Package 'sadists'

March 20, 2017

Maintainer Steven E. Pav <shabbychef@gmail.com>

Version 0.2.3

Date 2017-03-19

License LGPL-3

Title Some Additional Distributions

BugReports https://github.com/shabbychef/sadists/issues

Description Provides the density, distribution, quantile and generation functions of some obscure probability distributions, including the doubly non-central t, F, Beta, and Eta distributions; the lambda-prime and K-prime; the upsilon distribution; the (weighted) sum of non-central chi-squares to a power; the (weighted) sum of log non-central chi-squares; the product of non-central chi-squares to powers; the product of doubly non-central F variables; the product of independent normals.

Depends R (>= 3.0.2)

Imports PDQutils (>= 0.1.1), hypergeo, orthopolynom

Suggests shiny, testthat, ggplot2, xtable, knitr

URL https://github.com/shabbychef/sadists

VignetteBuilder knitr

Collate 'cumulants.r' 'dnbeta.r' 'dneta.r' 'dnf.r' 'dnt.r' 'kprime.r' 'lambdap.r' 'moments.r' 'prodchisqpow.r' 'proddnf.r' 'prodnormal.r' 'runExample.r' 'sadists.r' 'sumchisqpow.r' 'sumlogchisq.r' 'upsilon.r' 'utils.r'

RoxygenNote 5.0.1

NeedsCompilation no

Author Steven E. Pav [aut, cre]

Repository CRAN

Date/Publication 2017-03-20 06:13:11 UTC

R topics documented:

`dnbeta` *The doubly non-central Beta distribution.*

Description

Density, distribution function, quantile function and random generation for the doubly non-central Beta distribution.

Usage

```
ddnbeta(x, df1, df2, ncp1, ncp2, log = FALSE, order.max=6)

pdnbeta(q, df1, df2, ncp1, ncp2, lower.tail = TRUE, log.p = FALSE, order.max=6)

qdnbeta(p, df1, df2, ncp1, ncp2, lower.tail = TRUE, log.p = FALSE, order.max=6)

rdnbeta(n, df1, df2, ncp1, ncp2)
```

Arguments

`x, q`	vector of quantiles.
`df1, df2`	the degrees of freedom for the numerator and denominator. We do *not* recycle these versus the `x,q,p,n`.
`ncp1, ncp2`	the non-centrality parameters for the numerator and denominator. We do *not* recycle these versus the `x,q,p,n`.
`log`	logical; if TRUE, densities f are given as $\log(f)$.

order.max	the order to use in the approximate density, distribution, and quantile computations, via the Gram-Charlier, Edeworth, or Cornish-Fisher expansion.
p	vector of probabilities.
n	number of observations.
log.p	logical; if TRUE, probabilities p are given as $\log(p)$.
lower.tail	logical; if TRUE (default), probabilities are $P[X \leq x]$, otherwise, $P[X > x]$.

Details

Suppose $x_i \sim \chi^2(\delta_i, \nu_i)$ be independent non-central chi-squares for $i = 1, 2$. Then

$$Y = \frac{x_1}{x_1 + x_2}$$

takes a doubly non-central Beta distribution with degrees of freedom ν_1, ν_2 and non-centrality parameters δ_1, δ_2.

Value

ddnbeta gives the density, pdnbeta gives the distribution function, qdnbeta gives the quantile function, and rdnbeta generates random deviates.

Invalid arguments will result in return value NaN with a warning.

Note

The PDF, CDF, and quantile function are approximated, via the Edgeworth or Cornish Fisher approximations, which may not be terribly accurate in the tails of the distribution. You are warned.

The distribution parameters are *not* recycled with respect to the x, p, q or n parameters, for, respectively, the density, distribution, quantile and generation functions. This is for simplicity of implementation and performance. It is, however, in contrast to the usual R idiom for dpqr functions.

Author(s)

Steven E. Pav <shabbychef@gmail.com>

See Also

(doubly non-central) F distribution functions, ddnf, pdnf, qdnf, rdnf.

Examples

```
rv <- rdnbeta(500, df1=100,df2=500,ncp1=1.5,ncp2=12)
d1 <- ddnbeta(rv, df1=100,df2=500,ncp1=1.5,ncp2=12)
## Not run:
plot(rv,d1)

## End(Not run)
p1 <- ddnbeta(rv, df1=100,df2=500,ncp1=1.5,ncp2=12)
# should be nearly uniform:
## Not run:
```

```
plot(ecdf(p1))

## End(Not run)
q1 <- qdnbeta(ppoints(length(rv)), df1=100,df2=500,ncp1=1.5,ncp2=12)
## Not run:
qqplot(x=rv,y=q1)

## End(Not run)
```

---

dneta *The doubly non-central Eta distribution.*

Description

Density, distribution function, quantile function and random generation for the doubly non-central Eta distribution.

Usage

```
ddneta(x, df, ncp1, ncp2, log = FALSE, order.max=6)

pdneta(q, df, ncp1, ncp2, lower.tail = TRUE, log.p = FALSE, order.max=6)

qdneta(p, df, ncp1, ncp2, lower.tail = TRUE, log.p = FALSE, order.max=6)

rdneta(n, df, ncp1, ncp2)
```

Arguments

x, q	vector of quantiles.
df	the degrees of freedom for the denominator chi square. We do *not* recycle this versus the x,q,p,n.
ncp1, ncp2	the non-centrality parameters for the numerator and denominator. We do *not* recycle these versus the x,q,p,n.
log	logical; if TRUE, densities f are given as $\log(f)$.
order.max	the order to use in the approximate density, distribution, and quantile computations, via the Gram-Charlier, Edeworth, or Cornish-Fisher expansion.
p	vector of probabilities.
n	number of observations.
log.p	logical; if TRUE, probabilities p are given as $\log(p)$.
lower.tail	logical; if TRUE (default), probabilities are $P[X \leq x]$, otherwise, $P[X > x]$.

Details

Suppose Z is a normal with mean δ_1, independent of $X \sim \chi^2(\delta_2, \nu_2)$, a non-central chi-square with ν_2 degrees of freedom and non-centrality parameter δ_2. Then

$$Y = \frac{Z}{\sqrt{Z^2 + X}}$$

takes a doubly non-central Eta distribution with ν_2 degrees of freedom and non-centrality parameters δ_1, δ_2. The *square* of a doubly non-central Eta is a doubly non-central Beta variate.

