Package ‘extraDistr’

June 10, 2019

Type Package
Title Additional Univariate and Multivariate Distributions
Version 1.8.11
Date 2019-06-06
Author Tymoteusz Wolodzko
Maintainer Tymoteusz Wolodzko <twolodzko+extraDistr@gmail.com>
Description Density, distribution function, quantile function
and random generation for a number of univariate
and multivariate distributions. This package implements the
following distributions: Bernoulli, beta-binomial, beta-negative
binomial, beta prime, Bhattacharjee, Birnbaum-Saunders,
bivariate normal, bivariate Poisson, categorical, Dirichlet,
Dirichlet-multinomial, discrete gamma, discrete Laplace,
discrete normal, discrete uniform, discrete Weibull, Frechet,
gamma-Poisson, generalized extreme value, Gompertz,
generalized Pareto, Gumbel, half-Cauchy, half-normal, half-t,
Huber density, inverse chi-squared, inverse-gamma, Kumaraswamy,
Laplace, location-scale t, logarithmic, Lomax, multivariate
hypergeometric, multinomial, negative hypergeometric,
non-standard beta, normal mixture, Poisson mixture, Pareto,
power, reparametrized beta, Rayleigh, shifted Gompertz, Skellam,
slash, triangular, truncated binomial, truncated normal,
truncated Poisson, Tukey lambda, Wald, zero-inflated binomial,
zero-inflated negative binomial, zero-inflated Poisson.
License GPL-2
URL https://github.com/twolodzko/extradistr
BugReports https://github.com/twolodzko/extradistr/issues
Encoding UTF-8
LazyData TRUE
Depends R (>= 3.1.0)
LinkingTo Rcpp
**Imports**  Rcpp

**Suggests**  testthat, LaplacesDemon, VGAM, evd, hoa, skellam, triangle, actuar

**SystemRequirements**  C++11

**RoxygenNote**  6.1.1

**NeedsCompilation**  yes

**Repository**  CRAN

**Date/Publication**  2019-06-10 21:40:05 UTC

---

**R topics documented:**

- Bernoulli .................................................. 3
- BetaBinom ............................................... 4
- BetaNegBinom ........................................... 6
- BetaPrime ............................................... 7
- Bhattacharjee .......................................... 9
- BirnbaumSaunders ..................................... 10
- BivNormal .............................................. 12
- BivPoiss ................................................ 13
- Categorical .............................................. 14
- Dirichlet ............................................... 16
- DirMnoma ............................................... 17
- DiscreteGamma ......................................... 18
- DiscreteLaplace ....................................... 19
- DiscreteNormal ........................................ 21
- DiscreteUniform ....................................... 22
- DiscreteWeibull ....................................... 23
- extraDistr .............................................. 24
- Frechet ................................................ 25
- GammaPoiss ............................................. 26
- GEV ...................................................... 28
- Gompertz ............................................... 29
- GPD ...................................................... 30
- Gumbel .................................................. 32
- HalfCauchy ............................................. 33
- HalfNormal ............................................. 34
- HalfT ..................................................... 35
- Huber .................................................... 36
- InvChiSq ............................................... 38
- InvGamma ............................................... 39
- Kumaraswamy ......................................... 40
- Laplace ................................................ 42
- LocationScaleT ....................................... 43
- LogSeries .............................................. 44
- Lomax .................................................... 45
- MultiHypergeometric ................................ 46
Bernoulli

Description

Probability mass function, distribution function, quantile function and random generation for the Bernoulli distribution.

Usage

\[ \text{dbern}(x, \text{prob} = 0.5, \text{log} = \text{FALSE}) \]

\[ \text{pbern}(q, \text{prob} = 0.5, \text{lower.tail} = \text{TRUE}, \text{log.p} = \text{FALSE}) \]

\[ \text{qbern}(p, \text{prob} = 0.5, \text{lower.tail} = \text{TRUE}, \text{log.p} = \text{FALSE}) \]

\[ \text{rbern}(n, \text{prob} = 0.5) \]
Arguments

- **x, q**: vector of quantiles.
- **prob**: probability of success; \(0 < \text{prob} < 1\).
- **log, log.p**: logical; if TRUE, probabilities p are given as \(\log(p)\).
- **lower.tail**: logical; if TRUE (default), probabilities are \(P[X \leq x]\) otherwise, \(P[X > x]\).
- **p**: vector of probabilities.
- **n**: number of observations. If length(n) > 1, the length is taken to be the number required.

See Also

- **Binomial**

Examples

```r
prop.table(table(rbinom(1e5, 0.5)))
```

BetaBinom

**Beta-binomial distribution**

Description

Probability mass function and random generation for the beta-binomial distribution.

Usage

- **dbbinom(x, size, alpha = 1, beta = 1, log = FALSE)**
- **pbbinom(q, size, alpha = 1, beta = 1, lower.tail = TRUE, log.p = FALSE)**
- **rbbinom(n, size, alpha = 1, beta = 1)**

Arguments

- **x, q**: vector of quantiles.
- **size**: number of trials (zero or more).
- **alpha, beta**: non-negative parameters of the beta distribution.
- **log, log.p**: logical; if TRUE, probabilities p are given as \(\log(p)\).
- **lower.tail**: logical; if TRUE (default), probabilities are \(P[X \leq x]\) otherwise, \(P[X > x]\).
- **n**: number of observations. If length(n) > 1, the length is taken to be the number required.
Details

If \( p \sim \text{Beta}(\alpha, \beta) \) and \( X \sim \text{Binomial}(n, p) \), then \( X \sim \text{BetaBinomial}(n, \alpha, \beta) \).

Probability mass function

\[
f(x) = \binom{n}{x} \frac{B(x + \alpha, n - x + \beta)}{B(\alpha, \beta)}
\]

Cumulative distribution function is calculated using recursive algorithm that employs the fact that

\[
\Gamma(x) = (x - 1)!, \quad \text{and} \quad B(x, y) = \frac{\Gamma(x)\Gamma(y)}{\Gamma(x+y)}
\]

and that \( \binom{n}{k} = \prod_{i=1}^{k} \frac{n+1-i}{i} \). This enables re-writing probability mass function as

\[
f(x) = \left( \prod_{i=1}^{x} \frac{n+1-i}{i} \right) \frac{(\alpha+x-1)! (\beta+n-x-1)!}{(\alpha+\beta+n-1)!} \frac{1}{B(\alpha, \beta)}
\]

what makes recursive updating from \( x \) to \( x+1 \) easy using the properties of factorials

\[
f(x+1) = \left( \prod_{i=1}^{x} \frac{n+1-i}{i} \right) \frac{n+1-x+1}{x+1} \frac{(\alpha+x)! (\beta+n-x)!}{(\alpha+\beta+n)!} \frac{1}{B(\alpha, \beta)}
\]

and let's us efficiently calculate cumulative distribution function as a sum of probability mass functions

\[
F(x) = \sum_{k=0}^{x} f(k)
\]

See Also

Beta, Binomial

Examples

```r
x <- rbinom(1e5, 1000, 5, 13)
x <- 0:1000
hist(x, 100, freq = FALSE)
lines(x-0.5, dbinom(x, 1000, 5, 13), col = "red")
hist(pbinom(x, 1000, 5, 13))
x <- seq(0, 1000, by = 0.1)
plot(ecdf(x))
lines(x, pbinom(x, 1000, 5, 13), col = "red", lwd = 2)
```
**BetaNegBinom**  
**Beta-negative binomial distribution**

**Description**

Probability mass function and random generation for the beta-negative binomial distribution.

**Usage**

\[ \text{dbnbinom}(x, \text{size}, \alpha = 1, \beta = 1, \log = \text{FALSE}) \]

\[ \text{pbnbinom}(q, \text{size}, \alpha = 1, \beta = 1, \text{lower.tail} = \text{TRUE}, \log.p = \text{FALSE}) \]

\[ \text{rbnbinom}(n, \text{size}, \alpha = 1, \beta = 1) \]

**Arguments**

- \( x, q \) vector of quantiles.
- \( \text{size} \) number of trials (zero or more). Must be strictly positive, need not be integer.
- \( \alpha, \beta \) non-negative parameters of the beta distribution.
- \( \log, \log.p \) logical; if TRUE, probabilities \( p \) are given as \( \log(p) \).
- \( \text{lower.tail} \) logical; if TRUE (default), probabilities are \( P[X \leq x] \) otherwise, \( P[X > x] \).
- \( n \) number of observations. If \( \text{length}(n) > 1 \), the length is taken to be the number required.

**Details**

If \( p \sim \text{Beta}(\alpha, \beta) \) and \( X \sim \text{NegBinomial}(r, p) \), then \( X \sim \text{BetaNegBinomial}(r, \alpha, \beta) \).

Probability mass function

\[
f(x) = \frac{\Gamma(r + x) \, B(\alpha + r, \beta + x)}{x! \, \Gamma(r) \, B(\alpha, \beta)}
\]

Cumulative distribution function is calculated using recursive algorithm that employs the fact that \( \Gamma(x) = (x - 1)! \) and \( B(x, y) = \frac{\Gamma(x) \Gamma(y)}{\Gamma(x + y)} \). This enables re-writing probability mass function as

\[
f(x) = \frac{(r + x - 1)! \, (\alpha + r - 1)! \, (\beta + x - 1)!}{x! \, \Gamma(r) \, B(\alpha, \beta) \, (\alpha + \beta + r + x - 1)!}
\]

what makes recursive updating from \( x \) to \( x + 1 \) easy using the properties of factorials

\[
f(x + 1) = \frac{(r + x - 1)! \, (r + x) \, (\alpha + r - 1)! \, (\beta + x - 1)! \, (\beta + x)}{(\alpha + \beta + r + x - 1)! \, (\alpha + \beta + r + x)! \, B(\alpha, \beta)}
\]
and let’s us efficiently calculate cumulative distribution function as a sum of probability mass functions

\[ F(x) = \sum_{k=0}^{x} f(k) \]

See Also

*Beta, NegBinomial*

Examples

```r
x <- rbinom(1e5, 1000, 5, 13)
xx <- 0:1e5
hist(x, 100, freq = FALSE)
lines(xx-.5, dbinom(xx, 1000, 5, 13), col = "red")
hist(pbinom(xx, 1000, 5, 13))
x <- seq(0, 1e5, by = .1)
plot(ecdf(x))
lines(xx, pbinom(xx, 1000, 5, 13), col = "red", lwd = 2)
```

---

**BetaPrime**  
*Beta prime distribution*

**Description**

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

**Usage**

```r
dbetapr(x, shape1, shape2, scale = 1, log = FALSE)
pbetapr(q, shape1, shape2, scale = 1, lower.tail = TRUE, log.p = FALSE)
qbetapr(p, shape1, shape2, scale = 1, lower.tail = TRUE, log.p = FALSE)
rbetapr(n, shape1, shape2, scale = 1)
```
Arguments

- \(x\), \(q\) vector of quantiles.
- \(\text{shape1, shape2}\) non-negative parameters.
- \(\text{scale}\) positive valued scale parameter.
- \(\text{log, log.p}\) logical; if TRUE, probabilities \(p\) are given as \(\log(p)\).
- \(\text{lower.tail}\) logical; if TRUE (default), probabilities are \(P[X \leq x]\) otherwise, \(P[X > x]\).
- \(\text{p}\) vector of probabilities.
- \(n\) number of observations. If length\((n)\) > 1, the length is taken to be the number required.

Details

If \(X \sim \text{Beta}(\alpha, \beta)\), then \(\frac{X}{1-X} \sim \text{BetaPrime}(\alpha, \beta)\).

Probability density function

\[
f(x) = \frac{(x/\sigma)^{\alpha-1}(1 + x/\sigma)^{-\alpha-\beta}}{\text{B}(\alpha, \beta)\sigma}
\]

Cumulative distribution function

\[
F(x) = I_{\frac{x/\sigma}{1+x/\sigma}}(\alpha, \beta)
\]

See Also

- Beta

Examples

```r
x <- rbetapr(1e5, 5, 3, 2)
hist(x, 350, freq = FALSE, xlim = c(0, 100))
curve(dbetapr(x, 5, 3, 2), 0, 100, col = "red", add = TRUE, n = 500)
hist(pbetapr(x, 5, 3, 2))
plot(ecdf(x), xlim = c(0, 100))
curve(pbetapr(x, 5, 3, 2), 0, 100, col = "red", add = TRUE, n = 500)
```
Bhattacharjee distribution

Description
Density, distribution function, and random generation for the Bhattacharjee distribution.

Usage
\begin{verbatim}
dbhatt(x, mu = 0, sigma = 1, a = sigma, log = FALSE)
pbhatt(p, mu = 0, sigma = 1, a = sigma, lower.tail = TRUE, log.p = FALSE)
rbhatt(n, mu = 0, sigma = 1, a = sigma)
\end{verbatim}

Arguments
- **x**, **q**: vector of quantiles.
- **mu**, **sigma**, **a**: location, scale and shape parameters. Scale and shape must be positive.
- **log**, **log.p**: logical; if TRUE, probabilities p are given as log(p).
- **lower.tail**: logical; if TRUE (default), probabilities are \( P[X \leq x] \) otherwise, \( P[X > x] \).
- **n**: number of observations. If length(n) > 1, the length is taken to be the number required.

Details
If \( Z \sim \text{Normal}(0, 1) \) and \( U \sim \text{Uniform}(0, 1) \), then \( Z + U \) follows Bhattacharjee distribution.

Probability density function
\[
f(z) = \frac{1}{2a} \left[ \Phi \left( \frac{x - \mu + a}{\sigma} \right) - \Phi \left( \frac{x - \mu - a}{\sigma} \right) \right]
\]

Cumulative distribution function
\[
F(z) = \frac{\sigma}{2a} \left[ (x - \mu)\Phi \left( \frac{x - \mu + a}{\sigma} \right) - (x - \mu)\Phi \left( \frac{x - \mu - a}{\sigma} \right) + \phi \left( \frac{x - \mu + a}{\sigma} \right) \phi \left( \frac{x - \mu - a}{\sigma} \right) \right]
\]

References
Examples

```r
x <- rbhatt(1e5, 5, 3, 5)
hist(x, 100, freq = FALSE)
curve(dbhatt(x, 5, 3, 5), -20, 20, col = "red", add = TRUE)
hist(pbhatt(x, 5, 3, 5))
plot(ecdf(x))
curve(pbhatt(x, 5, 3, 5), -20, 20, col = "red", lwd = 2, add = TRUE)
```

---

**BirnbaumSaunders**  
**Birnbaum-Saunders (fatigue life) distribution**

Description

Density, distribution function, quantile function and random generation for the Birnbaum-Saunders (fatigue life) distribution.

