Package 'invgamma'

July 2, 2025

Type Package

Title The Inverse Gamma Distribution

Version 1.2

URL https://github.com/dkahle/invgamma, https://www.kahle.io/invgamma/

BugReports https://github.com/dkahle/invgamma/issues

Description Light weight implementation of the standard distribution functions for the inverse gamma distribution, wrapping those for the gamma distribution in the stats package.

License MIT + file LICENSE

RoxygenNote 7.3.2

Encoding UTF-8

Suggests testthat (>= 3.0.0)

Config/testthat/edition 3

NeedsCompilation no

Author David Kahle [aut, cre, cph], James Stamey [aut, cph]

Maintainer David Kahle <david.kahle@gmail.com>

Repository CRAN

Date/Publication 2025-07-02 04:50:02 UTC

Contents

`invchisq` *The Inverse (non-central) Chi-Squared Distribution*

Description

Density, distribution function, quantile function and random generation for the inverse chi-squared distribution.

Usage

```
dinvchisq(x, df, ncp = 0, log = FALSE)

pinvchisq(q, df, ncp = 0, lower.tail = TRUE, log.p = FALSE)

qinvchisq(p, df, ncp = 0, lower.tail = TRUE, log.p = FALSE)

rinvchisq(n, df, ncp = 0)
```

Arguments

`x`, `q`	vector of quantiles.
`df`	degrees of freedom (non-negative, but can be non-integer).
`ncp`	non-centrality parameter (non-negative).
`log`, `log.p`	logical; if `TRUE`, probabilities p are given as log(p).
`lower.tail`	logical; if `TRUE` (default), probabilities are $P[X \leq x]$; if `FALSE` $P[X > x]$.
`p`	vector of probabilities.
`n`	number of observations. If length(n) > 1, the length is taken to be the number required.

Details

The functions `(d/p/q/r)invchisq()` simply wrap those of the standard `(d/p/q/r)chisq()` R implementation, so look at, say, `stats::dchisq()` for details.

See Also

`stats::dchisq()`; these functions just wrap the `(d/p/q/r)chisq()` functions.

Examples

```
s <- seq(0, 3, .01)
plot(s, dinvchisq(s, 3), type = 'l')

f <- function(x) dinvchisq(x, 3)
q <- 2
integrate(f, 0, q)
```

```
(p <- pinvchisq(q, 3))
qinvchisq(p, 3) # = q
mean(rinvchisq(1e5, 3) <= q)



f <- function(x) dinvchisq(x, 3, ncp = 2)
q <- 1.5
integrate(f, 0, q)
(p <- pinvchisq(q, 3, ncp = 2))
qinvchisq(p, 3, ncp = 2) # = q
mean(rinvchisq(1e7, 3, ncp = 2) <= q)
```

---

invexp *The Inverse Exponential Distribution*

Description

Density, distribution function, quantile function and random generation for the inverse exponential distribution.

Usage

```
dinvexp(x, rate = 1, log = FALSE)

pinvexp(q, rate = 1, lower.tail = TRUE, log.p = FALSE)

qinvexp(p, rate = 1, lower.tail = TRUE, log.p = FALSE)

rinvexp(n, rate = 1)
```

Arguments

x, q	vector of quantiles.
rate	degrees of freedom (non-negative, but can be non-integer).
log, log.p	logical; if TRUE, probabilities p are given as log(p).
lower.tail	logical; if TRUE (default), probabilities are $P[X \leq x]$; if FALSE $P[X > x]$.
p	vector of probabilities.
n	number of observations. If length(n) > 1, the length is taken to be the number required.

Details

The functions `(d/p/q/r)invexp()` simply wrap those of the standard `(d/p/q/r)exp()` R implementation, so look at, say, `stats::dexp()` for details.

See Also

`stats::dexp()`; these functions just wrap the `(d/p/q/r)exp()` functions.

Examples

```
s <- seq(0, 10, .01)
plot(s, dinvexp(s, 2), type = 'l')

f <- function(x) dinvexp(x, 2)
q <- 3
integrate(f, 0, q)
(p <- pinvexp(q, 2))
qinvexp(p, 2) # = q
mean(rinvexp(1e5, 2) <= q)

pinvgamma(q, 1, 2)
```

---

invgamma *The Inverse Gamma Distribution*

Description

Density, distribution function, quantile function and random generation for the inverse gamma distribution.

Usage

```
dinvgamma(x, shape, rate = 1, scale = 1/rate, log = FALSE)

pinvgamma(q, shape, rate = 1, scale = 1/rate, lower.tail = TRUE, log.p = FALSE)

qinvgamma(p, shape, rate = 1, scale = 1/rate, lower.tail = TRUE, log.p = FALSE)

rinvgamma(n, shape, rate = 1, scale = 1/rate)
```

Arguments

x, q	vector of quantiles.

shape, rate, scale
: shape, rate, and scale parameters of corresponding gamma distribution. In particular, rate and scale are **not** the rate and scale of the inverse gamma distribution, but of the *gamma* distribution.

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

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

p
: vector of probabilities.

n
: number of observations. If length(n) > 1, the length is taken to be the number required.

Details

The inverse gamma distribution with parameters shape and rate has density

$$f(x) = \frac{rate^{shape}}{\Gamma(shape)} x^{-1-shape} e^{-rate/x}$$

it is the inverse of the standard gamma parameterization in R. If $X \sim InvGamma(shape, rate)$,

$$E[X] = \frac{rate}{shape - 1}$$

when $shape > 1$ and

$$Var(X) = \frac{rate^2}{(shape - 1)^2 (shape - 2)}$$

for $shape > 2$.

The functions (d/p/q/r)invgamma() simply wrap those of the standard (d/p/q/r)gamma() R implementation, so look at, say, stats::dgamma() for details.

See Also

stats::dgamma(); these functions just wrap the (d/p/q/r)gamma() functions.

Examples

```
s <- seq(0, 5, .01)
plot(s, dinvgamma(s, 7, 10), type = 'l')

f <- function(x) dinvgamma(x, 7, 10)
q <- 2
integrate(f, 0, q)
(p <- pinvgamma(q, 7, 10))
qinvgamma(p, 7, 10) # = q
mean(rinvgamma(1e5, 7, 10) <= q)

shape <- 3; rate <- 7
x <- rinvgamma(1e6, shape, rate)
mean(x) # = rate / (shape - 1)
var(x)  # = rate^2 / ( (shape - 1)^2 * (shape - 2) )
```

```
shape <- 7; rate <- 2.01
x <- rinvgamma(1e6, shape, rate)
mean(x) # = rate / (shape - 1)
var(x)  # = rate^2 / ( (shape - 1)^2 * (shape - 2) )

qnorm(log(.25), log.p = TRUE)
qnorm(.25)

qinvgamma(log(.25), shape = shape, rate = rate, log.p = TRUE)
qinvgamma(.25, shape = shape, rate = rate)

## Not run:  `rinvgamma()` warns when shape <= .01

rinvgamma(10, .01, rate) # warns


## End(Not run)
```

Index