Value

`ddneta` gives the density, `pdneta` gives the distribution function, `qdneta` gives the quantile function, and `rdneta` generates random deviates.

Invalid arguments will result in return value `NaN` with a warning.

Note

The PDF, CDF, and quantile function are approximated, via the Edgeworth or Cornish Fisher approximations, which may not be terribly accurate in the tails of the distribution. You are warned.

The distribution parameters are *not* recycled with respect to the `x`, `p`, `q` or `n` parameters, for, respectively, the density, distribution, quantile and generation functions. This is for simplicity of implementation and performance. It is, however, in contrast to the usual R idiom for dpqr functions.

Author(s)

Steven E. Pav <shabbychef@gmail.com>

See Also

(doubly non-central) t distribution functions, `ddnt`, `pdnt`, `qdnt`, `rdnt`.

(doubly non-central) Beta distribution functions, `ddnbeta`, `pdnbeta`, `qdnbeta`, `rdnbeta`.

Examples

```
rv <- rdneta(500, df=100,ncp1=1.5,ncp2=12)
d1 <- ddneta(rv, df=100,ncp1=1.5,ncp2=12)
## Not run:
plot(rv,d1)

## End(Not run)
p1 <- ddneta(rv, df=100,ncp1=1.5,ncp2=12)
# should be nearly uniform:
## Not run:
plot(ecdf(p1))

## End(Not run)
q1 <- qdneta(ppoints(length(rv)), df=100,ncp1=1.5,ncp2=12)
## Not run:
qqplot(x=rv,y=q1)
```

```
## End(Not run)
```

---

dnf *The doubly non-central F distribution.*

Description

Density, distribution function, quantile function and random generation for the doubly non-central F distribution.

Usage

```
ddnf(x, df1, df2, ncp1, ncp2, log = FALSE, order.max=6)

pdnf(q, df1, df2, ncp1, ncp2, lower.tail = TRUE, log.p = FALSE, order.max=6)

qdnf(p, df1, df2, ncp1, ncp2, lower.tail = TRUE, log.p = FALSE, order.max=6)

rdnf(n, df1, df2, ncp1, ncp2)
```

Arguments

`x, q`	vector of quantiles.
`df1, df2`	the degrees of freedom for the numerator and denominator. We do *not* recycle these versus the `x,q,p,n`.
`ncp1, ncp2`	the non-centrality parameters for the numerator and denominator. We do *not* recycle these versus the `x,q,p,n`.
`log`	logical; if TRUE, densities f are given as $\log(f)$.
`order.max`	the order to use in the approximate density, distribution, and quantile computations, via the Gram-Charlier, Edeworth, or Cornish-Fisher expansion.
`p`	vector of probabilities.
`n`	number of observations.
`log.p`	logical; if TRUE, probabilities p are given as $\log(p)$.
`lower.tail`	logical; if TRUE (default), probabilities are $P[X \leq x]$, otherwise, $P[X > x]$.

Details

Suppose $x_i \sim \chi^2(\delta_i, \nu_i)$ be independent non-central chi-squares for $i = 1, 2$. Then

$$Y = \frac{x_1/\nu_1}{x_2/\nu_2}$$

takes a doubly non-central F distribution with degrees of freedom ν_1, ν_2 and non-centrality parameters δ_1, δ_2.

Value

ddnf gives the density, pdnf gives the distribution function, qdnf gives the quantile function, and rdnf generates random deviates.

Invalid arguments will result in return value NaN with a warning.

Note

The PDF, CDF, and quantile function are approximated, via the Edgeworth or Cornish Fisher approximations, which may not be terribly accurate in the tails of the distribution. You are warned.

The distribution parameters are *not* recycled with respect to the x, p, q or n parameters, for, respectively, the density, distribution, quantile and generation functions. This is for simplicity of implementation and performance. It is, however, in contrast to the usual R idiom for dpqr functions.

Author(s)

Steven E. Pav <shabbychef@gmail.com>

See Also

(singly non-central) F distribution functions, df, pf, qf, rf.

Examples

```
rv <- rdnf(500, df1=100,df2=500,ncp1=1.5,ncp2=12)
d1 <- ddnf(rv, df1=100,df2=500,ncp1=1.5,ncp2=12)
## Not run:
plot(rv,d1)

## End(Not run)
p1 <- ddnf(rv, df1=100,df2=500,ncp1=1.5,ncp2=12)
# should be nearly uniform:
## Not run:
plot(ecdf(p1))

## End(Not run)
q1 <- qdnf(ppoints(length(rv)), df1=100,df2=500,ncp1=1.5,ncp2=12)
## Not run:
qqplot(x=rv,y=q1)

## End(Not run)
```

---

dnt *The doubly non-central t distribution.*

Description

Density, distribution function, quantile function and random generation for the doubly non-central t distribution.

Usage

```
ddnt(x, df, ncp1, ncp2, log = FALSE, order.max=6)

pdnt(q, df, ncp1, ncp2, lower.tail = TRUE, log.p = FALSE, order.max=6)

qdnt(p, df, ncp1, ncp2, lower.tail = TRUE, log.p = FALSE, order.max=6)

rdnt(n, df, ncp1, ncp2)
```

Arguments

x, q	vector of quantiles.
df	the degrees of freedom for the denominator, ν. We do *not* recycle these versus the x,q,p,n.
ncp1, ncp2	the non-centrality parameters for the numerator and denominator, respectively, μ and θ We do *not* recycle these versus the x,q,p,n.
log	logical; if TRUE, densities f are given as $\log(f)$.
order.max	the order to use in the approximate density, distribution, and quantile computations, via the Gram-Charlier, Edeworth, or Cornish-Fisher expansion.
p	vector of probabilities.
n	number of observations.
log.p	logical; if TRUE, probabilities p are given as $\log(p)$.
lower.tail	logical; if TRUE (default), probabilities are $P[X \leq x]$, otherwise, $P[X > x]$.

Details

Let $Z \sim \mathcal{N}(\mu, 1)$ independently of $X \sim \chi^2(\theta, \nu)$. The random variable

$$T = \frac{Z}{\sqrt{X/\nu}}$$

takes a *doubly non-central t distribution* with parameters ν, μ, θ.