Usage

```r
dfatigue(x, alpha, beta = 1, mu = 0, log = FALSE)

pfatigue(q, alpha, beta = 1, mu = 0, lower.tail = TRUE, log.p = FALSE)

qfatigue(p, alpha, beta = 1, mu = 0, lower.tail = TRUE, log.p = FALSE)

rfatigue(n, alpha, beta = 1, mu = 0)
```

Arguments

- `x, q` vector of quantiles.
- `alpha, beta, mu` shape, scale and location parameters. Scale and shape must be positive.
- `log, log.p` logical; if TRUE, probabilities p are given as log(p).
- `lower.tail` logical; if TRUE (default), probabilities are $P[X \leq x]$ otherwise, $P[X > x]$.
- `p` vector of probabilities.
- `n` number of observations. If length(n) > 1, the length is taken to be the number required.
Details

Probability density function

\[ f(x) = \left( \frac{\sqrt{x-\mu}}{2\alpha(x-\mu)} + \frac{\beta}{x-\mu} \right) \phi \left( \frac{1}{\alpha} \left( \sqrt{\frac{x-\mu}{\beta}} - \sqrt{\frac{\beta}{x-\mu}} \right) \right) \]

Cumulative distribution function

\[ F(x) = \Phi \left( \frac{1}{\alpha} \left( \sqrt{\frac{x-\mu}{\beta}} - \sqrt{\frac{\beta}{x-\mu}} \right) \right) \]

Quantile function

\[ F^{-1}(p) = \left[ \frac{\alpha}{2} \Phi^{-1}(p) + \sqrt{\left( \frac{\alpha}{2} \Phi^{-1}(p) \right)^2 + 1} \right]^2 \beta + \mu \]

References


Examples

```r
x <- rfatigue(1e5, .5, 2, 5)
hist(x, 100, freq = FALSE)
curve(dfatigue(x, .5, 2, 5), 2, 20, col = "red", add = TRUE)
hist(pdfatigue(x, .5, 2, 5))
plot(ecdf(x))
curve(pdfatigue(x, .5, 2, 5), 2, 20, col = "red", lwd = 2, add = TRUE)
```
Bivariate normal distribution

Description

Density, distribution function and random generation for the bivariate normal distribution.

Usage

dbvnorm(x, y = NULL, mean1 = 0, mean2 = mean1, sd1 = 1, sd2 = sd1, cor = 0, log = FALSE)

rbvnorm(n, mean1 = 0, mean2 = mean1, sd1 = 1, sd2 = sd1, cor = 0)

Arguments

x, y vectors of quantiles; alternatively x may be a two-column matrix (or data.frame) and y may be omitted.

mean1, mean2 vectors of means.

sd1, sd2 vectors of standard deviations.

cor vector of correlations (-1 < cor < 1).

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

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

Details

Probability density function

\[ f(x) = \frac{1}{2\pi \sqrt{1 - \rho^2} \sigma_1 \sigma_2} \exp \left\{ -\frac{1}{2(1 - \rho^2)} \left[ \left( \frac{x_1 - \mu_1}{\sigma_1} \right)^2 - 2\rho \left( \frac{x_1 - \mu_1}{\sigma_1} \right) \left( \frac{x_2 - \mu_2}{\sigma_2} \right) + \left( \frac{x_2 - \mu_2}{\sigma_2} \right)^2 \right] \right\} \]

References


See Also

Normal
Bivariate Poisson distribution

Description

Probability mass function and random generation for the bivariate Poisson distribution.

Usage

```
dbvpois(x, y = NULL, a, b, c, log = FALSE)
```

```
rivpois(n, a, b, c)
```

Arguments

- `x, y`: vectors of quantiles; alternatively `x` may be a two-column matrix (or data.frame) and `y` may be omitted.
- `a, b, c`: positive valued parameters.
- `log`: logical; if TRUE, probabilities `p` are given as log(p).
- `n`: number of observations. If `length(n) > 1`, the length is taken to be the number required.

Details

Probability mass function

\[ f(x) = \exp\left\{-(a + b + c)\right\} \frac{a^x}{x!} \frac{b^y}{y!} \sum_{k=0}^{\min(x,y)} \binom{x}{k} \binom{y}{k} k! \left(\frac{c}{ab}\right)^k \]
Categorical distribution

Description

Probability mass function, distribution function, quantile function and random generation for the categorical distribution.

Usage

dcat(x, prob, log = FALSE)

cat(q, prob, lower.tail = TRUE, log.p = FALSE)

qcat(p, prob, lower.tail = TRUE, log.p = FALSE, labels)

rcat(n, prob, labels)

rcatlp(n, log_prob, labels)

References


See Also

Poisson

Examples

x <- rbvpois(5000, 7, 8, 5)
image(prop.table(table(x[,1], x[,2])))
colMeans(x)
Arguments

- **x**, q: vector of quantiles.
- **prob**, log_prob: vector of length \( m \), or \( m \)-column matrix of non-negative weights (or their logarithms in log_prob).
- **log**, log.p: logical; if TRUE, probabilities \( p \) are given as log(p).
- **lower.tail**: logical; if TRUE (default), probabilities are \( P[X \leq x] \) otherwise, \( P[X > x] \).
- **p**: vector of probabilities.
- **labels**: if provided, labeled factor vector is returned. Number of labels needs to be the same as number of categories (number of columns in prob).
- **n**: number of observations. If length(n) > 1, the length is taken to be the number required.

Details

Probability mass function

\[
Pr(X = k) = \frac{w_k}{\sum_{j=1}^{m} w_j}
\]

Cumulative distribution function

\[
Pr(X \leq k) = \frac{\sum_{i=1}^{k} w_i}{\sum_{j=1}^{m} w_j}
\]

It is possible to sample from categorical distribution parametrized by vector of unnormalized log-probabilities \( \alpha_1, \ldots, \alpha_m \) without leaving the log space by employing the Gumbel-max trick (Maddison, Tarlow and Minka, 2014). If \( g_1, \ldots, g_m \) are samples from Gumbel distribution with cumulative distribution function \( F(g) = \exp(-\exp(-g)) \), then \( k = \arg \max_i \{ g_i + \alpha_i \} \) is a draw from categorical distribution parametrized by vector of probabilities \( p_1, \ldots, p_m \), such that \( p_i = \exp(\alpha_i) / [\sum_{j=1}^{m} \exp(\alpha_j)] \). This is implemented in `rcatlp` function parametrized by vector of log-probabilities `log_prob`.

References


Examples

# Generating 10 random draws from categorical distribution
# with k=3 categories occurring with equal probabilities
# parametrized using a vector

cat(10, c(1/3, 1/3, 1/3))

# or with k=5 categories parametrized using a matrix of probabilities
# (generated from Dirichlet distribution)
p <- rddirichlet(10, c(1, 1, 1, 1, 1))
rcat(10, p)

x <- rcat(1e5, c(0.2, 0.4, 0.3, 0.1))
plot(prop.table(table(x)), type = "h")
lines(0:5, dcat(0:5, c(0.2, 0.4, 0.3, 0.1)), col = "red")

p <- rddirichlet(1, rep(1, 20))
x <- rcat(1e5, matrix(rep(p, 2), nrow = 2, byrow = TRUE))
xx <- 0:21
plot(prop.table(table(x)))
lines(xx, dcat(xx, p), col = "red")

xx <- seq(0, 21, by = 0.01)
plot(ecdf(x))
lines(xx, pcat(xx, p), col = "red", lwd = 2)

pp <- seq(0, 1, by = 0.001)
plot(ecdf(x))
lines(qcat(pp, p), pp, col = "red", lwd = 2)

---

**Dirichlet**

**Dirichlet distribution**

**Description**

Density function, cumulative distribution function and random generation for the Dirichlet distribution.

**Usage**

ddirichlet(x, alpha, log = FALSE)

rddirichlet(n, alpha)

**Arguments**

- **x**  
  k-column matrix of quantiles.
- **alpha**  
  k-values vector or k-column matrix; concentration parameter. Must be positive.
- **log**  
  logical; if TRUE, probabilities p are given as log(p).
- **n**  
  number of observations. If length(n) > 1, the length is taken to be the number required.

**Details**

Probability density function

\[
f(x) = \frac{\Gamma(\sum_k \alpha_k)}{\prod_k \Gamma(\alpha_k)} \prod_k x_k^{\alpha_k - 1}
\]


References

Examples

```c
# Generating 10 random draws from Dirichlet distribution
# parametrized using a vector
rdirichlet(10, c(1, 1, 1, 1))

# or parametrized using a matrix where each row
# is a vector of parameters
alpha <- matrix(c(1, 1, 1, 1:3, 7:9), ncol = 3, byrow = TRUE)
rdirichlet(10, alpha)
```

<table>
<thead>
<tr>
<th>DirMnom</th>
<th>Dirichlet-multinomial (multivariate Polya) distribution</th>
</tr>
</thead>
</table>

Description
Density function, cumulative distribution function and random generation for the Dirichlet-multinomial (multivariate Polya) distribution.

Usage
```
ddirmnom(x, size, alpha, log = FALSE)
rdirmnom(n, size, alpha)
```

Arguments
- `x` \(k\)-column matrix of quantiles.
- `size` numeric vector; number of trials (zero or more).
- `alpha` \(k\)-values vector or \(k\)-column matrix; concentration parameter. Must be positive.
- `log` logical; if TRUE, probabilities p are given as \(\log(p)\).
- `n` number of observations. If `length(n) > 1`, the length is taken to be the number required.
Details

If \((p_1, \ldots, p_k) \sim \text{Dirichlet}(\alpha_1, \ldots, \alpha_k)\) and \((x_1, \ldots, x_k) \sim \text{Multinomial}(n, p_1, \ldots, p_k)\), then \((x_1, \ldots, x_k) \sim \text{DirichletMultinomial}(n, \alpha_1, \ldots, \alpha_k)\).

Probability density function

\[
f(x) = \frac{(n!) \Gamma \left( \sum \alpha_k \right)}{\Gamma \left( n + \sum \alpha_k \right)} \prod_{k=1}^{K} \frac{\Gamma(x_k + \alpha_k)}{(x_k)! \Gamma(\alpha_k)}
\]

References


See Also

Dirichlet, Multinomial

---

### DiscreteGamma

**Discrete gamma distribution**

**Description**

Probability mass function, distribution function and random generation for discrete gamma distribution.

**Usage**

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

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

rdgamma(n, shape, rate = 1, scale = 1/rate)

**Arguments**

- `x`, `q` vector of quantiles.
- `shape`, `scale` shape and scale parameters. Must be positive, scale strictly.
- `rate` an alternative way to specify the scale.
- `log`, `log.p` logical; if TRUE, probabilities p are given as log(p).
- `lower.tail` logical; if TRUE (default), probabilities are \(P[X \leq x]\) otherwise, \(P[X > x]\).
- `n` number of observations. If `length(n) > 1`, the length is taken to be the number required.
Details

Probability mass function of discrete gamma distribution $f_Y(y)$ is defined by discretization of continuous gamma distribution $f_Y(y) = S_X(y) - S_X(y + 1)$ where $S_X$ is a survival function of continuous gamma distribution.

References


See Also

* GammaDist, DiscreteNormal

Examples

```r
x <- rdgamma(1e5, 9, 1)
x <- 0:50
plot(prop.table(table(x)))
lines(xx, ddgamma(xx, 9, 1), col = "red")
hist(pdgamma(x, 9, 1))
plot(ecdf(x))
x <- seq(0, 50, 0.1)
lines(xx, pdgamma(xx, 9, 1), col = "red", lwd = 2, type = "s")
```

DiscreteLaplace  
*Discrete Laplace distribution*

Description

Probability mass, distribution function and random generation for the discrete Laplace distribution parametrized by location and scale.

Usage

```r
ddlaplace(x, location, scale, log = FALSE)
pdlaplace(q, location, scale, lower.tail = TRUE, log.p = FALSE)
rdlaplace(n, location, scale)
```
Arguments

- **x**, **q**: vector of quantiles.
- **location**: location parameter.
- **scale**: scale parameter; $0 < \text{scale} < 1$.
- **log**, **log.p**: logical; if TRUE, probabilities p are given as log(p).
- **lower.tail**: logical; if TRUE (default), probabilities are $P[X \leq x]$ otherwise, $P[X > x]$.
- **n**: number of observations. If length(n) > 1, the length is taken to be the number required.

Details

If $U \sim \text{Geometric}(1 - p)$ and $V \sim \text{Geometric}(1 - p)$, then $U - V \sim \text{DiscreteLaplace}(p)$, where geometric distribution is related to discrete Laplace distribution in similar way as exponential distribution is related to Laplace distribution.

Probability mass function

$$f(x) = \frac{1 - p}{1 + p} p^{|x - \mu|}$$

Cumulative distribution function

$$F(x) = \begin{cases} \frac{p^{-|x - \mu|}}{1 + p} & x < 0 \\ 1 - \frac{p^{-|x - \mu| + 1}}{1 + p} & x \geq 0 \end{cases}$$

References


Examples

```r
p <- 0.45
x <- rdlaplace(1e5, 0, p)
xx <- seq(-200, 200, by = 1)
plot(prop.table(table(x)))
lines(xx, dlaplace(xx, 0, p), col = "red")
hist(dlaplace(x, 0, p))
plot(ecdf(x))
lines(xx, pdlaplace(xx, 0, p), col = "red", type = "s")
```
DiscreteNormal

Discrete normal distribution

Description

Probability mass function, distribution function and random generation for discrete normal distribution.

Usage

ddnorm(x, mean = 0, sd = 1, log = FALSE)
pdnorm(q, mean = 0, sd = 1, lower.tail = TRUE, log.p = FALSE)
rdnorm(n, mean = 0, sd = 1)

Arguments

x, q vector of quantiles.
mean vector of means.
sd vector of standard deviations.
log, log.p logical; if TRUE, probabilities p are given as log(p).
lower.tail logical; if TRUE (default), probabilities are \( P[X \leq x] \) otherwise, \( P[X > x] \).
n number of observations. If length(n) > 1, the length is taken to be the number required.

