !isCall & expiry==.2
  prices.fit[use] <- putFMstable(strikes[use]/spot, timep2.fit) * spot
  use <- isCall & expiry==.5
  prices.fit[use] <- callFMstable(strikes[use]/spot, timep5.fit) * spot
  use <- !isCall & expiry==.5
  prices.fit[use] <- putFMstable(strikes[use]/spot, timep5.fit) * spot
  return(sum((prices.fit - prices)^2))
}
# Search on scales of log(2-alpha) and log(cv)
d <- function(param) discrepancy(2-exp(param[1]), exp(param[2]))
system.time(result <- nlm(d, p=c(-2,-1.5)))
# Estimated alpha
2-exp(result$estimate[1])
# Estimated cv
```

```
exp(result$estimate[2])

# Searching just for best alpha
d <- function(param) discrepancy(param, .2)
system.time(result <- optimize(d, lower=1.6, upper=1.98))
# Estimated alpha
result$minimum
```

---

stableParameters *Setting up Parameters to Describe both Extremal Stable Distributions and Finite Moment Log Stable Distributions*

Description

Functions which create stable distributions having specified properties. Each of these functions takes scalar arguments and produces a description of a single stable distribution.

Usage

```
setParam(alpha, oneminusalpha, twominusalpha, location, logscale, pm)
setMomentsFMstable(mean=1, sd=1, alpha, oneminusalpha, twominusalpha)
fitGivenQuantile(mean, sd, prob, value, tol=1.e-10)
matchQuartiles(quartiles, alpha, oneminusalpha, twominusalpha, tol=1.e-10)
```

Arguments

alpha	Stable distribution parameter which must be a single value satisfying $0 < \alpha <= 2$.
oneminusalpha	Alternative specification of stable distribution parameter: Specify 1-alpha.
twominusalpha	Alternative specification of stable distribution parameter: Specify 2-alpha.
location	Location parameter of stable distribution.
logscale	Logarithm of scale parameter of stable distribution.
pm	Parametrization used in specifying stable distribution which is maximally skewed to the right. Allowable values are 0, "S0", "M", 1, "S1", "A", 2, "CMS" or "C" for some common parametrizations.
mean	Mean of logstable distribution.
sd	Standard deviation of logstable distribution.
value, prob	Required probability distribution function (> 0) for a logstable distribution at a value (> 0).
quartiles	Vector of two quartiles to be matched by a logstable distribution.
tol	Tolerance for matching of quantile or quartiles.

Details

The parametrizations used internally by this package are Nolan's "S0" (or Zolotarev's "M") parametrization when `alpha` >= 0.5, and the Zolotarev's "C" parametrization (which was used by Chambers, Mallows and Struck (1976) when `alpha` < 0.5.

By using objects of class `stableParameters` to store descriptions of stable distributions, it will generally be possible to write code in a way which is not affected by the internal representation. Such usage is encouraged.

Value

Each of the functions described here produces an object of class `stableParameters` which describes a maximally skew stable distribution. Its components include at least the shape parameter `alpha`, a location parameter referred to as `location` and the logarithm of a scale parameter referred to as `logscale`.

Currently objects of this class also store information about how they were created, as well as storing the numbers 1-`alpha` and 2-`alpha` in order to improve computational precision.

References

Chambers, J.M., Mallows, C.L. and Stuck, B.W. (1976). A method for simulating stable random variables. Journal of the American Statistical Association, Vol. 71, 340–344.

Nolan, J.P. (2012). Stable Distributions. ISBN 9780817641597

Zolotarev, V.M. (1986). One-Dimensional Stable Distributions. Amer. Math. Soc. Transl. of Math. Monographs, Vol. 65. Amer Math. Soc., Providence, RI. (Original Russian version was published in 1983.)

See Also

Extremal stable distributions with parameters set up using these procedures can be used by functions such as `dEstable`. The corresponding finite moment log stable distributions can be dealt with using functions such as `dFMstable`.

Examples

```
setParam(alpha=1.5, location=1, logscale=-.6, pm="M")
setParam(alpha=.4, location=1, logscale=-.6, pm="M")
setMomentsFMstable(alpha=1.7, mean=.5, sd=.2)
fitGivenQuantile(mean=5, sd=1, prob=.001, value=.01, tol=1.e-10)
fitGivenQuantile(mean=20, sd=1, prob=1.e-20, value=1, tol=1.e-24)
matchQuartiles(quartiles=c(9,11), alpha=1.8)
```

Index