Value

ddnt gives the density, pdnt gives the distribution function, qdnt gives the quantile function, and rdnt generates random deviates.

Invalid arguments will result in return value NaN with a warning.

Note

The PDF, CDF, and quantile function are approximated, via the Edgeworth or Cornish Fisher approximations, which may not be terribly accurate in the tails of the distribution. You are warned.

The distribution parameters are *not* recycled with respect to the x, p, q or n parameters, for, respectively, the density, distribution, quantile and generation functions. This is for simplicity of implementation and performance. It is, however, in contrast to the usual R idiom for dpqr functions.

Author(s)

Steven E. Pav <shabbychef@gmail.com>

References

Krishnan, Marakatha. "Series Representations of the Doubly Noncentral t-Distribution." Journal of the American Statistical Association 63, no. 323 (1968): 1004-1012.

See Also

t distribution functions, dt, pt, qt, rt

Examples

```
rvs <- rdnt(128, 20, 1, 1)
dvs <- ddnt(rvs, 20, 1, 1)
pvs.H0 <- pdnt(rvs, 20, 0, 1)
pvs.HA <- pdnt(rvs, 20, 1, 1)
## Not run:
plot(ecdf(pvs.H0))
plot(ecdf(pvs.HA))

## End(Not run)
# compare to singly non-central
dv1 <- ddnt(1, df=10, ncp1=5, ncp2=0, log=FALSE)
dv2 <- dt(1, df=10, ncp=5, log=FALSE)
pv1 <- pdnt(1, df=10, ncp1=5, ncp2=0, log.p=FALSE)
pv11 <- pdnt(1, df=10, ncp1=5, ncp2=0.001, log.p=FALSE)
v2 <- pt(1, df=10, ncp=5, log.p=FALSE)

q1 <- qdnt(pv1, df=10, ncp1=5, ncp2=0, log.p=FALSE)
```

---

kprime *The K prime distribution.*

Description

Density, distribution function, quantile function and random generation for the K prime distribution.

Usage

```
dkprime(x, v1, v2, a, b = 1, order.max=6, log = FALSE)

pkprime(q, v1, v2, a, b = 1, order.max=6, lower.tail = TRUE, log.p = FALSE)

qkprime(p, v1, v2, a, b = 1, order.max=6, lower.tail = TRUE, log.p = FALSE)

rkprime(n, v1, v2, a, b = 1)
```

Arguments

x, q	vector of quantiles.
v1	the degrees of freedom in the numerator chisquare. When (positive) infinite, we recover a non-central t distribution with v2 degrees of freedom and non-centrality parameter a, scaled by b. This is not recycled against the x,q,p,n.
v2	the degrees of freedom in the denominator chisquare. When equal to infinity, we recover the Lambda prime distribution. This is not recycled against the x,q,p,n.
a	the non-centrality scaling parameter. When equal to zero, we recover the (central) t distribution. This is not recycled against the x,q,p,n.
b	the scaling parameter. This is not recycled against the x,q,p,n.
order.max	the order to use in the approximate density, distribution, and quantile computations, via the Gram-Charlier, Edeworth, or Cornish-Fisher expansion.
log	logical; if TRUE, densities f are given as $\log(f)$.
p	vector of probabilities.
n	number of observations.
log.p	logical; if TRUE, probabilities p are given as $\log(p)$.
lower.tail	logical; if TRUE (default), probabilities are $P[X \leq x]$, otherwise, $P[X > x]$.

Details

Suppose $y \sim \chi^2(\nu_1)$, and $x \sim t\left(\nu_2, a\sqrt{y/\nu_1}/b\right)$. Then the random variable

$$T = bx$$

takes a K prime distribution with parameters ν_1, ν_2, a, b. In Lecoutre's terminology, $T \sim K'_{\nu_1,\nu_2}(a, b)$

Equivalently, we can think of

$$T = \frac{bZ + a\sqrt{\chi^2_{\nu_1}/\nu_1}}{\sqrt{\chi^2_{\nu_2}/\nu_2}}$$

where Z is a standard normal, and the normal and the (central) chi-squares are independent of each other. When $a = 0$ we recover a central t distribution; when $\nu_1 = \infty$ we recover a rescaled non-central t distribution; when $b = 0$, we get a rescaled square root of a central F distribution; when $\nu_2 = \infty$, we recover a Lambda prime distribution.

Value

dkprime gives the density, pkprime gives the distribution function, qkprime gives the quantile function, and rkprime generates random deviates.

Invalid arguments will result in return value NaN with a warning.

Note

The PDF, CDF, and quantile function are approximated, via the Edgeworth or Cornish Fisher approximations, which may not be terribly accurate in the tails of the distribution. You are warned.

The distribution parameters are *not* recycled with respect to the x, p, q or n parameters, for, respectively, the density, distribution, quantile and generation functions. This is for simplicity of implementation and performance. It is, however, in contrast to the usual R idiom for dpqr functions.

Author(s)

Steven E. Pav <shabbychef@gmail.com>

References

Lecoutre, Bruno. "Two Useful distributions for Bayesian predictive procedures under normal models." Journal of Statistical Planning and Inference 79, no. 1 (1999): 93-105.

Poitevineau, Jacques, and Lecoutre, Bruno. "Implementing Bayesian predictive procedures: The K-prime and K-square distributions." Computational Statistics and Data Analysis 54, no. 3 (2010): 724-731. http://arxiv.org/abs/1003.4890v1

See Also

t distribution functions, dt, pt, qt, rt, lambda prime distribution functions, dlambdap, plambdap, qlambdap, rlambda

Examples

```
d1 <- dkprime(1, 50, 20, a=0.01)
d2 <- dkprime(1, 50, 20, a=0.0001)
d3 <- dkprime(1, 50, 20, a=0)
d4 <- dkprime(1, 10000, 20, a=1)
d5 <- dkprime(1, Inf, 20, a=1)
```

---

lambdap *The lambda prime distribution.*

Description

Density, distribution function, quantile function and random generation for the lambda prime distribution.