Details

Probability mass function

\[
f(x) = \Phi \left( \frac{x - \mu + 1}{\sigma} \right) - \Phi \left( \frac{x - \mu}{\sigma} \right)
\]

Cumulative distribution function

\[
F(x) = \Phi \left( \frac{\lfloor x \rfloor + 1 - \mu}{\sigma} \right)
\]

References


See Also

Normal
Examples

```r
x <- rdnorm(1e5, 0, 3)
xz <- -15:15
plot(prop.table(table(x)))
lines(xz, ddnorm(xz, 0, 3), col = "red")
hist(pdnorm(x, 0, 3))
plot(ecdf(x))
xz <- seq(-15, 15, 0.1)
lines(xz, pdnorm(xz, 0, 3), col = "red", lwd = 2, type = "s")
```

DiscreteUniform

**Discrete uniform distribution**

Description

Probability mass function, distribution function, quantile function and random generation for the discrete uniform distribution.

Usage

- `ddunif(x, min, max, log = FALSE)`
- `pdunif(q, min, max, lower.tail = TRUE, log.p = FALSE)`
- `qdunif(p, min, max, lower.tail = TRUE, log.p = FALSE)`
- `rdunif(n, min, max)`

Arguments

- `x, q` vector of quantiles.
- `min, max` lower and upper limits of the distribution. Must be finite.
- `log, log.p` logical; if TRUE, probabilities p are given as log(p).
- `lower.tail` logical; if TRUE (default), probabilities are \( P[X \leq x] \) otherwise, \( P[X > x] \).
- `p` vector of probabilities.
- `n` number of observations. If `length(n) > 1`, the length is taken to be the number required.

Details

If `min == max`, then discrete uniform distribution is a degenerate distribution.
**Examples**

```r
dx <- rdunif(1e5, 1, 10)
xx <- -1:11
plot(prop.table(table(xx)), type = “h”)
lines(xx, dddunif(xx, 1, 10), col = “red”)
hist(ddunif(xx, 1, 10))
xx <- seq(-1, 11, by = 0.01)
plot(ecdf(xx))
lines(xx, pdunif(xx, 1, 10), col = “red”)
```

---

**DiscreteWeibull**  
*Discrete Weibull distribution (type I)*

**Description**

Density, distribution function, quantile function and random generation for the discrete Weibull (type I) distribution.

**Usage**

```r
ddweibull(x, shape1, shape2, log = FALSE)
pdweibull(q, shape1, shape2, lower.tail = TRUE, log.p = FALSE)
qdweibull(p, shape1, shape2, lower.tail = TRUE, log.p = FALSE)
rdweibull(n, shape1, shape2)
```

**Arguments**

- `x, q` vector of quantiles.
- `shape1, shape2` parameters (named q, \( \beta \)). Values of shape2 need to be positive and \( 0 < \text{shape1} < 1 \).
- `log, log.p` logical; if TRUE, probabilities p are given as \log(p)\.
- `lower.tail` logical; if TRUE (default), probabilities are \( P[X \leq x] \) otherwise, \( P[X > x] \).
- `p` vector of probabilities.
- `n` number of observations. If `length(n) > 1`, the length is taken to be the number required.

**Details**

Probability mass function

\[
f(x) = q^x - q^{(x+1)\beta}
\]

Cumulative distribution function

\[
F(x) = 1 - q^{(x+1)\beta}
\]
Quantile function

\[ F^{-1}(p) = \left\lceil \left( \frac{\log(1 - p)}{\log(q)} \right)^{1/\beta} - 1 \right\rceil \]

References


See Also

Weibull

Examples

```r
x <- rdweibull(1e5, 0.32, 1)
x <- seq(-2, 100, by = 1)
plot(prop.table(table(x)), type = "h")
lines(x, ddweibull(x, .32, 1), col = "red")

# Notice: distribution of F(X) is far from uniform:
hist(pdweibull(x, .32, 1), 50)

plot(ecdf(x))
lines(x, pdweibull(x, .32, 1), col = "red", lwd = 2, type = "s")
```

Description

Density, distribution function, quantile function and random generation for a number of univariate and multivariate distributions.

Details

This package follows naming convention that is consistent with base R, where density (or probability mass) functions, distribution functions, quantile functions and random generation functions names are followed by `d`, `p`, `q`, and `r` prefixes.

Behaviour of the functions is consistent with base R, where for not valid parameters values NaN’s are returned, while for values beyond function support 0’s are returned (e.g. for non-integers in discrete distributions, or for negative values in functions with non-negative support).
All the functions vectorized and coded in C++ using **Rcpp**.

<table>
<thead>
<tr>
<th>Frechet</th>
<th>Frechet distribution</th>
</tr>
</thead>
</table>

**Description**

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

**Usage**

- `dfrechet(x, lambda = 1, mu = 0, sigma = 1, log = FALSE)`
- `pfrechet(q, lambda = 1, mu = 0, sigma = 1, lower.tail = TRUE, log.p = FALSE)`
- `qfrechet(p, lambda = 1, mu = 0, sigma = 1, lower.tail = TRUE, log.p = FALSE)`
- `rfrechet(n, lambda = 1, mu = 0, sigma = 1)`

**Arguments**

- `x, q` vector of quantiles.
- `lambda, sigma, mu` shape, scale, and location parameters. Scale and shape must be positive.
- `log, log.p` logical; if TRUE, probabilities `p` are given as `log(p)`.
- `lower.tail` logical; if TRUE (default), probabilities are \( P[X \leq x] \) otherwise, \( P[X > x] \).
- `p` vector of probabilities.
- `n` number of observations. If `length(n) > 1`, the length is taken to be the number required.

**Details**

Probability density function

\[
f(x) = \frac{\lambda}{\sigma} \left( \frac{x - \mu}{\sigma} \right)^{-1-\lambda} \exp \left( - \left( \frac{x - \mu}{\sigma} \right)^{-\lambda} \right)
\]

Cumulative distribution function

\[
F(x) = \exp \left( - \left( \frac{x - \mu}{\sigma} \right)^{-\lambda} \right)
\]

Quantile function

\[
F^{-1}(p) = \mu + \sigma - \log(p)^{-1/\lambda}
\]
References


Examples

```r
x <- rfrechet(1e5, 5, 2, 1.5)
xx <- seq(0, 1000, by = 0.1)
hist(x, 200, freq = FALSE)
lines(xx, dfrechet(xx, 5, 2, 1.5), col = "red")
hist(pfrechet(x, 5, 2, 1.5))
plot(ecdf(x))
lines(xx, pfrechet(xx, 5, 2, 1.5), col = "red", lwd = 2)
```

---

**GammaPoiss**

**Gamma-Poisson distribution**

Description

Probability mass function and random generation for the gamma-Poisson distribution.

Usage

```r
dgpois(x, shape, rate, scale = 1/rate, log = FALSE)
pgpois(q, shape, rate, scale = 1/rate, lower.tail = TRUE,
      log.p = FALSE)
rgpois(n, shape, rate, scale = 1/rate)
```

Arguments

- `x, q` vector of quantiles.
- `shape, scale` shape and scale parameters. Must be positive, scale strictly.
- `rate` an alternative way to specify the scale.
- `log, log.p` logical; if TRUE, probabilities p are given as log(p).
- `lower.tail` logical; if TRUE (default), probabilities are $P[X \leq x]$ otherwise, $P[X > x]$.
- `n` number of observations. If `length(n) > 1`, the length is taken to be the number required.
Gamma-Poisson distribution arises as a continuous mixture of Poisson distributions, where the mixing distribution of the Poisson rate $\lambda$ is a gamma distribution. When $X \sim \text{Poisson}(\lambda)$ and $\lambda \sim \text{Gamma}(\alpha, \beta)$, then $X \sim \text{GammaPoisson}(\alpha, \beta)$.

Probability mass function

$$f(x) = \frac{\Gamma(\alpha + x)}{x! \Gamma(\alpha)} \left( \frac{\beta}{1 + \beta} \right)^x \left( 1 - \frac{\beta}{1 + \beta} \right)^\alpha$$

Cumulative distribution function is calculated using recursive algorithm that employs the fact that $\Gamma(x) = (x-1)!$. This enables re-writing probability mass function as

$$f(x) = \frac{(\alpha + x - 1)!}{x! \Gamma(\alpha)} \left( \frac{\beta}{1 + \beta} \right)^x \left( 1 - \frac{\beta}{1 + \beta} \right)^\alpha$$

what makes recursive updating from $x$ to $x + 1$ easy using the properties of factorials

$$f(x + 1) = \frac{(\alpha + x - 1)! (\alpha + x)}{x! (x + 1) \Gamma(\alpha)} \left( \frac{\beta}{1 + \beta} \right)^x \left( 1 - \frac{\beta}{1 + \beta} \right)^\alpha$$

and let’s us efficiently calculate cumulative distribution function as a sum of probability mass functions

$$F(x) = \sum_{k=0}^{x} f(k)$$

See Also

Gamma, Poisson

Examples

```r
x <- rgpois(1e5, 7, 0.002)
xx <- seq(0, 12000, by = 1)
hist(x, 100, freq = FALSE)
lines(xx, dgpois(xx, 7, 0.002), col = "red")
hist(ppois(x, 7, 0.002))
xx <- seq(0, 12000, by = 0.1)
plot(ecdf(x))
lines(xx, pgpois(xx, 7, 0.002), col = "red", lwd = 2)
```
Description
Density, distribution function, quantile function and random generation for the generalized extreme value distribution.

Usage
\begin{verbatim}
dgev(x, mu = 0, sigma = 1, xi = 0, log = FALSE)
pgev(q, mu = 0, sigma = 1, xi = 0, lower.tail = TRUE, log.p = FALSE)
qgev(p, mu = 0, sigma = 1, xi = 0, lower.tail = TRUE, log.p = FALSE)
rgev(n, mu = 0, sigma = 1, xi = 0)
\end{verbatim}

Arguments
- \(x, q\) vector of quantiles.
- \(mu, sigma, xi\) location, scale, and shape parameters. Scale must be positive.
- \(log, log.p\) logical; if TRUE, probabilities p are given as log(p).
- \(lower.tail\) logical; if TRUE (default), probabilities are \(P(X \leq x)\) otherwise, \(P(X > x)\).
- \(p\) vector of probabilities.
- \(n\) number of observations. If length(n) > 1, the length is taken to be the number required.

Details
Probability density function
\[
f(x) = \begin{cases} 
\frac{1}{\sigma} \left(1 + \frac{x - \mu}{\xi \sigma}\right)^{-1}\exp\left(-\left(1 + \frac{x - \mu}{\xi \sigma}\right)^{-1}\xi\right) & \xi \neq 0 \\
\frac{1}{\sigma} \exp\left(-\frac{x - \mu}{\sigma}\right) \exp\left(-\exp\left(-\frac{x - \mu}{\sigma}\right)\right) & \xi = 0
\end{cases}
\]
Cumulative distribution function
\[
F(x) = \begin{cases} 
\exp\left(-\left(1 + \frac{x - \mu}{\xi \sigma}\right)^{1/\xi}\right) & \xi \neq 0 \\
\exp\left(-\exp\left(-\frac{x - \mu}{\sigma}\right)\right) & \xi = 0
\end{cases}
\]
Quantile function
\[
F^{-1}(p) = \begin{cases} 
\mu - \frac{\sigma}{\xi} (1 - (-\log(p))^{1/\xi}) & \xi \neq 0 \\
\mu - \sigma \log(-\log(p)) & \xi = 0
\end{cases}
\]
Gompertz

References


Examples

curve(dgev(x, xi = -1/2), -4, 4, col = "green", ylab = "")
curve(dgev(x, xi = 0), -4, 4, col = "red", add = TRUE)
curve(dgev(x, xi = 1/2), -4, 4, col = "blue", add = TRUE)
legend("topleft", col = c("green", "red", "blue"), lty = 1,
  legend = expression(xi == -1/2, xi == 0, xi == 1/2), bty = "n")

x <- rgev(1e5, 5, 2, .5)
hist(x, 1000, freq = FALSE, xlab = "x")
curve(dgev(x, 5, .5), 0, 50, col = "red", add = TRUE, n = 5000)
hist(gev(x, 5, 2, .5))
plot(ecdf(x), xlab = c(0, 50))
curve(gev(x, 5, 2, .5), 0, 50, col = "red", lwd = 2, add = TRUE)

Gompertz

Gompertz distribution

Description

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

Usage

dgompertz(x, a = 1, b = 1, log = FALSE)

pgompertz(q, a = 1, b = 1, lower.tail = TRUE, log.p = FALSE)

qgompertz(p, a = 1, b = 1, lower.tail = TRUE, log.p = FALSE)

rgompertz(n, a = 1, b = 1)

Arguments

x, q vector of quantiles.

a, b positive valued scale and location parameters.

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

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

p vector of probabilities.

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

Probability density function

\[
f(x) = a \exp \left( bx - \frac{a}{b} (\exp(bx) - 1) \right)
\]

Cumulative distribution function

\[
F(x) = 1 - \exp \left( -\frac{a}{b} (\exp(bx) - 1) \right)
\]

Quantile function

\[
F^{-1}(p) = \frac{1}{b} \log \left( 1 - \frac{b}{a} \log(1 - p) \right)
\]

References


Examples

```r
x <- rgompertz(1e5, 5, 2)
hist(x, 100, freq = FALSE)
curve(dgompertz(x, 5, 2), 0, 1, col = "red", add = TRUE)
hist(pgompertz(x, 5, 2), 0, 1, col = "red", add = TRUE)
plot(ecdf(x))
curve(pgompertz(x, 5, 2), 0, 1, col = "red", lwd = 2, add = TRUE)
```

GPD

**Generalized Pareto distribution**

Description

Density, distribution function, quantile function and random generation for the generalized Pareto distribution.

Usage

```r
dgpd(x, mu = 0, sigma = 1, xi = 0, log = FALSE)
pgpd(q, mu = 0, sigma = 1, xi = 0, lower.tail = TRUE, log.p = FALSE)
qgpd(p, mu = 0, sigma = 1, xi = 0, lower.tail = TRUE, log.p = FALSE)
rgpd(n, mu = 0, sigma = 1, xi = 0)
```
Arguments

\( x, q \) vector of quantiles.
\( \text{mu}, \text{sigma}, \text{xi} \) location, scale, and shape parameters. Scale must be positive.
\( \text{log}, \text{log.p} \) logical; if TRUE, probabilities \( p \) are given as \( \log(p) \).
\( \text{lower.tail} \) logical; if TRUE (default), probabilities are \( P[X \leq x] \) otherwise, \( P[X > x] \).
\( p \) vector of probabilities.
\( n \) number of observations. If \( \text{length}(n) > 1 \), the length is taken to be the number required.