Usage

```
dlambdap(x, df, t, log = FALSE, order.max=6)

plambdap(q, df, t, lower.tail = TRUE, log.p = FALSE, order.max=6)

qlambdap(p, df, t, lower.tail = TRUE, log.p = FALSE, order.max=6)

rlambdap(n, df, t)
```

Arguments

x, q	vector of quantiles.
df	the degrees of freedom in the chi square. This is not recycled against the x,q,p,n.
t	the scaling parameter on the chi. This is not recycled against the x,q,p,n.
log	logical; if TRUE, densities f are given as $\log(f)$.
order.max	the order to use in the approximate density, distribution, and quantile computations, via the Gram-Charlier, Edeworth, or Cornish-Fisher expansion.
p	vector of probabilities.
n	number of observations.
log.p	logical; if TRUE, probabilities p are given as $\log(p)$.
lower.tail	logical; if TRUE (default), probabilities are $P[X \leq x]$, otherwise, $P[X > x]$.

Details

Suppose $y \sim \chi^2(\nu)$, and Z is a standard normal.

$$T = Z + t\sqrt{y/\nu}$$

takes a lambda prime distribution with parameters ν, t. A lambda prime random variable can be viewed as a confidence level on a non-central t because

$$t = \frac{Z' + T}{\sqrt{y/\nu}}$$

Value

dlambdap gives the density, plambdap gives the distribution function, qlambdap gives the quantile function, and rlambdap generates random deviates.

Invalid arguments will result in return value NaN with a warning.

Note

The PDF, CDF, and quantile function are approximated, via the Edgeworth or Cornish Fisher approximations, which may not be terribly accurate in the tails of the distribution. You are warned.

The distribution parameters are *not* recycled with respect to the x, p, q or n parameters, for, respectively, the density, distribution, quantile and generation functions. This is for simplicity of implementation and performance. It is, however, in contrast to the usual R idiom for dpqr functions.

Author(s)

Steven E. Pav <shabbychef@gmail.com>

References

Lecoutre, Bruno. "Another look at confidence intervals for the noncentral t distribution." Journal of Modern Applied Statistical Methods 6, no. 1 (2007): 107–116. http://www.univ-rouen.fr/LMRS/Persopage/Lecoutre/telechargements/Lecoutre_Another_look-JMSAM2007_6(1).pdf

Lecoutre, Bruno. "Two useful distributions for Bayesian predictive procedures under normal models." Journal of Statistical Planning and Inference 79 (1999): 93–105.

See Also

t distribution functions, dt, pt, qt, rt, K prime distribution functions, dkprime, pkprime, qkprime, rkprime, upsilon distribution functions, dupsilon, pupsilon, qupsilon, rupsilon,

Examples

```
rv <- rlambdap(100, 50, t=0.01)
d1 <- dlambdap(1, 50, t=0.01)
pv <- plambdap(rv, 50, t=0.01)
qv <- qlambdap(ppoints(length(rv)), 50, t=1)
```

---

prodchisqpow *The product of (non-central) chi-squares raised to powers distribution.*

Description

Density, distribution function, quantile function and random generation for the distribution of the product of non-central chi-squares taken to powers.

Usage

```
dprodchisqpow(x, df, ncp=0, pow=1, log = FALSE, order.max=5)

pprodchisqpow(q, df, ncp=0, pow=1, lower.tail = TRUE, log.p = FALSE, order.max=5)

qprodchisqpow(p, df, ncp=0, pow=1, lower.tail = TRUE, log.p = FALSE, order.max=5)

rprodchisqpow(n, df, ncp=0, pow=1)
```

Arguments

x, q	vector of quantiles.
df	the vector of degrees of freedom. This is recycled against the ncp, pow, but not against the x,q,p,n.
ncp	the vector of non-centrality parameters. This is recycled against the df, pow, but not against the x,q,p,n.

pow	the vector of the power parameters. This is recycled against the df, ncp, but not against the x,q,p,n.
log	logical; if TRUE, densities f are given as $\log(f)$.
order.max	the order to use in the approximate density, distribution, and quantile computations, via the Gram-Charlier, Edeworth, or Cornish-Fisher expansion.
p	vector of probabilities.
n	number of observations.
log.p	logical; if TRUE, probabilities p are given as $\log(p)$.
lower.tail	logical; if TRUE (default), probabilities are $P[X \leq x]$, otherwise, $P[X > x]$.

Details

Let $X_i \sim \chi^2(\delta_i, \nu_i)$ be independently distributed non-central chi-squares, where ν_i are the degrees of freedom, and δ_i are the non-centrality parameters. Let p_i be given constants. Suppose

$$Y = \prod_i X_i^{p_i}.$$

Then Y follows a product of chi-squares to power distribution.

Value

dprodchisqpow gives the density, pprodchisqpow gives the distribution function, qprodchisqpow gives the quantile function, and rprodchisqpow generates random deviates.

Invalid arguments will result in return value NaN with a warning.

Note

The PDF, CDF, and quantile function are approximated, via the Edgeworth or Cornish Fisher approximations, which may not be terribly accurate in the tails of the distribution. You are warned.

The distribution parameters are *not* recycled with respect to the x, p, q or n parameters, for, respectively, the density, distribution, quantile and generation functions. This is for simplicity of implementation and performance. It is, however, in contrast to the usual R idiom for dpqr functions.

The PDQ functions are computed by translation of the sum of log chi-squares distribution functions.

Author(s)

Steven E. Pav <shabbychef@gmail.com>

References

Pav, Steven. Moments of the log non-central chi-square distribution. http://arxiv.org/abs/1503.06266

See Also

The sum of log of chi-squares distribution, dsumlogchisq, psumlogchisq, qsumlogchisq, rsumlogchisq, The upsilon distribution, dupsilon, pupsilon, qupsilon, rupsilon. The sum of chi-square powers distribution, dsumchisqpow, psumchisqpow, qsumchisqpow, rsumchisqpow.

Examples

```
df <- c(100,20,10)
ncp <- c(5,3,1)
pow <- c(1,0.5,1)
rvs <- rprodchisqpow(128, df, ncp, pow)
dvs <- dprodchisqpow(rvs, df, ncp, pow)
qvs <- pprodchisqpow(rvs, df, ncp, pow)
pvs <- qprodchisqpow(ppoints(length(rvs)), df, ncp, pow)
```

---

proddnf *The product of multiple doubly non-central F's distribution.*

Description

Density, distribution function, quantile function and random generation for the product of multiple independent doubly non-central F variates.