Details

Probability density function

\[
f(x) = \begin{cases} 
\frac{1}{\sigma} \left(1 + \frac{x - \mu}{\sigma \xi}ight)^{-\left(\xi + 1\right)/\xi} & \xi \neq 0 \\
\frac{1}{\sigma} \exp\left(-\frac{x - \mu}{\sigma}\right) & \xi = 0 
\end{cases}
\]

Cumulative distribution function

\[
F(x) = \begin{cases} 
1 - \left(1 + \frac{x - \mu}{\sigma \xi}\right)^{-1/\xi} & \xi \neq 0 \\
1 - \exp\left(-\frac{x - \mu}{\sigma}\right) & \xi = 0 
\end{cases}
\]

Quantile function

\[
F^{-1}(x) = \begin{cases} 
\mu + \sigma \frac{(1 - p)^{-\xi} - 1}{\xi} & \xi \neq 0 \\
\mu - \sigma \log(1 - p) & \xi = 0 
\end{cases}
\]

References


Examples

```r
x <- rgpd(1e5, 5, 2, .1)
hist(x, 100, freq = FALSE, xlim = c(0, 50))
curve(dgpd(x, 5, 2, .1), 0, 50, col = "red", add = TRUE, n = 5000)
hist(pgpd(x, 5, 2, .1))
plot(ecdf(x))
curve(pgpd(x, 5, 2, .1), 0, 50, col = "red", lwd = 2, add = TRUE)
```
Gumbel

Gumbel distribution

Description
Density, distribution function, quantile function and random generation for the Gumbel distribution.

Usage

dgumbel(x, mu = 0, sigma = 1, log = FALSE)
pgumbel(q, mu = 0, sigma = 1, lower.tail = TRUE, log.p = FALSE)
qgumbel(p, mu = 0, sigma = 1, lower.tail = TRUE, log.p = FALSE)
rgumbel(n, mu = 0, sigma = 1)

Arguments
x, q vector of quantiles.
mu, sigma location and scale parameters. Scale must be positive.
log, log.p logical; if TRUE, probabilities p are given as log(p).
lower.tail logical; if TRUE (default), probabilities are \( P[X \leq x] \) otherwise, \( P[X > x] \).
p vector of probabilities.
n number of observations. If length(n) > 1, the length is taken to be the number required.

Details
Probability density function

\[
f(x) = \frac{1}{\sigma} \exp \left( - \left( \frac{x - \mu}{\sigma} + \exp \left( - \frac{x - \mu}{\sigma} \right) \right) \right)
\]

Cumulative distribution function

\[
F(x) = \exp \left( - \exp \left( - \frac{x - \mu}{\sigma} \right) \right)
\]

Quantile function

\[
F^{-1}(p) = \mu - \sigma \log(-\log(p))
\]

References

Examples

```r
x <- rgumbel(1e5, 5, 2)
hist(x, 100, freq = FALSE)
curve(dgumbel(x, 5, 2), 0, 25, col = "red", add = TRUE)
hist(pgumbel(x, 5, 2))
plot(ecdf(x))
curve(pgumbel(x, 5, 2), 0, 25, col = "red", lwd = 2, add = TRUE)
```

---

**HalfCauchy**

**Half-Cauchy distribution**

**Description**

Density, distribution function, quantile function and random generation for the half-Cauchy distribution.

**Usage**

```r
dhcauchy(x, sigma = 1, log = FALSE)
phcauchy(q, sigma = 1, lower.tail = TRUE, log.p = FALSE)
qhcauchy(p, sigma = 1, lower.tail = TRUE, log.p = FALSE)
rhcauchy(n, sigma = 1)
```

**Arguments**

- `x`, `q`: vector of quantiles.
- `sigma`: positive valued scale parameter.
- `log`, `log.p`: logical; if TRUE, probabilities p are given as log(p).
- `lower.tail`: logical; if TRUE (default), probabilities are \( P[X \leq x] \) otherwise, \( P[X > x] \).
- `p`: vector of probabilities.
- `n`: number of observations. If length(n) > 1, the length is taken to be the number required.

**Details**

If \( X \) follows Cauchy centered at 0 and parametrized by scale \( \sigma \), then \(|X|\) follows half-Cauchy distribution parametrized by scale \( \sigma \). Half-Cauchy distribution is a special case of half-t distribution with \( \nu = 1 \) degrees of freedom.
References


See Also

halfT

Examples

x <- rhcauchy(1e5, 2)
hist(x, 2e5, freq = FALSE, xlim = c(0, 100))
curve(dhcauchy(x, 2), 0, 100, col = "red", add = TRUE)
hist(phcauchy(x, 2))
plot(ecdf(x), xlim = c(0, 100))
curve(phcauchy(x, 2), col = "red", lwd = 2, add = TRUE)

**HalfNormal**

*Half-normal distribution*

Description

Density, distribution function, quantile function and random generation for the half-normal distribution.

Usage

dhnorm(x, sigma = 1, log = FALSE)

phnorm(q, sigma = 1, lower.tail = TRUE, log.p = FALSE)

qhnorm(p, sigma = 1, lower.tail = TRUE, log.p = FALSE)

rhnorm(n, sigma = 1)

Arguments

- **x, q** vector of quantiles.
- **sigma** positive valued scale parameter.
- **log, log.p** logical; if TRUE, probabilities p are given as log(p).
- **lower.tail** logical; if TRUE (default), probabilities are \( P[X \leq x] \) otherwise, \( P[X > x] \).
- **p** vector of probabilities.
- **n** number of observations. If length(n) > 1, the length is taken to be the number required.
**HalfT**

Details

If $X$ follows normal distribution centered at 0 and parametrized by scale $\sigma$, then $|X|$ follows half-normal distribution parametrized by scale $\sigma$. Half-t distribution with $\nu = \infty$ degrees of freedom converges to half-normal distribution.

References


See Also

HalfT

Examples

```r
x <- rnorm(1e5, 2)
hist(x, 100, freq = FALSE)
curve(dhnorm(x, 2), 0, 8, col = "red", add = TRUE)
hist(phnorm(x, 2))
plot(ecdf(x))
curve(phnorm(x, 2), 0, 8, col = "red", lwd = 2, add = TRUE)
```

---

**HalfT**

*Half-t distribution*

Description

Density, distribution function, quantile function and random generation for the half-t distribution.

Usage

```r
dht(x, nu, sigma = 1, log = FALSE)
pht(q, nu, sigma = 1, lower.tail = TRUE, log.p = FALSE)
qht(p, nu, sigma = 1, lower.tail = TRUE, log.p = FALSE)
rht(n, nu, sigma = 1)
```
Arguments

- `x, q` vector of quantiles.
- `nu, sigma` positive valued degrees of freedom and scale parameters.
- `log, log.p` logical; if TRUE, probabilities `p` are given as `log(p)`.
- `lower.tail` logical; if TRUE (default), probabilities are $P[X \leq x]$ otherwise, $P[X > x]$.
- `p` vector of probabilities.
- `n` number of observations. If `length(n) > 1`, the length is taken to be the number required.

Details

If $X$ follows t distribution parametrized by degrees of freedom $\nu$ and scale $\sigma$, then $|X|$ follows half-t distribution parametrized by degrees of freedom $\nu$ and scale $\sigma$.

References


See Also

`HalfNormal`, `HalfCauchy`

Examples

```r
x <- rht(1e5, 2, 2)
hist(x, 500, freq = FALSE, xlim = c(0, 100))
curve(dht(x, 2, 2), 0, 100, col = "red", add = TRUE)
hist(pht(x, 2, 2))
plot(ecdf(x), xlim = c(0, 100))
curve(pht(x, 2, 2), 0, 100, col = "red", lwd = 2, add = TRUE)
```

---

Huber

"Huber density" distribution

Description

Density, distribution function, quantile function and random generation for the "Huber density" distribution.
Usage

dhuber(x, mu = 0, sigma = 1, epsilon = 1.345, log = FALSE)

phuber(q, mu = 0, sigma = 1, epsilon = 1.345, lower.tail = TRUE, log.p = FALSE)

qhuber(p, mu = 0, sigma = 1, epsilon = 1.345, lower.tail = TRUE, log.p = FALSE)

rhuber(n, mu = 0, sigma = 1, epsilon = 1.345)

Arguments

x, q vector of quantiles.
mu, sigma, epsilon

location, and scale, and shape parameters. Scale and shape must be positive.

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

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

p vector of probabilities.

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

Details

Huber density is connected to Huber loss and can be defined as:

$$f(x) = \frac{1}{2\sqrt{2\pi} \left( \Phi(k) + \phi(k)/k - \frac{1}{2} \right)} e^{-\rho_k(x)}$$

where

$$\rho_k(x) = \begin{cases} \frac{1}{2}x^2 & |x| \leq k \\ \frac{1}{2}|x| - \frac{1}{2}k^2 & |x| > k \end{cases}$$

References


Examples

```r
x <- rhuber(1e5, 5, 2, 3)
hist(x, 100, freq = FALSE)
curve(dhuber(x, 5, 2, 3), -20, 20, col = "red", add = TRUE, n = 5000)
hist(phuber(x, 5, 2, 3))
```
InvChiSq

Inverse chi-squared and scaled chi-squared distributions

Description

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

Usage

dinvchisq(x, nu, tau, log = FALSE)
pinvchisq(q, nu, tau, lower.tail = TRUE, log.p = FALSE)
qinvchisq(p, nu, tau, lower.tail = TRUE, log.p = FALSE)
rintvchisq(n, nu, tau)

Arguments

x, q vector of quantiles.
nu positive valued shape parameter.
tau positive valued scaling parameter; if provided it returns values for scaled chi-squared distributions.
log, log.p logical; if TRUE, probabilities p are given as log(p).
lower.tail logical; if TRUE (default), probabilities are P[X ≤ x] otherwise, 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

If X follows χ²(ν) distribution, then 1/X follows inverse chi-squared distribution parametrized by ν. Inverse chi-squared distribution is a special case of inverse gamma distribution with parameters α = ν/2 and β = 1/2; or α = ν/2 and β = ντ²/2 for scaled inverse chi-squared distribution.

See Also

Chisquare, GammaDist
Examples

```r
x <- rinvchisq(1e5, 20)
hist(x, 100, freq = FALSE)
curve(dinvchisq(x, 20), 0, 1, n = 501, col = "red", add = TRUE)
hist(pinvchisq(x, 20))
plot(ecdf(x))
curve(pinvchisq(x, 20), 0, 1, n = 501, col = "red", lwd = 2, add = TRUE)

# scaled
x <- rinvchisq(1e5, 10, 5)
hist(x, 100, freq = FALSE)
curve(dinvchisq(x, 10, 5), 0, 150, n = 501, col = "red", add = TRUE)
hist(pinvchisq(x, 10, 5))
plot(ecdf(x))
curve(pinvchisq(x, 10, 5), 0, 150, n = 501, col = "red", lwd = 2, add = TRUE)
```

---

**InvGamma**

*Inverse-gamma distribution*

---

Description

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

Usage

- `dinvgamma(x, alpha, beta = 1, log = FALSE)`
- `pinvgamma(q, alpha, beta = 1, lower.tail = TRUE, log.p = FALSE)`
- `qinvgamma(p, alpha, beta = 1, lower.tail = TRUE, log.p = FALSE)`
- `rinvgamma(n, alpha, beta = 1)`

Arguments

- `x, q` vector of quantiles.
- `alpha, beta` positive valued shape and scale parameters.
- `log, log.p` logical; if TRUE, probabilities p are given as log(p).
- `lower.tail` logical; if TRUE (default), probabilities are $P[X \leq x]$ otherwise, $P[X > x]$.
- `p` vector of probabilities.
- `n` number of observations. If `length(n) > 1`, the length is taken to be the number required.
Details

Probability mass function

\[ f(x) = \frac{\beta^\alpha x^{-\alpha-1} \exp\left(-\frac{\beta}{x}\right)}{\Gamma(\alpha)} \]

Cumulative distribution function

\[ F(x) = \frac{\gamma(\alpha, \frac{\beta}{x})}{\Gamma(\alpha)} \]

References


See Also

GammaDist

Examples

x <- rinvgamma(1e5, 20, 3)
hist(x, 100, freq = FALSE)
curve(dinvgamma(x, 20, 3), 0, 1, col = "red", add = TRUE, n = 5000)
hist(pinvgamma(x, 20, 3))
plot(ecdf(x))
curve(pinvgamma(x, 20, 3), 0, 1, col = "red", lwd = 2, add = TRUE, n = 5000)

Kumaraswamy

Kumaraswamy distribution

Description

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

Usage

dkumar(x, a = 1, b = 1, log = FALSE)

pkumar(q, a = 1, b = 1, lower.tail = TRUE, log.p = FALSE)

qkumar(p, a = 1, b = 1, lower.tail = TRUE, log.p = FALSE)

rkumar(n, a = 1, b = 1)
**Arguments**

- **x, q**  
  vector of quantiles.
- **a, b**  
  positive valued parameters.
- **log, log.p**  
  logical; if TRUE, probabilities p are given as log(p).
- **lower.tail**  
  logical; if TRUE (default), probabilities are \( P[X \leq x] \) otherwise, \( P[X > x] \).
- **p**  
  vector of probabilities.
- **n**  
  number of observations. If length(n) > 1, the length is taken to be the number required.

**Details**

Probability density function

\[
f(x) = abx^{a-1}(1-x^a)^{b-1}
\]

Cumulative distribution function

\[
F(x) = 1 - (1 - x^a)^b
\]

Quantile function

\[
F^{-1}(p) = 1 - (1 - p^{1/b})^{1/a}
\]

**References**


**Examples**

```r
x <- rkumar(1e5, 5, 16)
hist(x, 100, freq = FALSE)
curve(dkumar(x, 5, 16), 0, 1, col = "red", add = TRUE)
hist(pkumar(x, 5, 16))
plot(ecdf(x))
curve(pkumar(x, 5, 16), 0, 1, col = "red", lwd = 2, add = TRUE)
```
Laplace distribution

Description

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

Usage

dlaplace(x, mu = 0, sigma = 1, log = FALSE)

plaplace(q, mu = 0, sigma = 1, lower.tail = TRUE, log.p = FALSE)

qlaplace(p, mu = 0, sigma = 1, lower.tail = TRUE, log.p = FALSE)

rlaplace(n, mu = 0, sigma = 1)

Arguments

x, q vector of quantiles.
mu, sigma location and scale parameters. Scale must be positive.
log, log.p logical; if TRUE, probabilities p are given as log(p).
lower.tail logical; if TRUE (default), probabilities are \( P[X \leq x] \) otherwise, \( P[X > x] \).
p vector of probabilities.
n number of observations. If length(n) > 1, the length is taken to be the number required.

Details

Probability density function

\[
f(x) = \frac{1}{2\sigma} \exp \left( - \left| \frac{x - \mu}{\sigma} \right| \right)
\]

Cumulative distribution function

\[
F(x) = \begin{cases} 
\frac{1}{2} \exp \left( \frac{x - \mu}{\sigma} \right) & x < \mu \\
1 - \frac{1}{2} \exp \left( \frac{x - \mu}{\sigma} \right) & x \geq \mu 
\end{cases}
\]

Quantile function

\[
F^{-1}(p) = \begin{cases} 
\mu + \sigma \log(2p) & p < 0.5 \\
\mu - \sigma \log(2(1 - p)) & p \geq 0.5 
\end{cases}
\]

References


LocationScaleT

Examples

x <- rlaplace(1e5, 5, 16)
hist(x, 100, freq = FALSE)
curve(dlaplace(x, 5, 16), -200, 200, n = 500, col = "red", add = TRUE)
hist(plaplace(x, 5, 16))
plot(ecdf(x))
curve(plaplace(x, 5, 16), -200, 200, n = 500, col = "red", lwd = 2, add = TRUE)

LocationScaleT

Location-scale version of the t-distribution

Description

Probability mass function, distribution function and random generation for location-scale version of the t-distribution. Location-scale version of the t-distribution besides degrees of freedom $\nu$, is parametrized using additional parameters $\mu$ for location and $\sigma$ for scale ($\mu = 0$ and $\sigma = 1$ for standard t-distribution).

Usage

dlst(x, df, mu = 0, sigma = 1, log = FALSE)
plst(q, df, mu = 0, sigma = 1, lower.tail = TRUE, log.p = FALSE)
qlst(p, df, mu = 0, sigma = 1, lower.tail = TRUE, log.p = FALSE)
rlst(n, df, mu = 0, sigma = 1)

Arguments

- x, q: vector of quantiles.
- df: degrees of freedom ($> 0$, maybe non-integer). df = Inf is allowed.
- mu: vector of locations.
- sigma: vector of positive valued scale parameters.
- log, log.p: logical; if TRUE, probabilities p are given as log(p).
- lower.tail: logical; if TRUE (default), probabilities are $P[X \leq x]$ otherwise, $P[X > x]$.
- p: vector of probabilities.
- n: number of observations. If length(n) > 1, the length is taken to be the number required.

See Also

TDist
LogSeries

Examples

```r
x <- rlst(1e5, 1000, 5, 13)
hist(x, 100, ffreq = FALSE)
curve(dlst(x, 1000, 5, 13), -60, 60, col = "red", add = TRUE)
hist(plst(x, 1000, 5, 13))
plot(ecdf(x))
curve(plst(x, 1000, 5, 13), -60, 60, col = "red", lwd = 2, add = TRUE)
```

LogSeries

Logarithmic series distribution

Description

Density, distribution function, quantile function and random generation for the logarithmic series distribution.

Usage

```r
dlgser(x, theta, log = FALSE)
plgser(q, theta, lower.tail = TRUE, log.p = FALSE)
qlgser(p, theta, lower.tail = TRUE, log.p = FALSE)
rlgser(n, theta)
```

Arguments

- `x, q` vector of quantiles.
- `theta` vector; concentration parameter; \(0 < \theta < 1\).
- `log, log.p` logical; if TRUE, probabilities `p` are given as `log(p)`.
- `lower.tail` logical; if TRUE (default), probabilities are \(P[X \leq x]\) otherwise, \(P[X > x]\).
- `p` vector of probabilities.
- `n` number of observations. If `length(n) > 1`, the length is taken to be the number required.

Details

Probability mass function

\[
f(x) = \frac{-1}{\log(1 - \theta)} \frac{\theta^x}{x}
\]

Cumulative distribution function

\[
F(x) = \frac{-1}{\log(1 - \theta)} \sum_{k=1}^{x} \frac{\theta^x}{x}
\]
Quantile function and random generation are computed using algorithm described in Krishnamoorthy (2006).

References


Examples

```r
x <- rlgser(1e5, 0.66)
xx <- seq(0, 100, by = 1)
plot(prop.table(table(xx)), type = "h")
lines(xx, dlgser(xx, 0.66), col = "red")

# Notice: distribution of F(X) is far from uniform:
hist(plgser(x, 0.66), 50)
xx <- seq(0, 100, by = 0.01)
plot(ecdf(x))
lines(xx, plgser(xx, 0.66), col = "red", lwd = 2)
```

Lomax

$Lomax$ distribution

Description

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

Usage

```r
dlomax(x, lambda, kappa, log = FALSE)
plomax(q, lambda, kappa, lower.tail = TRUE, log.p = FALSE)
qlomax(p, lambda, kappa, lower.tail = TRUE, log.p = FALSE)
rlomax(n, lambda, kappa)
```

Arguments

- `x, q` vector of quantiles.
- `lambda, kappa` positive valued parameters.
- `log, log.p` logical; if TRUE, probabilities p are given as log(p).
lower.tail logical; if TRUE (default), probabilities are \( P[X \leq x] \) otherwise, \( P[X > x] \).

\( p \) vector of probabilities.

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

**Details**

Probability density function

\[
f(x) = \frac{\lambda^{\kappa}}{(1 + \lambda x)^{\kappa+1}}
\]

Cumulative distribution function

\[
F(x) = 1 - (1 + \lambda x)^{-\kappa}
\]

Quantile function

\[
F^{-1}(p) = \frac{(1 - p)^{-1/\kappa} - 1}{\lambda}
\]

**Examples**

```r
x <- rlomax(1e5, 5, 16)
hist(x, 100, freq = FALSE)
curve(dlomax(x, 5, 16), 0, 1, col = "red", add = TRUE, n = 5000)
hist(plomax(x, 5, 16), n, 1, col = "red", lwd = 2, add = TRUE)
```
Multinomial

Arguments

- **x**: \(m\)-column matrix of quantiles.
- **n**: \(m\)-length vector or \(m\)-column matrix of numbers of balls in \(m\) colors.
- **k**: the number of balls drawn from the urn.
- **log**: logical; if TRUE, probabilities \(p\) are given as \(\log(p)\).
- **nn**: number of observations. If \(\text{length}(n) > 1\), the length is taken to be the number required.

Details

Probability mass function

\[
f(x) = \frac{\prod_{i=1}^{m} \binom{n_i}{x_i}}{\binom{N}{k}}
\]

The multivariate hypergeometric distribution is a generalization of the hypergeometric distribution. It is used for sampling without replacement \(k\) out of \(N\) marbles in \(m\) colors, where each of the colors appears \(n_i\) times. Where \(k = \sum_{i=1}^{m} x_i\), \(N = \sum_{i=1}^{m} n_i\) and \(k \leq N\).

References


See Also

- Hypergeometric

Examples

```r
# Generating 10 random draws from multivariate hypergeometric
# distribution parametrized using a vector

rmvhyper(10, c(10, 12, 5, 8, 11), 33)
```

Multinomial distribution

Description

Probability mass function and random generation for the multinomial distribution.

Usage

- **dmnom**\(x, \text{size}, \text{prob}, \text{log} = \text{FALSE}\)
- **rmnom**\(n, \text{size}, \text{prob}\)
Arguments

- `x`  
  $k$-column matrix of quantiles.
- `size`  
  numeric vector; number of trials (zero or more).
- `prob`  
  $k$-column numeric matrix; probability of success on each trial.
- `log`  
  logical; if TRUE, probabilities p are given as log(p).
- `n`  
  number of observations. If length(n) > 1, the length is taken to be the number required.

Details

Probability mass function

$$f(x) = \frac{n!}{\prod_{i=1}^{k} x_i} \prod_{i=1}^{k} p_i^{x_i}$$

References


See Also

- `binomial`
- `multinomial`

Examples

```r
# Generating 10 random draws from multinomial distribution
# parametrized using a vector
(x <- rmnom(10, 3, c(1/3, 1/3, 1/3)))

# Results are consistent with dmultinom() from stats:
all.equal(dmultinom(x[1,], 3, c(1/3, 1/3, 1/3)),
          dmmnom(x[1, , drop = FALSE], 3, c(1/3, 1/3, 1/3)))
```

Description

Probability mass function, distribution function, quantile function and random generation for the negative hypergeometric distribution.
Usage

\[ \text{dnhyper}(x, n, m, r, \text{log} = \text{FALSE}) \]
\[ \text{pnhyper}(q, n, m, r, \text{lower.tail} = \text{TRUE}, \text{log.p} = \text{FALSE}) \]
\[ \text{qnhyper}(p, n, m, r, \text{lower.tail} = \text{TRUE}, \text{log.p} = \text{FALSE}) \]
\[ \text{rnhyper}(\text{nn, n, m, r}) \]

Arguments

- **x, q** vector of quantiles representing the number of balls drawn without replacement from an urn which contains both black and white balls.
- **n** the number of black balls in the urn.
- **m** the number of white balls in the urn.
- **r** the number of white balls that needs to be drawn for the sampling to be stopped.
- **log, log.p** logical; if TRUE, probabilities p are given as log(p).
- **lower.tail** logical; if TRUE (default), probabilities are \( P[X \leq x] \) otherwise, \( P[X > x] \).
- **p** vector of probabilities.
- **nn** number of observations. If length(n) > 1, the length is taken to be the number required.

Details

Negative hypergeometric distribution describes number of balls \( x \) observed until drawing without replacement to obtain \( r \) white balls from the urn containing \( m \) white balls and \( n \) black balls, and is defined as

\[
f(x) = \binom{x-1}{r-1} \binom{m+n-x}{m-r} / \binom{m+n}{n}
\]

The algorithm used for calculating probability mass function, cumulative distribution function and quantile function is based on Fortran program NHYPERG created by Berry and Mielke (1996, 1998). Random generation is done by inverse transform sampling.

References


See Also

Hypergeometric

Examples

x <- rhyper(1e5, 60, 35, 15)
x <- 15:95
plot(prop.table(table(x)))
lines(xx, dhyper(xx, 60, 35, 15), col = "red")
hist(phyper(x, 60, 35, 15))

xx <- seq(0, 100, by = 0.01)
plot(ecdf(x))
lines(xx, phyper(xx, 60, 35, 15), col = "red", lwd = 2)

NormalMix  Mixture of normal distributions

Description

Density, distribution function and random generation for the mixture of normal distributions.

Usage

dmixnorm(x, mean, sd, alpha, log = FALSE)

pmixnorm(q, mean, sd, alpha, lower.tail = TRUE, log.p = FALSE)

rmixnorm(n, mean, sd, alpha)

Arguments

x, q   vector of quantiles.
mean   matrix (or vector) of means.
sd     matrix (or vector) of standard deviations.
alpha  matrix (or vector) of mixing proportions; mixing proportions need to sum up to 1.
log, log.p

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

lower.tail

logical; if TRUE (default), probabilities are \( P[X \leq x] \) otherwise, \( P[X > x] \).

n

number of observations. If \( \text{length}(n) > 1 \), the length is taken to be the number required.

p

vector of probabilities.

Details

Probability density function

\[
f(x) = \alpha_1 f_1(x; \mu_1, \sigma_1) + \ldots + \alpha_k f_k(x; \mu_k, \sigma_k)
\]

Cumulative distribution function

\[
F(x) = \alpha_1 F_1(x; \mu_1, \sigma_1) + \ldots + \alpha_k F_k(x; \mu_k, \sigma_k)
\]

where \( \sum \alpha_i = 1 \).

Examples

```r
x <- rmixnorm(1e5, c(0.5, 3, 6), c(3, 1, 1), c(1/3, 1/3, 1/3))
hist(x, 100, freq = FALSE)
curve(dmixnorm(x, c(0.5, 3, 6), c(3, 1, 1), c(1/3, 1/3, 1/3)),
    -20, 20, n = 500, col = "red", add = TRUE)
hist(pmixnorm(x, c(0.5, 3, 6), c(3, 1, 1), c(1/3, 1/3, 1/3)))
plot(ecdf(x))
curve(pmixnorm(x, c(0.5, 3, 6), c(3, 1, 1), c(1/3, 1/3, 1/3)),
    -20, 20, n = 500, col = "red", lwd = 2, add = TRUE)
```

---

**NSBeta**

**Non-standard beta distribution**

**Description**

Non-standard form of beta distribution with lower and upper bounds denoted as min and max. By default min=0 and max=1 what leads to standard beta distribution.

**Usage**

```r
dnsbeta(x, shape1, shape2, min = 0, max = 1, log = FALSE)
pnsbeta(q, shape1, shape2, min = 0, max = 1, lower.tail = TRUE, log.p = FALSE)
qnsbeta(p, shape1, shape2, min = 0, max = 1, lower.tail = TRUE, log.p = FALSE)
rnsbeta(n, shape1, shape2, min = 0, max = 1)
```
Pareto

Arguments

- **x, q** vector of quantiles.
- **shape1, shape2** non-negative parameters of the Beta distribution.
- **min, max** lower and upper bounds.
- **log, log.p** logical; if TRUE, probabilities p are given as log(p).
- **lower.tail** logical; if TRUE (default), probabilities are \( P[X \leq x] \), otherwise, \( P[X > x] \).
- **p** vector of probabilities.
- **n** number of observations. If length(n) > 1, the length is taken to be the number required.

See Also

Beta

Examples

```r
x <- rnsbeta(1e5, 5, 13, -4, 8)
hist(x, 100, freq = FALSE)
curve(dnsbeta(x, 5, 13, -4, 8), -4, 6, col = "red", add = TRUE)
hist(pnsbeta(x, 5, 13, -4, 8))
plot(ecdf(x))
curve(pnsbeta(x, 5, 13, -4, 8), -4, 6, col = "red", lwd = 2, add = TRUE)
```

Pareto distribution

Description

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

Usage

```r
dpareto(x, a = 1, b = 1, log = FALSE)
ppareto(q, a = 1, b = 1, lower.tail = TRUE, log.p = FALSE)
qpareto(p, a = 1, b = 1, lower.tail = TRUE, log.p = FALSE)
rpareto(n, a = 1, b = 1)
```
Arguments

- **x, q**: vector of quantiles.
- **a, b**: positive valued scale and location parameters.
- **log, log.p**: logical; if TRUE, probabilities p are given as \( \log(p) \).
- **lower.tail**: logical; if TRUE (default), probabilities are \( P[X \leq x] \) otherwise, \( P[X > x] \).
- **p**: vector of probabilities.
- **n**: number of observations. If length(n) > 1, the length is taken to be the number required.