Usage

```
dproddnf(x, df1, df2, ncp1, ncp2, log = FALSE, order.max=4)

pproddnf(q, df1, df2, ncp1, ncp2, lower.tail = TRUE, log.p = FALSE, order.max=4)

qproddnf(p, df1, df2, ncp1, ncp2, lower.tail = TRUE, log.p = FALSE, order.max=4)

rproddnf(n, df1, df2, ncp1, ncp2)
```

Arguments

`x, q`	vector of quantiles.
`df1, df2`	the vectors of the degrees of freedom for the numerator and denominator. We do *not* recycle these versus the `x,q,p,n`.
`ncp1, ncp2`	the vectors of the non-centrality parameters for the numerator and denominator. We do *not* recycle these versus the `x,q,p,n`.
`log`	logical; if TRUE, densities f are given as $\log(f)$.
`order.max`	the order to use in the approximate density, distribution, and quantile computations, via the Gram-Charlier, Edeworth, or Cornish-Fisher expansion.
`p`	vector of probabilities.
`n`	number of observations.
`log.p`	logical; if TRUE, probabilities p are given as $\log(p)$.
`lower.tail`	logical; if TRUE (default), probabilities are $P[X \leq x]$, otherwise, $P[X > x]$.

Details

Let

$$x_j \sim F\left(\delta_{1,j}, \delta_{2,j}, \nu_{1,j}, \nu_{2,j}\right)$$

be independent doubly non-central F variates with non-centrality parameters $\delta_{i,j}$ and degrees of freedom $\nu_{i,j}$ for $i = 1, 2, \ldots, I$ and $j = 1, 2$. Then

$$Y = \prod_j x_j$$

takes a product of doubly non-central F's distribution. We take the parameters of this distribution as the four I length vectors of the two degrees of freedom and the two non-centrality parameters.

Value

`dproddnf` gives the density, `pproddnf` gives the distribution function, `qproddnf` gives the quantile function, and `rproddnf` generates random deviates.

Invalid arguments will result in return value `NaN` with a warning.

Note

The PDF, CDF, and quantile function are approximated, via the Edgeworth or Cornish Fisher approximations, which may not be terribly accurate in the tails of the distribution. You are warned.

The distribution parameters are *not* recycled with respect to the `x`, `p`, `q` or `n` parameters, for, respectively, the density, distribution, quantile and generation functions. This is for simplicity of implementation and performance. It is, however, in contrast to the usual R idiom for dpqr functions.

The PDQ functions are computed by translation of the sum of log chi-squares distribution functions.

Author(s)

Steven E. Pav <shabbychef@gmail.com>

References

Pav, Steven. Moments of the log non-central chi-square distribution. http://arxiv.org/abs/1503.06266

See Also

The sum of log of chi-squares distribution, `dsumlogchisq`, `psumlogchisq`, `qsumlogchisq`, `rsumlogchisq`. (doubly non-central) F distribution functions, `ddnf`, `pdnf`, `qdnf`, `rdnf`.

Examples

```
df1 <- c(10,20,5)
df2 <- c(1000,500,150)
ncp1 <- c(1,0,2.5)
ncp2 <- c(0,1.5,5)

rv <- rproddnf(500, df1=df1,df2=df2,ncp1=ncp1,ncp2=ncp2)
```

```
d1 <- dproddnf(rv, df1=df1,df2=df2,ncp1=ncp1,ncp2=ncp2)
## Not run:
plot(rv,d1)

## End(Not run)
p1 <- pproddnf(rv, df1=df1,df2=df2,ncp1=ncp1,ncp2=ncp2)
# should be nearly uniform:
## Not run:
plot(ecdf(p1))

## End(Not run)
q1 <- qproddnf(ppoints(length(rv)), df1=df1,df2=df2,ncp1=ncp1,ncp2=ncp2)
## Not run:
qqplot(x=rv,y=q1)

## End(Not run)
```

---

prodnormal *The product of normal random variates.*

Description

Density, distribution function, quantile function and random generation for the distribution of the product of indepdendent normal random variables.

Usage

```
dprodnormal(x, mu, sigma, log = FALSE, order.max=5)

pprodnormal(q, mu, sigma, lower.tail = TRUE, log.p = FALSE, order.max=5)

qprodnormal(p, mu, sigma, lower.tail = TRUE, log.p = FALSE, order.max=5)

rprodnormal(n, mu, sigma)
```

Arguments

x, q	vector of quantiles.
mu	the vector of means. This is recycled against the sigma, but not against the x,q,p,n.
sigma	the vector of standard deviations. This is recycled against the mu, but not against the x,q,p,n.
log	logical; if TRUE, densities f are given as $\log(f)$.
order.max	the order to use in the approximate density, distribution, and quantile computations, via the Gram-Charlier, Edeworth, or Cornish-Fisher expansion.
p	vector of probabilities.
n	number of observations.

log.p	logical; if TRUE, probabilities p are given as $\log(p)$.
lower.tail	logical; if TRUE (default), probabilities are $P[X \leq x]$, otherwise, $P[X > x]$.

Details

Let $Z_i \sim \mathcal{N}\left(\mu_i, \sigma_i^2\right)$ be independently distributed normal variates, with means μ_i and variances σ_i^2. Suppose

$$Y = \prod_i Z_i.$$

Then Y follows a product of normals distribution.

Value

dprodnormal gives the density, pprodnormal gives the distribution function, qprodnormal gives the quantile function, and rprodnormal generates random deviates.

Invalid arguments will result in return value NaN with a warning.

Note

The PDF, CDF, and quantile function are approximated, via the Edgeworth or Cornish Fisher approximations, which may not be terribly accurate in the tails of the distribution. You are warned.

The distribution parameters are *not* recycled with respect to the x, p, q or n parameters, for, respectively, the density, distribution, quantile and generation functions. This is for simplicity of implementation and performance. It is, however, in contrast to the usual R idiom for dpqr functions.