Details

Probability density function

\[
f(x) = \frac{ab^a}{x^{a+1}}
\]

Cumulative distribution function

\[
F(x) = 1 - \left( \frac{b}{x} \right)^a
\]

Quantile function

\[
F^{-1}(p) = \frac{b}{(1 - p)^{1/a}}
\]

References


Examples

```r
x <- rpareto(1e5, 5, 16)
hist(x, 100, freq = FALSE)
curve(dpareto(x, 5, 16), 0, 200, col = "red", add = TRUE)
hist(ppareto(x, 5, 16))
plot(ecdf(x))
curve(ppareto(x, 5, 16), 0, 200, col = "red", lwd = 2, add = TRUE)
```
PoissonMix

*Mixture of Poisson distributions*

**Description**

Density, distribution function and random generation for the mixture of Poisson distributions.

**Usage**


dmixpois(x, lambda, alpha, log = FALSE)

pmixpois(q, lambda, alpha, lower.tail = TRUE, log.p = FALSE)

rmixpois(n, lambda, alpha)

**Arguments**

- **x**, **q**
  - vector of quantiles.
- **lambda**
  - matrix (or vector) of (non-negative) means.
- **alpha**
  - matrix (or vector) of mixing proportions; mixing proportions need to sum up to 1.
- **log**, **log.p**
  - logical; if TRUE, probabilities p are given as log(p).
- **lower.tail**
  - logical; if TRUE (default), probabilities are \( P[X \leq x] \) otherwise, \( P[X > x] \).
- **n**
  - number of observations. If length(n) > 1, the length is taken to be the number required.
- **p**
  - vector of probabilities.

**Details**

Probability density function

\[
  f(x) = \alpha_1 f_1(x; \lambda_1) + \ldots + \alpha_k f_k(x; \lambda_k)
\]

Cumulative distribution function

\[
  F(x) = \alpha_1 F_1(x; \lambda_1) + \ldots + \alpha_k F_k(x; \lambda_k)
\]

where \( \sum_i \alpha_i = 1 \).

Examples

```r
x <- rmixpois(1e5, c(5, 12, 19), c(1/3, 1/3, 1/3))
xx <- seq(-1, 50)
plot(prop.table(table(x)))
lines(xx, dmixpois(xx, c(5, 12, 19), c(1/3, 1/3, 1/3)), col = "red")
hist(pmixpois(x, c(5, 12, 19), c(1/3, 1/3, 1/3)))

xx <- seq(0, 50, by = 0.01)
plot(ecdf(xx))
lines(xx, pmixpois(xx, c(5, 12, 19), c(1/3, 1/3, 1/3)), col = "red", lwd = 2)
```

Description

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

Usage

- `dpower(x, alpha, beta, log = FALSE)`
- `ppower(q, alpha, beta, lower.tail = TRUE, log.p = FALSE)`
- `qpower(p, alpha, beta, lower.tail = TRUE, log.p = FALSE)`
- `rpower(n, alpha, beta)`

Arguments

- `x`, `q` vector of quantiles.
- `alpha`, `beta` parameters.
- `log`, `log.p` logical; if TRUE, probabilities `p` are given as log(p).
- `lower.tail` logical; if TRUE (default), probabilities are $P[X \leq x]$ otherwise, $P[X > x]$.
- `p` vector of probabilities.
- `n` number of observations. If `length(n) > 1`, the length is taken to be the number required.

Details

Probability density function

$$f(x) = \frac{\beta x^{B-1}}{\alpha^B}$$
Cumulative distribution function

\[ F(x) = \frac{x^\beta}{\alpha^\beta} \]

Quantile function

\[ F^{-1}(p) = \alpha p^{1/\beta} \]

Examples

```r
x <- rpower(1e5, 5, 16)
hist(x, 100, freq = FALSE)
curve(dpwr(x, 5, 16), 2, 6, col = "red", add = TRUE, n = 5000)
hist(ppwr(x, 5, 16))
plot(ecdf(x))
curve(ppwr(x, 5, 16), 2, 6, col = "red", lwd = 2, add = TRUE)
```

PropBeta

**Beta distribution of proportions**

Description

Probability mass function, distribution function and random generation for the reparametrized beta distribution.

Usage

```r
dprop(x, size, mean, prior = 0, log = FALSE)
pprop(q, size, mean, prior = 0, lower.tail = TRUE, log.p = FALSE)
qprop(p, size, mean, prior = 0, lower.tail = TRUE, log.p = FALSE)
rprop(n, size, mean, prior = 0)
```

Arguments

- `x`, `q` vector of quantiles.
- `size` non-negative real number; precision or number of binomial trials.
- `mean` mean proportion or probability of success on each trial; \(0 < \text{mean} < 1\).
- `prior` (see below) with `prior = 0` (default) the distribution corresponds to re-parametrized beta distribution used in beta regression. This parameter needs to be non-negative.
- `log`, `log.p` logical; if TRUE, probabilities `p` are given as `log(p)`.
- `lower.tail` logical; if TRUE (default), probabilities are \(P[X \leq x]\) otherwise, \(P[X > x]\).
- `p` vector of probabilities.
- `n` number of observations. If `length(n) > 1`, the length is taken to be the number required.
Details

Beta can be understood as a distribution of \( x = \frac{k}{\phi} \) proportions in \( \phi \) trials where the average proportion is denoted as \( \mu \), so it’s parameters become \( \alpha = \phi \mu \) and \( \beta = \phi (1 - \mu) \) and its density function becomes

\[
f(x) = \frac{x^{\alpha + \pi - 1} (1-x)^{\beta + \pi - 1}}{B(\alpha, \beta)}
\]

where \( \pi \) is a prior parameter, so the distribution is a posterior distribution after observing \( \phi \mu \) successes and \( \phi (1 - \mu) \) failures in \( \phi \) trials with binomial likelihood and symmetric Beta(\( \pi, \pi \)) prior for probability of success. Parameter value \( \pi = 1 \) corresponds to uniform prior; \( \pi = 1/2 \) corresponds to Jeffreys prior; \( \pi = 0 \) corresponds to "uninformative" Haldane prior, this is also the re-parametrized distribution used in beta regression. With \( \pi = 0 \) the distribution can be understood as a continuous analog to binomial distribution dealing with proportions rather then counts. Alternatively \( \phi \) may be understood as precision parameter (as in beta regression).

Notice that in pre-1.8.4 versions of this package, prior was not settable and by default fixed to one, instead of zero. To obtain the same results as in the previous versions, use prior = 1 in each of the functions.

References


See Also

beta, binomial

Examples

```r
x <- rprop(1e5, 100, 0.33)
hist(x, 100, freq = FALSE)
curve(dprop(x, 100, 0.33), 0, 1, col = "red", add = TRUE)
hist(pprop(x, 100, 0.33))
plot(ecdf(x))
curve(pprop(x, 100, 0.33), 0, 1, col = "red", lwd = 2, add = TRUE)

n <- 500
p <- 0.23
k <- rbinom(1e5, n, p)
hist(k/n, freq = FALSE, 100)
curve(dprop(x, n, p), 0, 1, col = "red", add = TRUE, n = 500)
```
Rademacher  
*Random generation from Rademacher distribution*

**Description**
Random generation for the Rademacher distribution (values -1 and +1 with equal probability).

**Usage**
```r
rsign(n)
```

**Arguments**
- `n` number of observations. If `length(n) > 1`, the length is taken to be the number required.

---

Rayleigh  
*Rayleigh distribution*

**Description**
Density, distribution function, quantile function and random generation for the Rayleigh distribution.

**Usage**
```r
drayleigh(x, sigma = 1, log = FALSE)
prayleigh(q, sigma = 1, lower.tail = TRUE, log.p = FALSE)
qrayleigh(p, sigma = 1, lower.tail = TRUE, log.p = FALSE)
rrayleigh(n, sigma = 1)
```

**Arguments**
- `x`, `q` vector of quantiles.
- `sigma` positive valued parameter.
- `log`, `log.p` logical; if TRUE, probabilities `p` are given as `log(p)`.
- `lower.tail` logical; if TRUE (default), probabilities are $P[X \leq x]$ otherwise, $P[X > x]$.
- `p` vector of probabilities.
- `n` number of observations. If `length(n) > 1`, the length is taken to be the number required.
Details

Probability density function

\[ f(x) = \frac{x}{\sigma^2} \exp\left(-\frac{x^2}{2\sigma^2}\right) \]

Cumulative distribution function

\[ F(x) = 1 - \exp\left(-\frac{x^2}{2\sigma^2}\right) \]

Quantile function

\[ F^{-1}(p) = \sqrt{-2\sigma^2 \log(1 - p)} \]

References


Examples

```r
x <- rrayleigh(1e5, 13)
hist(x, 100, freq = FALSE)
curve(drayleigh(x, 13), 0, 60, col = "red", add = TRUE)
hist(prayleigh(x, 13))
plot(ecdf(x))
curve(prayleigh(x, 13), 0, 60, col = "red", lwd = 2, add = TRUE)
```

ShiftGomp  

**Shifted Gompertz distribution**

Description

Density, distribution function, and random generation for the shifted Gompertz distribution.

Usage

\[
dsgomp(x, b, \text{eta}, \text{log} = \text{FALSE})
\]

\[
psgomp(q, b, \text{eta}, \text{lower.tail} = \text{TRUE}, \text{log}.p = \text{FALSE})
\]

\[
rsgomp(n, b, \text{eta})
\]
Arguments

- `x`, `q`: vector of quantiles.
- `b`, `eta`: positive valued scale and shape parameters; both need to be positive.
- `log`, `log.p`: logical; if `TRUE`, probabilities `p` are given as `log(p)`.
- `lower.tail`: logical; if `TRUE` (default), probabilities are `P[X ≤ x]` otherwise, `P[X > x].`
- `n`: number of observations. If `length(n) > 1`, the length is taken to be the number required.

Details

If $X$ follows exponential distribution parametrized by scale $b$ and $Y$ follows reparametrized Gumbel distribution with cumulative distribution function $F(x) = \exp(-\eta \exp(-bx))$ parametrized by scale $b$ and shape $\eta$, then $\max(X, Y)$ follows shifted Gompertz distribution parametrized by scale $b > 0$ and shape $\eta > 0$. The above relation is used by `rsgomp` function for random generation from shifted Gompertz distribution.

Probability density function

$$f(x) = be^{-bx} \exp(-\eta \exp(-bx)) \left[ 1 + \eta(1 - \exp(-bx)) \right]$$

Cumulative distribution function

$$F(x) = (1 - \exp(-bx)) \exp(-\eta \exp(-bx))$$

References


Examples

```r
x <- rsgomp(1e5, 0.4, 1)
hist(x, 50, freq = FALSE)
curve(dsgomp(x, 0.4, 1), 0, 30, col = "red", add = TRUE)
hist(psgomp(x, 0.4, 1))
plot(ecdf(x))
curve(psgomp(x, 0.4, 1), 0, 30, col = "red", lwd = 2, add = TRUE)
```
**Description**

Probability mass function and random generation for the Skellam distribution.

**Usage**

```r
dskellam(x, mu1, mu2, log = FALSE)
rskellam(n, mu1, mu2)
```

**Arguments**

- `x`: vector of quantiles.
- `mu1`, `mu2`: positive valued parameters.
- `log`: logical; if TRUE, probabilities p are given as log(p).
- `n`: number of observations. If `length(n) > 1`, the length is taken to be the number required.

**Details**

If $X$ and $Y$ follow Poisson distributions with means $\mu_1$ and $\mu_2$, than $X - Y$ follows Skellam distribution parametrized by $\mu_1$ and $\mu_2$.

Probability mass function

$$f(x) = e^{-(\mu_1 + \mu_2)} \left(\frac{\mu_1}{\mu_2}\right)^{k/2} I_k(2\sqrt{\mu_1\mu_2})$$

**References**


**Examples**

```r
x <- rskellam(1e5, 5, 13)
x <- -40:40
plot(prop.table(table(x)), type = "h")
lines(xx, dskellam(xx, 5, 13), col = "red")
```
### Slash

**Slash distribution**

**Description**
Probability mass function, distribution function and random generation for slash distribution.

**Usage**

- `dslash(x, mu = 0, sigma = 1, log = FALSE)`
- `pslash(q, mu = 0, sigma = 1, lower.tail = TRUE, log.p = FALSE)`
- `rslash(n, mu = 0, sigma = 1)`

**Arguments**

- **x, q** vector of quantiles.
- **mu** vector of locations
- **sigma** vector of positive valued scale parameters.
- **log, log.p** logical; if TRUE, probabilities p are given as log(p).
- **lower.tail** logical; if TRUE (default), probabilities are $P[X \leq x]$ otherwise, $P[X > x]$.
- **n** number of observations. If length(n) > 1, the length is taken to be the number required.

**Details**

If $Z \sim \text{Normal}(0, 1)$ and $U \sim \text{Uniform}(0, 1)$, then $Z/U$ follows slash distribution.

Probability density function

$$f(x) = \begin{cases} \frac{\phi(0) - \phi(x)}{x^2} & x \neq 0 \\ \frac{1}{2\sqrt{2\pi}} & x = 0 \end{cases}$$

Cumulative distribution function

$$F(x) = \begin{cases} \Phi(x) - \frac{\phi(0) - \phi(x)}{x} & x \neq 0 \\ \frac{1}{2} & x = 0 \end{cases}$$

**Examples**

```r
x <- rslash(1e5, 5, 3)
hist(x, 1e5, freq = FALSE, xlim = c(-100, 100))
curve(dslash(x, 5, 3), -100, 100, col = "red", n = 500, add = TRUE)
hist(pslash(x, 5, 3))
plot(ecdf(x), xlim = c(-100, 100))
curve(pslash(x, 5, 3), -100, 100, col = "red", lwd = 2, n = 500, add = TRUE)
```
**Triangular**

**Description**

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

**Usage**

\[
dtriang(x, a = -1, b = 1, c = (a + b)/2, \log = FALSE)
\]

\[
ptriang(q, a = -1, b = 1, c = (a + b)/2, \text{lower.tail = TRUE, log.p = FALSE})
\]

\[
qtriang(p, a = -1, b = 1, c = (a + b)/2, \text{lower.tail = TRUE, log.p = FALSE})
\]

\[
rtriang(n, a = -1, b = 1, c = (a + b)/2)
\]

**Arguments**

- **x**, **q**: vector of quantiles.
- **a**, **b**, **c**: minimum, maximum and mode of the distribution.
- **log**, **log.p**: logical; if TRUE, probabilities p are given as log(p).
- **lower.tail**: logical; if TRUE (default), probabilities are \(P[X \leq x]\) otherwise, \(P[X > x]\).
- **p**: vector of probabilities.
- **n**: number of observations. If `length(n) > 1`, the length is taken to be the number required.