Author(s)

Steven E. Pav <shabbychef@gmail.com>

Examples

```
mu <- c(100,20,10)
sigma <- c(10,50,10)
rvs <- rprodnormal(128, mu, sigma)
dvs <- dprodnormal(rvs, mu, sigma)
qvs <- pprodnormal(rvs, mu, sigma)
pvs <- qprodnormal(ppoints(length(rvs)), mu, sigma)
```

---

runExample *Run Shiny Application*

Description

Runs a shiny application which draws from the given distributions, then illustrates the fidelity of the density, CDF, and quantile functions.

Usage

```
runExample(port=NULL,launch.browser=TRUE,
 host=getOption('shiny.host','127.0.0.1'),display.mode='auto')
```

Arguments

port	The TCP port that the application should listen on. If the port is not specified, and the shiny.port option is set (with options(shiny.port = XX)), then that port will be used. Otherwise, use a random port.
launch.browser	If true, the system's default web browser will be launched automatically after the app is started. Defaults to true in interactive sessions only. This value of this parameter can also be a function to call with the application's URL.
host	The IPv4 address that the application should listen on. Defaults to the shiny.host option, if set, or "127.0.0.1" if not. See Details.
display.mode	The mode in which to display the application. If set to the value "showcase", shows application code and metadata from a DESCRIPTION file in the application directory alongside the application. If set to "normal", displays the application normally. Defaults to "auto", which displays the application in the mode given in its DESCRIPTION file, if any.

Details

Launches shiny applications, and optionally, your system's web browser. Draws are taken from the random variable, and d-d, q-q, and p-p plots are available.

Author(s)

Steven E. Pav <shabbychef@gmail.com>

References

Attali, D. "Supplementing your R package with a shiny app." http://deanattali.com/2015/04/21/r-package-shiny-app/

Examples

```
## Not run:
runExample(launch.browser=TRUE)

## End(Not run)
```

`sadists` *Some Additional Distributions*

Description

Some Additional Distributions.

Details

A collection of distributions which can be approximated via Edgeworth and Cornish-Fisher expansions

Sum of (non-central) chi-square to powers

Let $X_i \sim \chi^2(\delta_i, \nu_i)$ be independently distributed non-central chi-squares, where ν_i are the degrees of freedom, and δ_i are the non-centrality parameters. Let w_i and p_i be given constants. Suppose

$$Y = \sum_i w_i X_i^{p_i}.$$

Then Y follows a weighted sum of chi-squares to power distribution. The special case where all the p_i are one is a 'sum of chi-squares' distribution; The special case where all the p_i are one half is a 'sum of chis' distribution;

Lambda Prime

Introduced by Lecoutre, the lambda prime distribution finds use in inference on the Sharpe ratio under normal returns. Suppose $y \sim \chi^2(\nu)$, and Z is a standard normal.

$$T = Z + t\sqrt{y/\nu}$$

takes a lambda prime distribution with parameters ν, t. A lambda prime random variable can be viewed as a confidence variable on a non-central t because

$$t = \frac{Z' + T}{\sqrt{y/\nu}}$$

Upsilon

The upsilon distribution generalizes the lambda prime to the case of the sum of multiple chi variables. That is, suppose $y_i \sim \chi^2(\nu_i)$ independently and independently of Z, a standard normal. Then

$$T = Z + \sum_i t_i \sqrt{y_i/\nu_i}$$

takes an upsilon distribution with parameter vectors $[\nu_1, \nu_2, \ldots, \nu_k]', [t_1, t_2, ..., t_k]'$.

The upsilon distribution is used in certain tests of the Sharpe ratio for independent observations.

K Prime

Introduced by Lecoutre, the K prime family of distributions generalize the (singly) non-central t, and lambda prime distributions. Suppose $y \sim \chi^2(\nu_1)$, and $x \sim t\left(\nu_2, a\sqrt{y/\nu_1}/b\right)$. Then the random variable

$$T = bx$$

takes a K prime distribution with parameters ν_1, ν_2, a, b. In Lecoutre's terminology, $T \sim K'_{\nu_1,\nu_2}(a, b)$ Equivalently, we can think of

$$T = \frac{bZ + a\sqrt{\chi^2_{\nu_1}/\nu_1}}{\sqrt{\chi^2_{\nu_2}/\nu_2}}$$

where Z is a standard normal, and the normal and the (central) chi-squares are independent of each other. When $a = 0$ we recover a central t distribution; when $\nu_1 = \infty$ we recover a rescaled non-central t distribution; when $b = 0$, we get a rescaled square root of a central F distribution; when $\nu_2 = \infty$, we recover a Lambda prime distribution.

Doubly Noncentral t

The doubly noncentral t distribution generalizes the (singly) noncentral t distribution to the case where the numerator is the square root of a scaled noncentral chi-square distribution. That is, if $X \sim \mathcal{N}(\mu, 1)$ independently of $Y \sim \chi^2(k, \theta)$, then the random variable

$$T = \frac{X}{\sqrt{Y/k}}$$

takes a doubly non-central t distribution with parameters k, μ, θ.

Doubly Noncentral F

The doubly noncentral F distribution generalizes the (singly) noncentral F distribution to the case where the numerator is a scaled noncentral chi-square distribution. That is, if $X \sim \chi^2(n_1, \theta_1)$ independently of $Y \sim \chi^2(n_2, \theta_2)$, then the random variable

$$T = \frac{X/n_1}{Y/n_2}$$

takes a doubly non-central F distribution with parameters $n_1, n_2, \theta_1, \theta_2$.

Parameter recycling

It should be noted that the functions provided by sadists do *not* recycle their distribution parameters against the `x, p, q` or `n` parameters. This is in contrast to the common R idiom, and may cause some confusion. This is mostly for reasons of performance, but also because some of the distributions have vector-valued parameters; recycling over these would require the user to provide *lists* of parameters, which would be unpleasant.

Legal Mumbo Jumbo

Note

This package is maintained as a hobby.

Author(s)

Steven E. Pav <shabbychef@gmail.com>

References

Paolella, Marc. Intermediate Probability: A Computational Approach. Wiley, 2007. http://www.wiley.com/WileyCDA/WileyTitle/productCd-0470026375.html

Lecoutre, Bruno. "Two Useful distributions for Bayesian predictive procedures under normal models." Journal of Statistical Planning and Inference 79, no. 1 (1999): 93-105.

Poitevineau, Jacques, and Lecoutre, Bruno. "Implementing Bayesian predictive procedures: The K-prime and K-square distributions." Computational Statistics and Data Analysis 54, no. 3 (2010): 724-731. http://arxiv.org/abs/1003.4890v1

Walck, C. "HAndbook on Statistical Distributions for experimentalists." 1996. http://www.stat.rice.edu/~dobelman/textfiles/DistributionsHandbook.pdf

---

`sadists-NEWS` *News for package 'sadists'*

Description

History of the 'sadists' package.

Version 0.2.3 (2017-03-19)

- add product of normals distribution.
- move github figures to location CRAN understands.

Version 0.2.2 (2016-03-03)

- work around bad `rchisq` when $df = 0 = ncp$ (?)
- incompatibilities in vignette with ggplot2 release.

Version 0.2.1 (2015-06-12)

- shiny app (h/t Dean Attali).

Version 0.2.0 (2015-04-01)

- add doubly non-central Beta and Eta distributions.
- add (sum of) log chi-square distribution.
- have products of chi-square depend on transform of this distribution.

Initial Version 0.1.0 (2015-03-07)

- first CRAN release.

`sumchisqpow` *The sum of (non-central) chi-squares raised to powers distribution.*

Description

Density, distribution function, quantile function and random generation for the distribution of the weighted sum of non-central chi-squares taken to powers.

Usage

```
dsumchisqpow(x, wts, df, ncp=0, pow=1, log = FALSE, order.max=6)

psumchisqpow(q, wts, df, ncp=0, pow=1, lower.tail = TRUE, log.p = FALSE, order.max=6)

qsumchisqpow(p, wts, df, ncp=0, pow=1, lower.tail = TRUE, log.p = FALSE, order.max=6)

rsumchisqpow(n, wts, df, ncp=0, pow=1)
```

Arguments

`x, q`	vector of quantiles.
`wts`	the vector of weights. This is recycled against the `df, ncp, pow`, but not against the `x,q,p,n`.
`df`	the vector of degrees of freedom. This is recycled against the `wts, ncp, pow`, but not against the `x,q,p,n`.
`ncp`	the vector of non-centrality parameters. This is recycled against the `wts, df, pow`, but not against the `x,q,p,n`.
`pow`	the vector of the power parameters. This is recycled against the `wts, df, ncp`, but not against the `x,q,p,n`.
`log`	logical; if TRUE, densities f are given as $\log(f)$.
`order.max`	the order to use in the approximate density, distribution, and quantile computations, via the Gram-Charlier, Edeworth, or Cornish-Fisher expansion.
`p`	vector of probabilities.
`n`	number of observations.
`log.p`	logical; if TRUE, probabilities p are given as $\log(p)$.
`lower.tail`	logical; if TRUE (default), probabilities are $P[X \leq x]$, otherwise, $P[X > x]$.

Details

Let $X_i \sim \chi^2(\delta_i, \nu_i)$ be independently distributed non-central chi-squares, where ν_i are the degrees of freedom, and δ_i are the non-centrality parameters. Let w_i and p_i be given constants. Suppose

$$Y = \sum_i w_i X_i^{p_i}.$$

Then Y follows a weighted sum of chi-squares to power distribution.

Value

`dsumchisqpow` gives the density, `psumchisqpow` gives the distribution function, `qsumchisqpow` gives the quantile function, and `rsumchisqpow` generates random deviates.

Invalid arguments will result in return value `NaN` with a warning.

Note

The PDF, CDF, and quantile function are approximated, via the Edgeworth or Cornish Fisher approximations, which may not be terribly accurate in the tails of the distribution. You are warned.

The distribution parameters are *not* recycled with respect to the `x, p, q` or `n` parameters, for, respectively, the density, distribution, quantile and generation functions. This is for simplicity of implementation and performance. It is, however, in contrast to the usual R idiom for dpqr functions.

The 'sum of chisquare power' distribution does *not* generalize the 'chi-bar-square' distribution, whose *density* is the sum of chi-square densities.

Author(s)

Steven E. Pav <shabbychef@gmail.com>

See Also

The upsilon distribution, `dupsilon`, `pupsilon`, `qupsilon`, `rupsilon`.

Examples

```
wts <- c(1,-3,4)
df <- c(100,20,10)
ncp <- c(5,3,1)
pow <- c(1,0.5,1)
rvs <- rsumchisqpow(128, wts, df, ncp, pow)
dvs <- dsumchisqpow(rvs, wts, df, ncp, pow)
qvs <- psumchisqpow(rvs, wts, df, ncp, pow)
pvs <- qsumchisqpow(ppoints(length(rvs)), wts, df, ncp, pow)
```

sumlogchisq *The sum of the logs of (non-central) chi-squares distribution.*

Description

Density, distribution function, quantile function and random generation for the distribution of the weighted sum of logs of non-central chi-squares.

Usage

```
dsumlogchisq(x, wts, df, ncp=0, log = FALSE, order.max=6)

psumlogchisq(q, wts, df, ncp=0, lower.tail = TRUE, log.p = FALSE, order.max=6)

qsumlogchisq(p, wts, df, ncp=0, lower.tail = TRUE, log.p = FALSE, order.max=6)

rsumlogchisq(n, wts, df, ncp=0)
```

Arguments

x, q	vector of quantiles.
wts	the vector of weights. This is recycled against the df, ncp, but not against the x,q,p,n.
df	the vector of degrees of freedom. This is recycled against the wts, ncp, but not against the x,q,p,n.
ncp	the vector of non-centrality parameters. This is recycled against the wts, df, but not against the x,q,p,n.
log	logical; if TRUE, densities f are given as $\log(f)$.
order.max	the order to use in the approximate density, distribution, and quantile computations, via the Gram-Charlier, Edeworth, or Cornish-Fisher expansion.
p	vector of probabilities.
n	number of observations.
log.p	logical; if TRUE, probabilities p are given as $\log(p)$.
lower.tail	logical; if TRUE (default), probabilities are $P[X \leq x]$, otherwise, $P[X > x]$.