**Details**

Probability density function

\[
f(x) = \begin{cases} 
\frac{2(x-a)}{(b-a)(c-a)} & x < c \\
\frac{2}{b-a} & x = c \\
\frac{2(b-x)}{(b-a)(b-c)} & x > c 
\end{cases}
\]

Cumulative distribution function

\[
F(x) = \begin{cases} 
\frac{(x-a)^2}{(b-a)(c-a)} & x \leq c \\
1 - \frac{(b-x)^2}{(b-a)(b-c)} & x > c 
\end{cases}
\]

Quantile function

\[
F^{-1}(p) = \begin{cases} 
a + \sqrt{p \times (b-a)(c-a)} & p \leq \frac{c-a}{b-a} \\
b - \sqrt{(1-p)(b-a)(b-c)} & p > \frac{c-a}{b-a} 
\end{cases}
\]

For random generation MINMAX method described by Stein and Keblis (2009) is used.
References

Examples

```r
x <- rtriang(1e5, 5, 7, 6)
hist(x, 100, freq = FALSE)
curve(dtriang(x, 5, 7, 6), 3, 10, n = 500, col = "red", add = TRUE)
hist(ptriang(x, 5, 7, 6))
plot(ecdf(x))
curve(ptriang(x, 5, 7, 6), 3, 10, n = 500, col = "red", lwd = 2, add = TRUE)
```

---

TruncBinom

Truncated binomial distribution

Description
Density, distribution function, quantile function and random generation for the truncated binomial distribution.

Usage

dtbinom(x, size, prob, a = -Inf, b = Inf, log = FALSE)

ptbinom(q, size, prob, a = -Inf, b = Inf, lower.tail = TRUE, log.p = FALSE)

qtbinom(p, size, prob, a = -Inf, b = Inf, lower.tail = TRUE, log.p = FALSE)

rtbinom(n, size, prob, a = -Inf, b = Inf)

Arguments

x, q vector of quantiles.

size number of trials (zero or more).

prob probability of success on each trial.

a, b lower and upper truncation points (a < x <= b).

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

lower.tail logical; if TRUE (default), probabilities are P[X ≤ x] otherwise, P[X > x].
TruncNormal

\[ \text{p} \quad \text{vector of probabilities.} \]
\[ \text{n} \quad \text{number of observations. If length(n) > 1, the length is taken to be the number required.} \]

Examples

```r
x <- rtbinom(1e5, 100, 0.83, 76, 86)
xx <- seq(0, 100)
plot(prop.table(table(x)))
lines(xx, dtbinom(xx, 100, 0.83, 76, 86), col = "red")
hist(ptbinom(x, 100, 0.83, 76, 86))

xx <- seq(0, 100, by = 0.01)
plot(ecdf(x))
lines(xx, ptbinom(xx, 100, 0.83, 76, 86), col = "red", lwd = 2)
uu <- seq(0, 1, by = 0.001)
lines(qtbinom(uu, 100, 0.83, 76, 86), uu, col = "blue", lty = 2)
```

---

**TruncNormal**

*Truncated normal distribution*

**Description**

Density, distribution function, quantile function and random generation for the truncated normal distribution.

**Usage**

```r
dtnorm(x, mean = 0, sd = 1, a = -Inf, b = Inf, log = FALSE)
ptnorm(q, mean = 0, sd = 1, a = -Inf, b = Inf, lower.tail = TRUE, log.p = FALSE)
qtnorm(p, mean = 0, sd = 1, a = -Inf, b = Inf, lower.tail = TRUE, log.p = FALSE)
rtnorm(n, mean = 0, sd = 1, a = -Inf, b = Inf)
```

**Arguments**

- `x, q` vector of quantiles.
- `mean, sd` location and scale parameters. Scale must be positive.
- `a, b` lower and upper truncation points \((a < x <= b, \text{with } a = -\text{Inf} \text{ and } b = \text{Inf by default}).\)
- `log, log.p` logical; if TRUE, probabilities p are given as \(\log(p)\).
lower.tail logical; if TRUE (default), probabilities are $P[X \leq x]$ otherwise, $P[X > x]$.
p vector of probabilities.
n number of observations. If length(n) > 1, the length is taken to be the number required.

**Details**

Probability density function

$$f(x) = \frac{\phi\left(\frac{x - \mu}{\sigma}\right)}{\Phi\left(\frac{b - \mu}{\sigma}\right) - \Phi\left(\frac{a - \mu}{\sigma}\right)}$$

Cumulative distribution function

$$F(x) = \frac{\Phi\left(\frac{x - \mu}{\sigma}\right) - \Phi\left(\frac{a - \mu}{\sigma}\right)}{\Phi\left(\frac{b - \mu}{\sigma}\right) - \Phi\left(\frac{a - \mu}{\sigma}\right)}$$

Quantile function

$$F^{-1}(p) = \Phi^{-1}\left(\Phi\left(\frac{a - \mu}{\sigma}\right) + p \times \left[\Phi\left(\frac{b - \mu}{\sigma}\right) - \Phi\left(\frac{a - \mu}{\sigma}\right)\right]\right)$$

For random generation algorithm described by Robert (1995) is used.

**References**


**Examples**

```r
x <- rtrnorm(1e5, 5, 3, b = 7)
hist(x, 100, freq = FALSE)
curve(dtnorm(x, 5, 3, b = 7), -8, 8, col = "red", add = TRUE)
hist(ptnorm(x, 5, 3, b = 7))
plot(ecdf(x))
curve(ptnorm(x, 5, 3, b = 7), -8, 8, col = "red", lwd = 2, add = TRUE)

R <- 1e5
partmp <- par(mfrow = c(2,4), mar = c(2,2,2,2))
hist(rtrnorm(R), freq = FALSE, main = "", xlab = "", ylab = "")
curve(dtnorm(x), -5, 5, col = "red", add = TRUE)
hist(rtrnorm(R, a = 0), freq = FALSE, main = "", xlab = "", ylab = "")
curve(dtnorm(x, a = 0), -1, 5, col = "red", add = TRUE)
hist(rtrnorm(R, b = 0), freq = FALSE, main = "", xlab = "", ylab = "")
```
curve(dttnorm(x, b = 0), -5, 5, col = "red", add = TRUE)

hist(rttnorm(R, a = 0, b = 1), freq = FALSE, main = "", xlab = ", ylab = "")
curve(dttnorm(x, a = 0, b = 1), -1, 2, col = "red", add = TRUE)

hist(rttnorm(R, a = -1, b = 0), freq = FALSE, main = "", xlab = ", ylab = "")
curve(dttnorm(x, a = -1, b = 0), -2, 2, col = "red", add = TRUE)

hist(rttnorm(R, mean = -6, a = 0), freq = FALSE, main = "", xlab = ", ylab = "")
curve(dttnorm(x, mean = -6, a = 0), -2, 1, col = "red", add = TRUE)

hist(rttnorm(R, mean = 8, b = 0), freq = FALSE, main = "", xlab = ", ylab = "")
curve(dttnorm(x, mean = 8, b = 0), -2, 1, col = "red", add = TRUE)

hist(rttnorm(R, a = 3, b = 5), freq = FALSE, main = "", xlab = ", ylab = "")
curve(dttnorm(x, a = 3, b = 5), 2, 5, col = "red", add = TRUE)

par(partmp)

---

TruncPoisson

Truncated Poisson distribution

Description

Density, distribution function, quantile function and random generation for the truncated Poisson distribution.

Usage

dtpois(x, lambda, a = -Inf, b = Inf, log = FALSE)

ptpois(q, lambda, a = -Inf, b = Inf, lower.tail = TRUE, log.p = FALSE)

qtpois(p, lambda, a = -Inf, b = Inf, lower.tail = TRUE, log.p = FALSE)

rtpois(n, lambda, a = -Inf, b = Inf)

Arguments

x, q vector of quantiles.

lambda vector of (non-negative) means.

a, b lower and upper truncation points (a < x <= b).

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

lower.tail logical; if TRUE (default), probabilities are P[X <= x] otherwise, P[X > x].
TukeyLambda

\[ p \] vector of probabilities.
\[ n \] number of observations. If \( \text{length}(n) > 1 \), the length is taken to be the number required.

References


Examples

```r
x <- rtpois(1e5, 14, 16)
xz <- seq(-1, 50)
plot(prop.table(table(x)))
lines(xz, dtpois(xz, 14, 16), col = "red")
hist(ptpois(x, 14, 16))
xz <- seq(0, 50, by = 0.01)
plot(ecdf(x))
lines(xz, ptpois(xz, 14, 16), col = "red", lwd = 2)

ux <- seq(0, 1, by = 0.001)
lines(qtpois(ux, 14, 16), ux, col = "blue", lty = 2)

# Zero-truncated Poisson
x <- rtpois(1e5, 5, 0)
xz <- seq(-1, 50)
plot(prop.table(table(x)))
lines(xz, dtpois(xz, 5, 0), col = "red")
hist(ptpois(x, 5, 0))
xz <- seq(0, 50, by = 0.01)
plot(ecdf(x))
lines(xz, ptpois(xz, 5, 0), col = "red", lwd = 2)
lines(qtpois(ux, 5, 0), ux, col = "blue", lty = 2)
```

TukeyLambda

**Tukey lambda distribution**

**Description**

Quantile function, and random generation for the Tukey lambda distribution.
TukeyLambda

Usage

\texttt{qtlambda(p, lambda, lower.tail = TRUE, log.p = FALSE)}

\texttt{rtlambd(a(n, lambda)}

Arguments

- \texttt{p} vector of probabilities.
- \texttt{lambda} shape parameter.
- \texttt{lower.tail} logical; if \texttt{TRUE} (default), probabilities are \( P[X \leq x] \) otherwise, \( P[X > x] \).
- \texttt{log.p} logical; if \texttt{TRUE}, probabilities \texttt{p} are given as \log(p).
- \texttt{n} number of observations. If \text{length}(n) > 1, the length is taken to be the number required.

Details

Tukey lambda distribution is a continuous probability distribution defined in terms of its quantile function. It is typically used to identify other distributions.

Quantile function:

\[
F^{-1}(p) = \begin{cases} 
\frac{1}{\lambda} \left[ p^\lambda - (1-p)^\lambda \right] & \lambda \neq 0 \\
\log(p/(1-p)) & \lambda = 0
\end{cases}
\]

References


Examples

\begin{verbatim}
pp = seq(0, 1, by = 0.001)
partmp <- par(mfrow = c(2,3))
plot(qtlambda(pp, -1), pp, type = "l", main = "lambda = -1 (Cauchy)"
plot(qtlambda(pp, 0), pp, type = "l", main = "lambda = 0 (logistic)"
plot(qtlambda(pp, 0.14), pp, type = "l", main = "lambda = 0.14 (normal)"
plot(qtlambda(pp, 0.5), pp, type = "l", main = "lambda = 0.5 (concave)"
plot(qtlambda(pp, 1), pp, type = "l", main = "lambda = 1 (uniform)"
plot(qtlambda(pp, 2), pp, type = "l", main = "lambda = 2 (uniform)"

hist(rtlambda(1e5, -1), freq = FALSE, main = "lambda = -1 (Cauchy)"
hist(rtlambda(1e5, 0), freq = FALSE, main = "lambda = 0 (logistic)"
hist(rtlambda(1e5, 0.14), freq = FALSE, main = "lambda = 0.14 (normal)"
hist(rtlambda(1e5, 0.5), freq = FALSE, main = "lambda = 0.5 (concave)"
hist(rtlambda(1e5, 1), freq = FALSE, main = "lambda = 1 (uniform)"
hist(rtlambda(1e5, 2), freq = FALSE, main = "lambda = 2 (uniform)"
\end{verbatim}
Description
Density, distribution function and random generation for the Wald distribution.

Usage
\[ \text{dwald}(x, \mu, \lambda, \log = \text{FALSE}) \]
\[ \text{pwald}(q, \mu, \lambda, \text{lower.tail = TRUE, log.p = FALSE}) \]
\[ \text{rwald}(n, \mu, \lambda) \]

Arguments
- \( x, q \): vector of quantiles.
- \( \mu, \lambda \): location and shape parameters. Scale must be positive.
- \( \log, \log.p \): logical; if TRUE, probabilities \( p \) are given as \( \log(p) \).
- \( \text{lower.tail} \): logical; if TRUE (default), probabilities are \( P[X \leq x] \) otherwise, \( P[X > x] \).
- \( n \): number of observations. If \( \text{length}(n) > 1 \), the length is taken to be the number required.
- \( p \): vector of probabilities.

Details
Probability density function
\[ f(x) = \sqrt{\frac{\lambda}{2\pi x^3}} \exp \left( \frac{-\lambda(x - \mu)^2}{2\mu^2x} \right) \]

Cumulative distribution function
\[ F(x) = \Phi \left( \sqrt{\frac{\lambda}{x}} \left( \frac{x}{\mu} - 1 \right) \right) + \exp \left( \frac{2\lambda}{\mu} \right) \Phi \left( \sqrt{\frac{\lambda}{x}} \left( \frac{x}{\mu} + 1 \right) \right) \]

Random generation is done using the algorithm described by Michael, Schucany and Haas (1976).