Details

Let $X_i \sim \chi^2(\delta_i, \nu_i)$ be independently distributed non-central chi-squares, where ν_i are the degrees of freedom, and δ_i are the non-centrality parameters. Let w_i be given constants. Suppose

$$Y = \sum_i w_i \log X_i.$$

Then Y follows a weighted sum of log of chi-squares distribution.

Value

dsumlogchisq gives the density, psumlogchisq gives the distribution function, qsumlogchisq gives the quantile function, and rsumlogchisq generates random deviates.

Invalid arguments will result in return value NaN with a warning.

Note

The PDF, CDF, and quantile function are approximated, via the Edgeworth or Cornish Fisher approximations, which may not be terribly accurate in the tails of the distribution. You are warned.

The distribution parameters are *not* recycled with respect to the x, p, q or n parameters, for, respectively, the density, distribution, quantile and generation functions. This is for simplicity of implementation and performance. It is, however, in contrast to the usual R idiom for dpqr functions.

Author(s)

Steven E. Pav <shabbychef@gmail.com>

References

Pav, Steven. Moments of the log non-central chi-square distribution. http://arxiv.org/abs/1503.06266

See Also

The product of chi-squares to a power, dprodchisqpow, pprodchisqpow, qprodchisqpow, rprodchisqpow.

Examples

```
wts <- c(1,-3,4)
df <- c(100,20,10)
ncp <- c(5,3,1)
rvs <- rsumlogchisq(128, wts, df, ncp)
dvs <- dsumlogchisq(rvs, wts, df, ncp)
qvs <- psumlogchisq(rvs, wts, df, ncp)
pvs <- qsumlogchisq(ppoints(length(rvs)), wts, df, ncp)
```

---

upsilon *The upsilon distribution.*

Description

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

Usage

```
dupsilon(x, df, t, log = FALSE, order.max=6)

pupsilon(q, df, t, lower.tail = TRUE, log.p = FALSE, order.max=6)

qupsilon(p, df, t, lower.tail = TRUE, log.p = FALSE, order.max=6)

rupsilon(n, df, t)
```

Arguments

x, q	vector of quantiles.
df	the degrees of freedom in the chi square. a vector. we do *not* vectorize over this variable.
t	the scaling parameter on the chi. a vector. should be the same length as df. we do *not* vectorize over this variable.
log	logical; if TRUE, densities f are given as $\log(f)$.
order.max	the order to use in the approximate density, distribution, and quantile computations, via the Gram-Charlier, Edeworth, or Cornish-Fisher expansion.
p	vector of probabilities.
n	number of observations.
log.p	logical; if TRUE, probabilities p are given as $\log(p)$.
lower.tail	logical; if TRUE (default), probabilities are $P[X \leq x]$, otherwise, $P[X > x]$.

Details

Suppose $x_i \sim \chi^2(\nu_i)$ independently and independently of Z, a standard normal. Then

$$\Upsilon = Z + \sum_i t_i \sqrt{x_i/\nu_i}$$

takes an upsilon distribution with parameter vectors $[\nu_1, \nu_2, \ldots, \nu_k]', [t_1, t_2, ..., t_k]'$.

The upsilon distribution is used in certain tests of the Sharpe ratio for independent observations, and generalizes the lambda prime distribution, which can be written as $Z + t\sqrt{x/\nu}$.

Value

dupsilon gives the density, pupsilon gives the distribution function, qupsilon gives the quantile function, and rupsilon generates random deviates.

Invalid arguments will result in return value NaN with a warning.

Note

the PDF and CDF are approximated by an Edgeworth expansion; the quantile function is approximated by a Cornish-Fisher expansion.

The PDF, CDF, and quantile function are approximated, via the Edgeworth or Cornish Fisher approximations, which may not be terribly accurate in the tails of the distribution. You are warned.

The distribution parameters are *not* recycled with respect to the `x`, `p`, `q` or `n` parameters, for, respectively, the density, distribution, quantile and generation functions. This is for simplicity of implementation and performance. It is, however, in contrast to the usual R idiom for dpqr functions.

Author(s)

Steven E. Pav <shabbychef@gmail.com>

References

Lecoutre, Bruno. "Another look at confidence intervals for the noncentral t distribution." Journal of Modern Applied Statistical Methods 6, no. 1 (2007): 107–116. http://www.univ-rouen.fr/LMRS/Persopage/Lecoutre/telechargements/Lecoutre_Another_look-JMSAM2007_6(1).pdf

Lecoutre, Bruno. "Two useful distributions for Bayesian predictive procedures under normal models." Journal of Statistical Planning and Inference 79 (1999): 93–105.

Pav, Steven. "Inference on the Sharpe ratio via the upsilon distribution.' Arxiv (2015). http://arxiv.org/abs/1505.00829

See Also

lambda-prime distribution functions, `dlambdap`, `plambdap`, `qlambdap`, `rlambdap`. Sum of chi-squares to power distribution functions, `dsumchisqpow`, `psumchisqpow`, `qsumchisqpow`, `rsumchisqpow`.

Examples

```
mydf <- c(100,30,50)
myt <- c(-1,3,5)
rv <- rupsilon(500, df=mydf, t=myt)
d1 <- dupsilon(rv, df=mydf, t=myt)
## Not run:
plot(rv,d1)

## End(Not run)
p1 <- pupsilon(rv, df=mydf, t=myt)
# should be nearly uniform:
## Not run:
plot(ecdf(p1))

## End(Not run)
q1 <- qupsilon(ppoints(length(rv)),df=mydf,t=myt)
## Not run:
qqplot(x=rv,y=q1)

## End(Not run)
## Not run:
require(SharpeR)
ope <- 252
n.sim <- 500
```

```
n.term <- 3
set.seed(234234)
pp <- replicate(n.sim,{
  # these are population parameters
  a <- rnorm(n.term)
  psi <- 6 * rnorm(length(a)) / sqrt(ope)
  b <- sum(a * psi)
  df <- 100 + ceiling(200 * runif(length(psi)))
  comm <- 1 / sqrt(sum(a^2 / df))
  cdf <- df - 1
  # now independent draws from the SR distribution:
  x <- rsr(length(df), df, zeta=psi, ope=1)
  # now compute a p-value under the true null
  pupsilon(comm * b,df=cdf,t=comm*a*x)
  })
# ought to be uniform:
plot(ecdf(pp))

## End(Not run)
```

Index