References
Examples

```r
x <- rwald(1e5, 5, 16)
hist(x, 100, freq = FALSE)
curve(dwald(x, 5, 16), 0, 50, col = "red", add = TRUE)
hist(pwald(x, 5, 16))
plot(ecdf(x))
curve(pwald(x, 5, 16), 0, 50, col = "red", lwd = 2, add = TRUE)
```

---

**ZIB**  
Zero-inflated binomial distribution

**Description**
Probability mass function and random generation for the zero-inflated binomial distribution.

**Usage**

```r
dzib(x, size, prob, pi, log = FALSE)
pzib(q, size, prob, pi, lower.tail = TRUE, log.p = FALSE)
qzib(p, size, prob, pi, lower.tail = TRUE, log.p = FALSE)
rzib(n, size, prob, pi)
```

**Arguments**

- `x, q` vector of quantiles.
- `size` number of trials (zero or more).
- `prob` probability of success in each trial. 0 < prob <= 1.
- `pi` probability of extra zeros.
- `log, log.p` logical; if TRUE, probabilities p are given as log(p).
- `lower.tail` logical; if TRUE (default), probabilities are P[X ≤ x] otherwise, 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**
Probability density function

\[
f(x) = \begin{cases} 
\pi + (1 - \pi)(1 - p)^n & x = 0 \\
(1 - \pi)(\binom{n}{x})p^x(1 - p)^{n-x} & x > 0 
\end{cases}
\]
See Also

Binomial

Examples

```r
x <- rzib(1e5, 10, 0.6, 0.33)
xx <- 2:20
plot(prop.table(table(x)), type = "h")
lines(xx, dzib(xx, 10, 0.6, 0.33), col = "red")

xx <- seq(0, 20, by = 0.01)
plot(ecdf(x))
lines(xx, pzib(xx, 10, 0.6, 0.33), col = "red")
```

ZINB

Zero-inflated negative binomial distribution

Description

Probability mass function and random generation for the zero-inflated negative binomial distribution.

Usage

```r
dzinb(x, size, prob, pi, log = FALSE)
pzinb(q, size, prob, pi, lower.tail = TRUE, log.p = FALSE)
qzinb(p, size, prob, pi, lower.tail = TRUE, log.p = FALSE)
rzinb(n, size, prob, pi)
```

Arguments

- `x, q`: vector of quantiles.
- `size`: target for number of successful trials, or dispersion parameter (the shape parameter of the gamma mixing distribution). Must be strictly positive, need not be integer.
- `prob`: probability of success in each trial. $0 < \text{prob} \leq 1$.
- `pi`: probability of extra zeros.
- `log, log.p`: logical; if TRUE, probabilities `p` are given as log(p).
- `lower.tail`: logical; if TRUE (default), probabilities are $P[X \leq x]$ otherwise, $P[X > x]$.
- `p`: vector of probabilities.
- `n`: number of observations. If length(n) > 1, the length is taken to be the number required.
Details

Probability density function

\[
f(x) = \begin{cases} 
\pi + (1 - \pi)p^r & x = 0 \\
(1 - \pi)(x+r-1)p^r(1 - p)^x & x > 0
\end{cases}
\]

See Also

NegBinomial

Examples

```r
x <- rzinb(1e5, 100, 0.6, 0.33)
x <- 0:200
plot(prop.table(table(x)), type = "h")
lines(xx, dzinb(xx, 100, 0.6, 0.33), col = "red")

xx <- seq(0, 200, by = 0.01)
plot(ecdf(x))
lines(xx, pzinb(xx, 100, 0.6, 0.33), col = "red")
```

ZIP

Zero-inflated Poisson distribution

Description

Probability mass function and random generation for the zero-inflated Poisson distribution.

Usage

\[
\begin{align*}
dzip(x, \lambda, \pi, \log = FALSE) \\
pzip(q, \lambda, \pi, \text{lower.tail = TRUE, log.p = FALSE}) \\
qzip(p, \lambda, \pi, \text{lower.tail = TRUE, log.p = FALSE}) \\
rzip(n, \lambda, \pi)
\end{align*}
\]

Arguments

- `x, q` vector of quantiles.
- `lambda` vector of (non-negative) means.
- `pi` probability of extra zeros.
- `log, log.p` logical; if TRUE, probabilities p are given as log(p).
- `lower.tail` logical; if TRUE (default), probabilities are \( P[X \leq x] \) otherwise, \( P[X > x] \).
p  vector of probabilities.
n  number of observations. If length(n) > 1, the length is taken to be the number required.

Details
Probability density function

\[ f(x) = \begin{cases} 
\pi + (1 - \pi)e^{-\lambda} & x = 0 \\
(1 - \pi)\frac{\lambda^x e^{-\lambda}}{x!} & x > 0 
\end{cases} \]

See Also
Poisson

Examples

```r
x <- rzip(1e5, 6, 0.33)
x <- seq(0, 20, by = 0.01)
plot(ecdf(x))
lines(x, pzip(x, 6, 0.33), col = "red")
```

Index

*Topic distribution

Bernoulli, 3
BetaBinom, 4
BetaNegBinom, 6
BetaPrime, 7
Bhattacharjee, 9
BirnbaumSaunders, 10
BivNormal, 12
BivPois, 13
Categorical, 14
Dirichlet, 16
DirMnom, 17
DiscreteGamma, 18
DiscreteLaplace, 19
DiscreteNormal, 21
DiscreteUniform, 22
DiscreteWeibull, 23
Frechet, 25
GammaPois, 26
GEV, 28
Gompertz, 29
GPD, 30
Gumbel, 32
HalfCauchy, 33
HalfNormal, 34
HalfT, 35
Huber, 36
InvChiSq, 38
InvGamma, 39
Kumaraswamy, 40
Laplace, 42
LocationScaleT, 43
LogSeries, 44
Lomax, 45
MultHypergeometric, 46
Multinomial, 47
NegHyper, 48
NormalMix, 50
NSBeta, 51
Pareto, 52
PoissonMix, 54
PowerDist, 55
PropBeta, 56
Rademacher, 58
Rayleigh, 58
ShiftGomp, 59
Skellam, 61
Slash, 62
Triangular, 63
TruncBinom, 64
TruncNormal, 65
TruncPoisson, 67
TukeyLambda, 68
Wald, 70
ZIB, 71
ZINB, 72
ZIP, 73
Bernoulli, 3
Beta, 5, 7, 8, 52
beta, 57
BetaBinom, 4
BetaNegBinom, 6
BetaPrime, 7
Bhattacharjee, 9
Binomial, 4, 5, 48, 72
binomial, 57
BirnbaumSaunders, 10
BivNormal, 12
BivPois, 13
Categorical, 14
Chisquare, 38
dbbinom (BetaBinom), 4
dbern (Bernoulli), 3
dbeta (BetaPrime), 7
dbhattacharjee (Bhattacharjee), 9
dbnbinom (BetaNegBinom), 6
INDEX

dbvnorm (BivNormal), 12
dbypois (BivPois), 13
dcat (Categorical), 14
ddgamma (DiscreteGamma), 18
ddirichlet (Dirichlet), 16
ddimnom (DirMnom), 17
ddlaplace (DiscreteLaplace), 19
ddnorm (DiscreteNormal), 21
dduinf (DiscreteUniform), 22
ddweibull (DiscreteWeibull), 23
dfatigue (BirnbaumSaunders), 10
dfrechet (Frechet), 25
dgev (GEV), 28
dgompertz (Gompertz), 29
dgpd (GPD), 30
dgpois (GammaPois), 26
dgumbel (Gumbel), 32
dhcauchy (HalfCauchy), 33
dlnorm (HalfNormal), 34
dlht (HalfT), 35
dhuber (Huber), 36
dinvchisq (InvChiSq), 38
dinvgamma (InvGamma), 39
Dirichlet, 16, 18
DirMnom, 17
DiscreteGamma, 18
DiscreteLaplace, 19
DiscreteNormal, 19, 21
DiscreteUniform, 22
DiscreteWeibull, 23
dkumar (Kumaraswamy), 40
dlaplace (Laplace), 42
dlgser (LogSeries), 44
dlomax (Lomax), 45
dlstat (LocationScaleT), 43
dmixnorm (NormalMix), 50
dmixpois (PoissonMix), 54
dmnom (Multinomial), 47
dmvhyper (MultiHypergeometric), 46
dnorm (NegHyper), 48
dsnsbeta (NSBeta), 51
dpareto (Pareto), 52
dpower (PowerDist), 55
dprop (PropBeta), 56
drayleigh (Rayleigh), 58
dsgomp (ShiftGomp), 59
dskellam (Skellam), 61
dslash (Slash), 62
dtnorm (TruncNormal), 64
dtnorm (TruncNormal), 65
dtpois (TruncPoisson), 67
dtriang (Triangular), 63
dwald (Wald), 70
dzib (ZIB), 71
dzinb (ZINB), 72
dzip (ZIP), 73
extraDistr, 24
extraDistr-package (extraDistr), 24
Frechet, 25
Gamma, 27
GammaDist, 19, 38, 40
GammaPoiss, 26
GEV, 28
Gompertz, 29
GPD, 30
Gumbel, 32
HalfCauchy, 33, 36
HalfNormal, 34, 36
HalfT, 34, 35, 35
Huber, 36
Hypergeometric, 47, 50
InvChiSq, 38
InvGamma, 39
Kumaraswamy, 40
Laplace, 42
LocationScaleT, 43
LogSeries, 44
Lomax, 45
MultiHypergeometric, 46
Multinomial, 18, 47, 48
NegBinomial, 7, 73
NegHyper, 48
Normal, 12, 21
NormalMix, 50
NSBeta, 51
Pareto, 52
pbbinom (BetaBinom), 4
pbern (Bernoulli), 3
INDEX

pbetapr (BetaPrime), 7
pbhattr (Bhattacharjee), 9
pbnbinom (BetaNegBinom), 6
pcat (Categorical), 14
pdgamma (DiscreteGamma), 18
pdlaplace (DiscreteLaplace), 19
pdnorm (DiscreteNormal), 21
pdunif (DiscreteUniform), 22
pdweibull (DiscreteWeibull), 23
pfatigue (BirnbaumSaunders), 10
pfrechet (Frechet), 25
pgn (Gev), 28
pgompertz (Gompertz), 29
pgpd (GPD), 30
pgpois (GammaPoisson), 26
pgumbel (Gumbel), 32
phcauchy (HalfCauchy), 33
phnorm (HalfNormal), 34
ph (HalfT), 35
phuber (Huber), 36
pinvchisq (InvChisq), 38
pinvgamma (InvGamma), 39
pkumar (Kumaraswamy), 40
plaplace (Laplace), 42
plgser (LogSeries), 44
plomax (Lomax), 45
plst (LocationScaleT), 43
pmixnorm (NormalMix), 50
pmixpois (PoissonMix), 54
pnhyper (NegHyper), 48
pnsbeta (NSBeta), 51
Poisson, 4, 27, 74
PoissonMix, 54
PowerDist, 55
ppareto (Pareto), 52
ppower (PowerDist), 55
pprop (PropBeta), 56
prayleigh (Rayleigh), 58
PropBeta, 56
psgomp (ShiftGomp), 59
pslash (Slash), 62
ptbinom (TruncBinom), 64
ptnorm (TruncNormal), 65
ptpois (TruncPoisson), 67
ptriangular (Triangular), 63
pwald (Wald), 70
pzib (ZIB), 71
pzib (ZINB), 72
pzip (ZIP), 73
qbern (Bernoulli), 3
qbetapr (BetaPrime), 7
qcat (Categorical), 14
qdunif (DiscreteUniform), 22
qdweibull (DiscreteWeibull), 23
qfatigue (BirnbaumSaunders), 10
qfrechet (Frechet), 25
qgev (Gev), 28
qgompertz (Gompertz), 29
qgpd (GPD), 30
qgumbel (Gumbel), 32
qhcauchy (HalfCauchy), 33
qfnorm (HalfNormal), 34
qht (HalfT), 35
qhuber (Huber), 36
qinvchisq (InvChisq), 38
qinvgamma (InvGamma), 39
qkumar (Kumaraswamy), 40
qlaplace (Laplace), 42
qlgser (LogSeries), 44
qlomax (Lomax), 45
qlst (LocationScaleT), 43
qnhyper (NegHyper), 48
qnsbeta (NSBeta), 51
qpareto (Pareto), 52
qpower (PowerDist), 55
qprop (PropBeta), 56
qrayleigh (Rayleigh), 58
qtb-binom (TruncBinom), 64
qtlambda (TukeyLambda), 68
qtnorm (TruncNormal), 65
qtpois (TruncPoisson), 67
qtriangular (Triangular), 63
qzib (ZIB), 71
qzinb (ZINB), 72
qzip (ZIP), 73
Rademacher, 58
Rayleigh, 58
rbbinom (BetaBinom), 4
rbern (Bernoulli), 3
rbetapr (BetaPrime), 7
rbhattr (Bhattacharjee), 9
rbnbinom (BetaNegBinom), 6
rbvnorm (BivNormal), 12
rbvpois (BivPoisson), 13
rcat (Categorical), 14
rcatlp (Categorical), 14
rdgamma (DiscreteGamma), 18
rdgamma (Dirichlet), 16
rdnorm (DirMnorm), 17
rdlaplace (DiscreteLaplace), 19
rdnorm (DiscreteNormal), 21
rdunif (DiscreteUniform), 22
rdweibull (DiscreteWeibull), 23
rfatigue (BirnbaumSaunders), 10
rfrechet (Frechet), 25
rgev (GEV), 28
rgompertz (Gompertz), 29
rgpois (GammaPois), 26
rgumbel (Gumbel), 32
rhcauchy (HalfCauchy), 33
rhnorm (HalfNormal), 34
rh (Halft), 35
rhuber (Huber), 36
rinvchisq (InvChiSq), 38
rinvgamma (InvGamma), 39
rkumar (Kumaraswamy), 40
rlaplace (Laplace), 42
rlgser (LogSeries), 44
rlomax (Lomax), 45
rlst (LocationScaleT), 43
rmixnorm (NormalMix), 50
rmixpois (PoissonMix), 54
rnmnom (Multinomial), 47
rmvhyper (MultiHypergeometric), 46
rnhyper (NegHyper), 48
rnsbeta (NSBeta), 51
rpareto (Pareto), 52
rpower (PowerDist), 55
rprop (PropBeta), 56
rrayleigh (Rayleigh), 58
rsgomp (ShiftGomp), 59
rsign (Rademacher), 58
rskellam (Skellam), 61
rslash (Slash), 62
rtbinom (TruncBinom), 64
rtlambda (TukeyLambda), 68
rtlnorm (TruncNormal), 65
rtpois (TruncPoisson), 67
rttriang (Triangular), 63
rwald (Wald), 70
rzip (ZIP), 73
ShiftGomp, 59
Skellam, 61
Slash, 62
TDist, 43
Triangular, 63
TruncBinom, 64
TruncNormal, 65
TruncPoisson, 67
TukeyLambda, 68
Wald, 70
ZIB, 71
ZINB, 72
ZIP, 73
weibull, 24